A Learning-Based Approach for Vehicle-to-Vehicle Computation Offloading

Xingxia Dai, Zhu Xiao, *Senior Member, IEEE*, Hongbo Jiang, *Senior Member, IEEE*, Hongyang Chen, *Senior Member, IEEE*, Geyong Min, *Member, IEEE*, Schahram Dustdar, *Fellow, IEEE*, and Jiannong Cao, *Fellow, IEEE*

***Abstract*—Vehicle-to-vehicle (V2V) computation offloading has emerged as a promising solution to facilitate computing-intensive vehicular task processing, where task vehicles (i.e., TaVs) will be requested to offload computing-intensive tasks to server vehicles (i.e., SeVs) in order to keep task delay low. However, it is challenging for TaVs to obtain the optimal V2V computation offloading decisions (i.e., realizing the minimal task delay) due to the constraints, including: 1) incomplete offloading information; 2) degraded Quality-of-Service (QoS) of SeVs; and 3) privacy leakage risks. In this article, we develop a learning-based V2V computation offloading algorithm enhanced by SeV's *ability* & *trustfulness awareness* to solve these problems. We emphasize that the proposed algorithm learns the offloading performance of candidate SeVs based on history offloading selections, without requiring the complete offloading information in advance. Additionally, both the QoS of SeVs and safe V2V computation offloading are enhanced in the proposed learning-based algorithm. Furthermore, we conduct extensive simulation experiments to validate the proposed algorithm. The results demonstrate that the proposed algorithm reduces the average task delay by 35% and 40%, and at the same time decreases the learning regret by 39% and 41%, compared to the algorithms without SeV's ability and trustfulness awareness.**

***Index Terms*—Ability and trustfulness awareness, learning-based approach, vehicle-to-vehicle (V2V) computation offloading.**

Manuscript received 4 August 2022; revised 28 October 2022; accepted 9 December 2022. Date of publication 13 December 2022; date of current version 7 April 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62272152, Grant 62271452, and Grant U20A20181; in part by the Key Research and Development Project of Hunan Province of China under Grant 2022GK2020; in part by the Hunan Natural Science Foundation of China under Grant 2022JJ30171; in part by the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) under Grant GML-KF-22-22 and Grant GML-KF-22-23; in part by the Shenzhen Science and Technology Program under Grant CYJ20220530160408019; in part by the CAAI-Huawei MindSpore Open Fund; in part by the Funding Projects of Zhejiang Lab under Grant 2021LC0AB05 and Grant 2022PI0AC01; and in part by the Humanities and Social Sciences Foundation of the Ministry of Education under Grant 21YJCZH183. *(Corresponding authors: Zhu Xiao; Hongbo Jiang.)*

Xingxia Dai, Zhu Xiao, and Hongbo Jiang are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, Hunan, China, and also with the Shenzhen Research Institute, Hunan University, Shenzhen 518055, China (e-mail: xingxdai718@gmail.com; zhxiao@hnu.edu.cn; hongbojiang2004@gmail.com).

Hongyang Chen is with the Research Center for Graph Computing, Zhejiang Lab, Hangzhou 311121, China (e-mail: dr.h.chen@ieee.org).

Geyong Min is with the Department of Computer Science, College of Engineering, Mathematics, and Physical Sciences, University of Exeter, EX4 4QF Exeter, U.K. (e-mail: g.min@exeter.ac.uk).

Schahram Dustdar is with the Research Division of Distributed Systems, TU Wien, 1040 Vienna, Austria (e-mail: dustdar@infosys.tuwien.ac.at).

Jiannong Cao is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: csjcao@comp.polyu.edu.hk).

Digital Object Identifier 10.1109/JIOT.2022.3228811

I. Introduction

With the development of 6G and Internet of Vehicles (IoV) technologies, vehicles have been more *connected* and *intelligent*. These features facilitate the emergence of various vehicular tasks for safe and convenient driving. Several vehicular tasks, such as online path planning and traffic abnormity detection, require massive computing resources and completion within strict deadlines. Vehicular computers performing computing-intensive tasks could, due to the constrained individual computing capabilities, suffer from prolonged task delay [1], [2], [3].

A promising solution to resolve this problem is vehicular computation offloading [4], [5], [6], [7], [8], where computing-intensive vehicular tasks can be offloaded to edge nodes. Currently, computation offloading in vehicular networks involves vehicle-to-infrastructure (V2I) computation offloading and vehicle-to-vehicle (V2V) computation offloading [8], [9], [10]. In V2I computation offloading, fixed entities serve as the vehicular edge nodes, such as base stations (BSs) and road side units (RSUs). Clearly, V2I computation offloading relies on massive fixed edge nodes and, thus, inevitably incurs high deployment and operation expenditure [11]. Moreover, it is impractical for V2I computation offloading to handle a mass of computation tasks due to limited computing resources [12]. For example, a conventional fixed edge node only serves a few tens of vehicles per unit area at a price of high cost in urban intelligent transportation systems [13].

Compared with V2I computation offloading, V2V offloading enables to overcome the above-mentioned issues. V2V computation offloading stems from the fact that substantial vehicular computation resources are chronically underutilized, which is caused by the mismatches between vehicular tasks and computing resources. In V2V computation offloading, vehicles with surplus computing resources provide the offloading services for computing-intensive vehicular tasks [14], without requiring additional deployments of fixed entities. Thus, V2V computation offloading offers a more flexible offloading paradigm, alleviating computation workload of the fixed entities and facilitating resource utilization among vehicles. Additionally, advanced communication technologies

support reliable wireless connectivity among vehicles, such as dedicated short-range communication (DSRC), long-term-evolution vehicle (LTE-V), and IEEE 802.11p. In particular, the communication link duration of V2V computation offloading is longer compared with that of V2I computation offloading when the two vehicles have the same driving heading [15]. In V2V computation offloading, we call vehicles with surplus on-board computation resources Server Vehicles (SeVs), and vehicles requested by computing-hungry tasks task vehicles (TaVs). SeVs deliver offloading services for TaVs to keep vehicular task delay low.

Several previous efforts have studied V2V computation offloading [5], [12], [16]. In these works, computation offloading relies on complete offloading information, including communication rate and available computing resources. On this basis, marvelous algorithms are proposed to either minimize task delay or maximize energy efficiency. Such assumptions, however, are difficult if not impossible to be satisfied in real-world V2V computation offloading. The reason is that there is no prior offloading information about channel states and available computing resources. Specifically, vehicular mobility complicates the prediction of network topology; worse still, available computing resources dynamically fluctuate due to heterogeneous computing capacity and diverse vehicular tasks.

Although existing learning-based algorithms (e.g., [17], [18], [19]) enable to tackle incomplete offloading information, we emphasize that the algorithms generally cannot be applied to V2V computation offloading directly. The reason is that they often ignore the impact of degraded SeV's Quality-of-Service (QoS) and task privacy leakage. V2V computation offloading inevitably decreases the remaining computing resources for processing tasks initiated from SeVs, even may degrade QoS of SeVs. Additionally, computation offloading increases the privacy leakage risks of these tasks. An attacker can eavesdrop on the privacy of offloaded tasks by attacking the target SeVs based on the history offloading inference. For example, when an entertainment-related task is offloaded from a TaV to the SeV, identity privacy may be leaked. Evidently, without the considerations of SeV's QoS and task privacy, learning-based V2V computation offloading is easily trapped by suboptimal offloading solutions.

To address these challenges, this article proposes a learning-based V2V computation offloading approach enhanced by the SeV's *ability & trustfulness awareness*. Specifically, TaVs are jointly aware of SeV's ability and trustfulness, thereby ensuring the QoS of SeVs and decreasing privacy leakage risks of offloaded tasks. Considering the ability awareness and trustfulness awareness, TaVs select the optimal ones from the candidate SeVs to achieve delay-minimal computation offloading without requiring the complete offloading information in advance. To the best of our knowledge, this learning-based computation offloading algorithm has not been studied before in V2V computation offloading. The contributions of this article are highlighted as follows.

1) We study a V2V computation offloading problem (details in Section III), where computing-intensive vehicular tasks are offloaded to vehicles with surplus computing resources. In the problem, we jointly consider incomplete offloading information, SeV's ability and trustfulness issues, aiming to the minimal task delay by optimizing V2V computation offloading decisions.
2) We propose a learning-based V2V computation offloading algorithm. This algorithm enables TaVs to learn the offloading performance of candidate SeVs based on the history offloading selections. The unique feature of this new algorithm is that it does not require the complete offloading information in advance, thereby facilitating its implementation and deployment in the real world.
3) We investigate two kinds of awareness to enhance the proposed learning-based algorithm, i.e., SeV's ability awareness and SeV's trustfulness awareness (detailed in Section IV). Specifically, SeV's ability awareness is beneficial for ensuring SeV's QoS via concerning the SeV offloading abilities; besides, SeV's trustfulness awareness facilitates safe computation offloading by assessing the SeV's trustfulness based on privacy entropy.
4) We analyze the performance loss caused by observation variances in the proposed learning-based algorithm (detailed in Section V-C). Additionally, we conduct extensive simulations using real-world vehicle trajectory data sets. The results validate the effectiveness of our proposed algorithm in terms of various parameters, such as task delay and learning regret (detailed in Section VI).

The remainder of this article is organized as follows. Section II provides an overview of related works. Section III presents the system models and problem formulation followed by problem analysis in Section IV. Section V presents the learning-based approach for V2V computation offloading. In Section VI, we present the evaluations, followed by the conclusion in Section VII.

II. Related Works

Computation offloading has been studied in many existing works, such as [20], [21], and [22], where computing-intensive tasks can be offloaded from end devices to edge nodes. In this section, we review *vehicular computation offloading* and *learning-based computation offloading*.

A. Vehicular Computation Offloading

In vehicular networks, existing works on computation offloading can be divided into two categories: 1) *V2I computation offloading* and 2) *V2V computation offloading* [12]. In the former offloading, vehicles commonly offload computing-intensive tasks to static edge nodes, such as RSUs and BSs [16], [23]. In [16], vehicular tasks are offloaded to RSUs or executed locally. By optimizing task partition, computation offloading decisions, and system configuration, the authors achieve the minimal task delay while maintaining the maximum application accuracy. In [23], a vehicle offloads its tasks to the BS when the vehicle is within the V2I communication coverage. Based on this, the authors formulate computation offloading models to achieve the maximum completion ratio of time-critical vehicular tasks. Tang et al. [24] proposed a dynamic offloading model, where multiple moving

vehicles divide their tasks into sequential subtasks and can offload the subtasks to RSUs to achieve the minimal total task delay and waiting time. In [25], vehicles offload their tasks to the RSU for execution under the long-term energy consumption constraint, aiming at the minimal average response time. Tang and Wu [26] designed a general caching-enabled VEC scheme, where task caching and offloading are jointly considered in VEC networks. Guided by this, the authors strive for effective caching and offloading strategies to achieve the minimal weighted sum of the service time and energy consumption. In [27], computation-intensive vehicular applications are offloaded to RSUs to seek powerful edge processing capabilities. Furthermore, each application can be divided into multiple dependent tasks to be executed by different edge servers, so as further to minimize the average completion time of various applications. While V2I computation offloading provides relatively stable computation offloading services for user devices, such offloading relies on massive fixed edge nodes and inevitably incurs high deployment and operation expenditure [11]. What is worse, limited computing resources render it impractical to handle a mass of offloading tasks and, thus, incompetent to support computation demands from urban intelligent transportation systems.

Different from V2I computation offloading, V2V computation offloading integrates underutilized vehicular computing resources without requiring additional network deployments. In this way, V2V computation offloading offers a more flexible vehicular computation offloading paradigm and achieves higher resource utilization efficiency [2], [5], [12]. Chen and Xu [2] leveraged the task replication technique to improve computation offloading performance. A vehicular task can be offloaded to multiple candidate vehicles with surplus computing resources. Wang et al. [5] considered a dynamic vehicular network, where computing-intensive vehicular tasks can be offloaded to neighboring vehicular clusters for minimal system energy consumption. Liu et al. [12] proposed a vehicle-mounted edge mechanism to remedy the coverage limitation of static edge nodes. By jointly considering path planning and resource allocation, the maximum completed tasks of V-edge can be achieved.

B. Learning-Based Computation Offloading

Learning-based algorithms have been studied in computation offloading [17], [18], [19], [28], [29], [30], [31], [32], [33]. Ouyang et al. [17] proposed a Thompson-sampling-enhanced online learning algorithm to cope with unknown future information and system dynamics, aiming at optimizing perceived latency and service migration cost. Sun et al. [18] proposed an adaptive learning-based computation offloading algorithm, enhanced by input awareness and occurrence awareness, to minimize the average task delay. Zhou et al. [19] proposed an online learning algorithm to achieve a well tradeoff between delay and energy consumption. The algorithm adopts adversarial multiarmed bandit theory to realize ultrareliable and low-latency communication. Luo et al. [28] derived a self-learning distributed computation offloading scheme based on a game theorem, where vehicles make computation offloading decisions to minimize the system cost. Shang et al. [29] leveraged deep learning techniques to optimize user association, data partition, transmit power and computing resources, targeting the minimal energy consumption of end-users. Lin et al. [30] investigated a contextual clustering of bandits approach to address online computation offloading in heterogeneous vehicular networks with unknown environment dynamics. By learning the relationship between historical observations and rewards, this approach minimizes the expectation of total offloading energy consumption under task delay constraints. Yang et al. [31] considered a computation offloading problem in dynamic fog networks, where end users offload tasks to fog nodes under the unknown statistics of arrival tasks. A learning-based approach is proposed to achieve the minimal network latency. Tekin and Liu [32] investigated both the rested bandits' and restless bandits' online learning problems in the manner of a centralized and a decentralized setting. Liu et al. [33] studied a restless multiarmed bandit problem, where the reward state of each arm follows an unknown Markovian rule.

While the aforementioned learning-based algorithms enable computation offloading without requiring the complete offloading information, the following issues, which play a vital role in effective and safe V2V computation offloading, are insufficiently studied: 1) in V2V computation offloading, SeVs are required to process the tasks offloaded from TaVs. The offloading inevitably reduces the computation resources of SeVs. In particular, when SeVs generate large own computational workload, their QoS may be degraded and, hence, decrease computation offloading efficiency and 2) the trustfulness issue is mostly ignored in the existing learning-based algorithms. Without this consideration, task privacy leakage risks increase. To address the above issues, we propose a learning-based algorithm enhanced by SeV's ability awareness and trustfulness awareness in this article.

III. System Models and Problem Formulation

In this section, we first elaborate on system models, including system overview, task model, offloading model, and delay model. After that, we formulate the V2V computation offloading problem.

A. System Overview

The timeline of V2V computation offloading in our system is discretized into time slots $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$. At time slot t, vehicle v is characterized by a tuple $(l_v^t, v_v^t, h_v^t, F^v)$, where l_v^t indicates the location (in longitude and latitude), v_v^t represents the velocity (in meters per second), h_v^t denotes the driving heading (in directions) and F^v is the maximum computing resources (in CPU cycles). Vehicular roles (i.e., TaV and SeV), due to diverse requested tasks, may change across epochs $\mathcal{B}_b = [t_b, t_{b'}]$, where $b = 1, \ldots, B$, $t_1 = 1$ and $t_{B'} = T$. TaVs select SeVs to perform V2V computation offloading per time slot in order to keep task delay low [34]. As shown in Fig. 1, three candidate SeVs (SeVs 1–3) could provide V2V computation offloading service for TaV 1, and currently SeV 3 is selected. It is noted that V2V computation

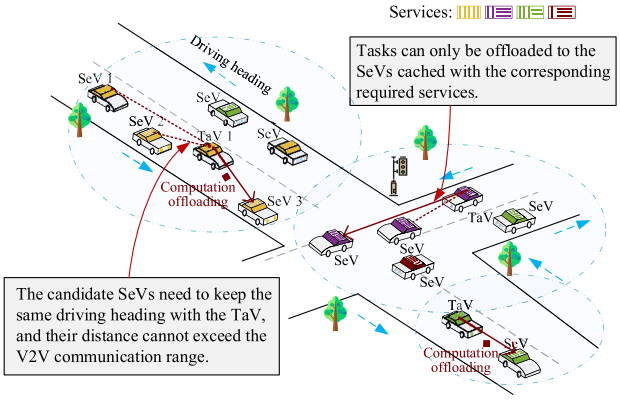


Fig. 1. Illustration of V2V computation offloading. TaVs offload tasks to SeVs for small task computation delay. The candidate SeVs are required to keep the same driving heading as the TaV, be within the V2V communication coverage, and be configured with the requested computation services.

TABLE I
SUMMARY OF THE MAIN SYMBOLS

Notations	Definitions
$\mathcal{T}, \mathcal{B}_b$	the time slot set and epoch set
l_v^t, v_v^t,	the location and velocity of vehicle v
h_v^t, F^v,	the driving heading and maximum computing resources
$\mathcal{U}^t, \mathcal{N}^t \mathcal{S}^t$	the TaV set, task set and SeV set
b_n^t, c_n^t	the task input-data size and computational workload.
τ_n^t, o_n^t	the task stipulated deadline and output/input ratio.
$\alpha_{n,s}$	the service caching of SeV s to task n.
$x_{n,s}^t$	the V2V computation offloading decision
$r_{u,s}^{(up),t}$	the task upload rate between TaV u and SeV s
$T_{n,s}^{(up),t}$	the upload delay of task n
$\mathcal{P}_{n,s}^t$	the allocation policy of computation resource
$f_{n,s}^t$	the computation resources allocated to task n
f_s	the total computation resources of SeV s
$\mathcal{N}_s^t$	the tasks offloaded to SeV s at time slot t
$T_{n,s}^{(pr),t}$	the processing delay of task n at time slot t
$T_{n,s}^{(fb),t}$	the result feedback delay of task n
A_s^t	the offloading ability of SeV s at time slot t
H_s^t	the trustfulness of SeV s at time slot t

offloading is easily integrated into V2I computation offloading. Generally, a TaV can seek V2I computation offloading when there is no candidate SeV. Yet, this scenario is beyond the scope of this article. We list the key notations as Table I for better readability.

B. Task Model

At each time slot $t \in \mathcal{T}$, TaV $u \in \mathcal{U}^t$ generates a computing-intensive task. The vehicular tasks remain unchanged within a time slot while may vary dynamically across different time slots. We assume that the tasks have sequential dependency [35]. Task $n \in \mathcal{N}^t$ is performed as a tuple of $(b_n^t, c_n^t, \tau_n^t, o_n^t)$ at time slot t, where b_n^t indicates the task input-data size (in bits), c_n^t represents computational workload (in required CPU cycles), τ_n^t denotes the task stipulated deadline (in second) and o_n^t is the output/input ratio (in percent) [12]. For notational convenience, task n is assumed to be generated by TaV u in this work.

To process a task, dedicated computation services are required. However, SeV $s \in \mathcal{S}^t$ has constrained storage space and only caches limited computation services. We introduce a binary variable $\alpha_{n,s} \in \{0, 1\}$ to illustrate whether a computation service requested by task n is cached ($\alpha_{n,s} = 1$) or not ($\alpha_{n,s} = 0$) in SeV s. Guided by the computation service caching, TaVs offload computing-intensive tasks to the SeVs for task processing.

C. Offloading Model

V2V computation offloading is divided into three phases, i.e., vehicular role determination, candidate SeV recognition, and computation offloading decisions.

Vehicular Role Determination: In V2V computation offloading, vehicular roles (i.e., SeVs and TaVs) change across epochs. For a specific vehicle, it may serve as a SeV when requested by a lightweight computational task; or become a TaV when requested by a computing-intensive video streaming task [34]. To determine the roles, each vehicle calculates local task delay c_v^t/F^v, where c_v^t is the computational workload of vehicle v at time slot t, and F^v denotes the maximum computing resources of vehicle v. A vehicle performs as a SeV when the local delay is less than the task deadline; otherwise, the vehicle acts as a TaV.

Supported by DSRC standards, the periodic beaconing messages disseminate the physical information among vehicles, such as the information of vehicular role, location, velocity, and driving heading [18].

Candidate SeV Recognition: For a TaV u, SeV s is recognized as a candidate SeV when ensuring: 1) SeV s is in the same driving heading with TaV u, i.e., $h_s^t = h_u^t$; 2) the distance between SeV s and TaV u is within the V2V communication coverage of R at the beginning of time slot t, i. e., $|l_s^t - l_u^t| \leq R$; and 3) SeV s caches the computation service requested by task n, i.e., $\alpha_{n,s} = 1$. The former two constraints are used to maintain reliable communication links between TaVs and SeVs [18]; the latter one ensures the support of computation service. Let $\mathcal{S}_n^t \neq \emptyset$ denote the candidate SeV set for TaV u at time slot t. Note that the candidate SeV set may change across epochs due to vehicular mobility and diverse requested tasks.

Computation Offloading Decisions: After determining candidate SeVs, each TaV offloads computing-intensive tasks to the selected SeV in order to keep task delay low. Let $x_{n,s}^t \in \{0, 1\}$ denote the V2V computation offloading decision. When $x_{n,s}^t = 1$, task n is offloaded to SeV s at time slot t; otherwise, SeV s is not selected. Without loss of generality, we assume that each task can be offloaded to a single SeV per time slot to maintain task continuity [36].

D. Delay Model

Task delay involves upload delay, processing delay, and result feedback delay. Several factors affect task delay, including communication links, data size, computational workload, and allocated computing resources.

Upload Delay: Based on the computation offloading decision, a TaV uploads its task of b_n^t data bits to the target SeV. In V2V computation offloading, we assume that the network states are identical per time slot, while the states vary across time slots [37]. We define $g_{u,s}^t$ and $I_{u,s}^t$ as the channel gain

and the interference between TaV u and SeV s at time slot t, respectively. Given fixed wireless bandwidth W, noise power δ^2 and transmission power p, we obtain the task upload rate between TaV u and SeV s at time slot t

$$r_{u,s}^{(up),t} = W \log_2\left(1 + \frac{p g_{u,s}^t}{\delta^2 + I_{u,s}^t}\right), s \in \mathcal{S}_n^t, t \in \mathcal{T}. \tag{1}$$

Based on the upload rate, we obtain the upload delay

$$T_{n,s}^{(up),t} = \frac{b_n^t}{r_{u,s}^{(up),t}}, s \in \mathcal{S}_n^t, t \in \mathcal{T}. \tag{2}$$

We emphasize that the upload rates between SeVs and TaVs cannot be obtained in advance, since network states are hard to model or predict in V2V computation offloading.

Processing Delay: After a TaV uploads its task to SeV s, the SeV will allocate computing resources for processing the task. Since a SeV may serve for multiple TaVs, we consider that the computation resource allocation of SeV s to task n at time slot t follows the allocation policy of $P_{n,s}^t \in \mathcal{P}_s^t$. Without loss generality, the policy is assumed to be a set of discrete coefficients of computation resource provisioning [38]. Guided by this, the computation resources allocated from edge server s to task n at time slot t are expressed as follows:

$$f_{n,s}^t = f_s \frac{P_{n,s}^t}{\sum_{i \in \mathcal{N}_s^t} P_{i,s}^t} \ \forall x_{n,s}^t = 1, s \in \mathcal{S}_n^t, t \in \mathcal{T} \tag{3}$$

where f_s is the total computation resources of SeV s and $\mathcal{N}_s^t$ is the tasks offloaded to SeV s at time slot t. Observe that when $P_{n,s}^t = 1 \ \forall x_{n,s}^t = 1$, edge server m evenly allocates computation resources for the offloaded vehicular tasks. In practice, several process schedulers enable approximation of this computation configuration process, such as distributed weighted round-robin (DWRR) [39]. It is assumed that the allocated computing resources remain fixed per time slot while changes across time slots [18].

Hence, the processing delay of task n at time slot t can be written as

$$T_{n,s}^{(pr),t} = \frac{c_n^t}{f_{n,s}^t}, s \in \mathcal{S}_n^t, t \in \mathcal{T}. \tag{4}$$

It is noted that the allocated computing resources are unavailable before computation offloading due to heterogeneous computing capacity and diverse vehicular tasks.

Result Feedback Delay: After processing task n, SeV s needs to feedback the computation result of $b_n^t o_n^t$ bits to TaV u. When the distance between SeV s and TaV u is within the V2V communication coverage, SeV s can directly transmit the computation results to TaV u. If the distance exceeds the V2V communication coverage due to vehicular mobility, the computation results need to be transmitted via edge relaying and, hence, incur the relay delay ω. We denote $T_{n,s}^{(fb),t}$ as the result feedback delay, expressed as

$$T_{n,s}^{(fb),t} = \begin{cases} \dfrac{b_n^t o_n^t}{r_{s,u}^{(dl),t}}, & |l_s^t - l_u^t| \le R \quad (5) \\ \dfrac{b_n^t o_n^t}{r_{s,u}^{(dl),t}} + \omega, & |l_s^t - l_u^t| > R \quad (6) \end{cases}$$

where $r_{s,u}^{(dl),t}$ is the result feedback rate between SeV s and TaV u at time slot t.

E. Problem Formulation

In V2V computation offloading, task delay affects QoS directly, especially for delay-sensitive vehicular tasks [18]. Conditioned on $\mathcal{S}_n^t \neq \emptyset$ (i.e., there is at least one SeV for task n at time slot t), task delay involves the upload delay, processing delay and result feedback delay. When task n is offloaded to SeV s at time slot t, task delay is expressed as

$$T_{n,s}^t = T_{n,s}^{(up),t} + T_{n,s}^{(pr),t} + T_{n,s}^{(fb),t}, s \in \mathcal{S}_n^t, t \in \mathcal{T}. \tag{7}$$

To reduce the impact of randomness per time slot, our objective is to minimize the average overall task delay within T time slots [40]. Besides, a series of constraints need to be guaranteed to implement V2V computation offloading. Mathematically, we formulate the V2V computation offloading problem as follows:

$$\min_{x_{n,s}^t, \mathcal{P}_s^t} \ \frac{1}{T} \sum_{t \in \mathcal{T}} \sum_{n \in \mathcal{N}^t} \sum_{s \in \mathcal{S}_n^t} \mathbb{E}\{T_{n,s}^t\} \tag{8}$$

$$s.t. \begin{cases} f_{n,s}^t \le F^s & (9) \\ \alpha_{n,s} \in \{0, 1\} & (10) \\ x_{n,s}^t \in \{0, 1\} & (11) \\ \sum_{s \in \mathcal{S}_n^t} x_{n,s}^t = 1. & (12) \end{cases}$$

Constraint (9) is the *computing resource constraint*, i.e., the allocated computing resources cannot exceed the maximum SeV computing resources per time slot. Constraint (10) denotes the *service constraint*, i.e., a computation service will be cached or not in a SeV. Constraints (11) and (12) are *offloading decision constraints*, i.e., each TaV selects a single SeV s for task processing to maintain task continuity.

To address the above problem as defined in (8) under the constraints of (9) through (12), the complete offloading information is required in advance, including the information on channel states and available computing resources. Unfortunately, vehicular mobility complicates the prediction of network topology; available computing resources dynamically fluctuate caused by heterogeneous computing capacity and diverse vehicular tasks. As such, TaVs need to make V2V computation offloading decisions based on the incomplete offloading information. This issue incapacitates traditional optimization solutions, e.g., dynamic programming algorithms. What is worse, V2V computation offloading inevitably decreases the remaining SeV computing resources, and may degrade QoS of SeVs. Additionally, computation offloading increases the privacy leakage risks of these tasks. While existing learning-based algorithms enable to tackle incomplete offloading information, the algorithms often ignore the impact of degraded SeV's QoS and task privacy leakage. Without these considerations, V2V computation offloading is easily trapped by suboptimal offloading solutions. Thus, the existing algorithms cannot be applied to the V2V computation offloading problem directly.

IV. Problem Analysis

To resolve the V2V computation offloading problem, the above-mentioned challenges (i.e., incomplete offloading information, degraded QoS of SeVs, and task privacy leakage risks) are required to be addressed. To this end, in this section, we first analyze the offloading information, and then we design SeV's ability and trustfulness awareness to facilitate effective and safe V2V computation offloading. After that, we summarize the problem analysis.

A. Information Analysis

We analyze the offloading information based on whether TaVs are aware of the information in advance.

On the one hand, TaVs generate computing-intensive tasks per time slot, and the task information (i.e., b_n^t, c_n^t, τ_n^t, and o_n^t) can be acquired by TaVs before computation offloading. Besides, physical information of SeV s at time slot t (i.e., l_s^t, v_s^t, and h_s^t) can be obtained by TaVs via DSRC standards and LTE-V. On the other hand, some offloading information is unavailable for TaVs, such as the communication rate between TaVs and SeVs (i.e., $r_{s,u}^t$) and the allocated computing resources from SeVs (i.e., $f_{n,s}^t$).

Remarks: It is noteworthy that V2V computation offloading with *complete* offloading information is often not the case in the real world since the following issues remain: 1) the transmission rate between TaVs and SeVs is complicated to model or predict due to vehicular mobility and varying network topology and 2) TaVs generate diverse tasks and SeVs have heterogeneous computing capacities. The facts hinder TaVs from obtaining information on allocated computing resources.

The challenge motivates us to seek V2V computation offloading without requiring complete offloading information in advance. To that end, we propose a learning-based V2V computation offloading algorithm, detailed in Section V.

B. Ability Awareness

In V2V computation offloading, all vehicles have an inherent incentive to join the cooperation, since they may want to use resources from other vehicles. If a vehicle is uncooperative, e.g., always using computing resources from other vehicles but never sharing its own resources, the vehicular network does not have to allow the vehicle to receive computation offloading services. Based on the cooperative paradigm, there are many works that study V2V computation offloading, such as [12], [41], and [42].

While SeVs and TaVs behave in a cooperative manner, it is not straightforward to achieve effective V2V computation offloading. The reason is that V2V computation offloading inevitably prolongs task computation delay of SeVs and, hence, degrades the QoS, especially for the SeVs not having excessive spare resources. More explicitly, for a specific SeV, its tasks are processed locally and the delay is determined by its task computational workload and computing resources. When implementing V2V computation offloading, the SeV will receive the offloading requests from TaVs. Then, the SeV is required to allocate its computing resources to the TaVs for task processing. In V2V computation offloading, we recognize that the offloaded tasks hold the same priority as the local tasks of SeVs. The intuition is that V2V computation offloading concentrates on the whole system performance rather than individual interests. When requested by TaVs, the SeVs with moderate computing resources may have to sacrifice their own QoS to guarantee the system's performance. For example, the computation delay of a SeV will double after contributing half of its computing resources to TaVs.

More importantly, offloading abilities of SeVs are heterogeneous due to different individual computing resources and heterogeneous computational workload. To achieve effective V2V computation offloading meanwhile guaranteeing QoS of SeVs, it is crucial to measure the abilities of SeVs in vehicular networks. To that end, we define a SeV's ability-awareness function to assess the offloading ability of SeV s at time slot t, expressed as

$$A_s^t = \gamma F^s \sqrt{\sigma_s^t}, s \in \mathcal{S}_n^t, t \in \mathcal{T} \tag{13}$$

where γ is a normalized coefficient to make the ability range from 0 to 1, and F^s is the maximum computing resources of SeV s. We define $\sigma_s^t = c_s^t(c - c_s^t)$, where c_s^t represents the task computational workload of SeV s at time slot t, $c \in (c_s^t,\ 2c_s^t)$. When SeV s has heavy computational workload, σ_s^t will drop sharply and, hence, go against large ability. Based on (13), the ability of SeV s at time slot t is jointly determined by the maximum computing resources (i.e., F^s) and the computational workload (i.e. c_s^t). Suppose that: 1) SeV s has large computing resources and 2) small computational workload, then the SeV will possess a great ability for processing tasks offloaded by TaVs.

Remarks: The ability-awareness scheme is beneficial for V2V computation offloading via resolving the *degraded QoS of SeVs.* Specifically, SeV's ability awareness jointly considers the maximum computing resources and computational workload of SeVs. When a SeV has moderate computing resources and large own computational workload, the ability-awareness scheme ensures that the SeV has less offloading ability. As such, when implementing V2V computation offloading, TaVs should incline to select SeVs with large abilities to maintain the QoS of SeVs.

C. Trustfulness Awareness

Despite offloading computing-intensive tasks to SeVs facilitates less computation delay, privacy leakage risks in V2V computation offloading cause tremendous difficulties for fully reaping the benefits of task offloading [43]. Specifically, acting as both resource requester and provider, vehicles are often owned by individuals and generate massive private data; besides, in the open V2V edge computing network, there are inevitably malicious attackers which destroy the reliability and stability of V2V computation offloading [44].

In V2V computation offloading, an attacker tries to eavesdrop on the privacy information of the offloaded tasks when tasks are offloaded to SeVs. Then, the attacker will infer the selected probability of candidate SeVs based on history offloading selections and, hence, attack the SeV with the largest probabilities [45]. If the attacked SeV is exactly

selected to deliver computation offloading services, the privacy leakage turns out. Consequently, the spatiotemporal interactions of drivers with others may be exposed when mobility data is leakages [46]. What is worse, the privacy leakage may incur incorrect driving strategies and, thus, deteriorate traffic safety for driving-related tasks.

To address the issue of privacy leakage, we assess SeV's trustfulness in providing computation offloading guidance, facilitating safe V2V computation offloading. Since vehicles in the same area often have similar task requests [47], the SeVs equipped with large computing capabilities may be frequently selected to deliver computation offloading services. The frequent selections inherently increase the attacked risks of those SeVs. On this basis, we introduce privacy entropy to measure trustfulness [48]. The trustfulness of SeV s at time slot t is expressed as

$$H_s^t = -p_s^t \log p_s^t - \left(1 - p_s^t\right) \log\left(1 - p_s^t\right), s \in \mathcal{S}_n^t, t \in \mathcal{T} \quad (14)$$

where p_s^t denotes the probability that SeV s is selected to provide computation offloading services at time slot t. A large trustfulness of H_s^t represents low privacy leakage risks of SeV s. Clearly, when $p_s^t = 1/2$, SeV s holds the largest selected uncertainty and has a large trustfulness to process the offloaded tasks. In this case, the SeV is hard to be inferred by an attacker and, thus, decreases privacy leakage risks of offloaded tasks.

Remarks: Trustfulness awareness is beneficial for safe V2V computation offloading. Specifically, privacy entropy quantifies the trustfulness of SeVs. A SeV with larger trustfulness offers a higher probability of safe computation offloading, which provides guidance for SeV selections. As such, in V2V computation offloading, TaVs are prone to selecting SeVs with high trustfulness to reduce privacy leakage risks.

D. Summary

The above-mentioned three challenges impede effective and safe V2V computation offloading, summarized as: 1) incomplete offloading information; 2) degraded QoS of SeVs; 3) privacy leakage risks.

Incomplete offloading information calls for a learning-based algorithm. The algorithm enables TaVs to make computation offloading decisions based on the history offloading performance of SeVs, without requiring the complete offloading information in advance. Furthermore, considering the degraded QoS of SeVs and privacy leakage risks of offloaded tasks, the proposed learning-based algorithm needs to be aware of SeV's ability and trustfulness to facilitate effective and safe V2V computation offloading.

V. Learning-Based V2V Computation Offloading

In this section, we propose a learning-based V2V computation offloading algorithm. The algorithm enables SeV *ability* & *trustfulness awareness*, named LTO-ATA. Besides, we conduct theoretical analysis on the upper regret bound for the LTO-ATA algorithm.

A. Learning-Based Solution to Tackle Incomplete Offloading Information

In V2V computation offloading, TaVs have to make offloading decisions based on incomplete offloading information. Specifically, vehicular mobility complicates the prediction of network topology; fluctuant computing capabilities hinder TaVs from obtaining the allocated computing resources from SeVs. These facts lead to lacking transmission rate information and computing resource information. Incomplete offloading information motivates us to seek learning-based V2V computation offloading. In this way, TaVs observe and learn offloading performance of candidate SeVs based on history offloading selections, so as to offload its task to the SeV with minimal task delay, without requiring the complete offloading information in advance.

MAB serves as a promising learning-based solution for V2V computation offloading [49]. In an MAB problem, a TaV acts as a "gambler" and a candidate SeV serves as an "arm." The gambler selects an arm without requiring complete offloading information in advance and correspondingly acquires a reward (i.e., task delay). To achieve maximum rewards (i.e., the minimal task delay), there are two selections for the TaV: 1) exploring a new SeV for a possible less task delay and 2) exploiting the SeV with minimal task delay up to now to avoid unnecessary exploration. Clearly, the selections inherently involve the *exploration–exploitation dilemma.* The classical MAB algorithms enable well-balanced exploration and exploitation via continuous learning, such as the upper confidence bounds (UCBs) algorithm [49].

However, V2V computation offloading incapacitates the existing algorithms. The reasons are as follows.

1) Vehicular roles change across epochs. TaVs need to implement the learning-based algorithm for different "arms" at each epoch.
2) V2V computation offloading may degrade the QoS of SeVs. As such, SeV's ability should be considered in the learning-based algorithm.
3) The learning-based algorithm is required to measure SeV's trustfulness to avoid privacy leakage risks in V2V computation offloading.

Without these considerations, existing learning-based algorithms cannot be applied for V2V computation offloading directly.

B. LTO-ATA Algorithm

Motivated by the limitations of existing algorithms, we propose the LTO-ATA algorithm. We emphasize that the LTO-ATA algorithm enables SeV's ability awareness and SeV's trustfulness awareness, without requiring the complete offloading information in advance. Specifically, a TaV learns the offloading performance of candidate SeVs based on history offloading selections. Since the SeV states (i.e., transmission rate and available computing resource) are changing across time slots, single learning cannot reflect the offloading performance of the SeV. As such, continuous learning is required in the LTO-ATA algorithm to smooth observation

Algorithm 1: LTO-ATA Algorithm

Input : Physical information l_v^t, v_v^t, h_v^t, F^v and task information b_n^t, c_n^t, τ_n^t, o_n^t.

Output: Vehicular computation offloading decision $x_{n,s}^t$, computation resource allocation policy $\mathcal{P}_s^t$.

1 **Phase 1: Initialization**
2 Default $\mathcal{S}_n^0 = \emptyset$, $K_{n,s}^0 = 0$ $\bar{T}_{n,s}^0 = 0$.
3 **for** $t = 1$ ***to*** $t = T$ **do**
4 **Phase 2: Vehicular role determination**
5 **if** $c_v^t / F^v < \tau_v^t$ **then**
6 Vehicle v acts as a SeV.
7 **end**
8 **else**
9 Vehicle v acts as a TaV.
10 **end**
11 **Phase 3: Candidate SeV recognition**
12 **if** $h_s^t = h_u^t$, $|l_s^t - l_u^t| \leq R$, *and* $\alpha_{n,s} = 1$ **then**
13 Sev s serves as a candidate, $\mathcal{S}_n^t \leftarrow s$.
14 **end**
15 **Phase 4: Learning while offloading**
16 Update TaV set $\mathcal{U}^t$ and SeV set $\mathcal{S}^t$.
17 **if** $\exists$ *SeV* s, $K_{n,s}^{t-1} = 0$ **then**
18 SeV s is selected at the t-th time slot, $\xi_n^t = s$ and $K_{n,\xi_n^t}^t = 1$.
19 **end**
20 **else**
21 Calculate the index value for each candidate SeV $s \in \mathcal{S}_n^t$ based on (15).
22 Offload task n to ξ_n^t, such that (16) is satisfied.
23 **end**
24 Calculate the vehicular task delay $T_{n,\xi_n^t}^t$.
25 Update $K_{n,s}^t$ based on (18).
26 Update $\bar{T}_{n,s}^t$ based on (19).
27 **Phase 5: Computation resource allocation**
28 Obtain the task set of $\mathcal{N}_s^t$ of SeV s at time slot t.
29 Determine $\mathcal{P}_{n,s}^t$ for each task $n \in \mathcal{N}_s^t$.
30 **end**

variance [50]. Besides, to ensure effective and safe V2V computation offloading, TaVs need to be aware of SeV's ability and trustfulness in the algorithm. On this basis, TaVs decide to select the current optimal SeV, or explore other candidate SeVs for possible less task delay. After several learning iterations, the TaV finds out the optimal SeV to implement V2V computation offloading. The computation offloading achieves low task delay meanwhile considering QoS of SeV and task privacy risks. Clearly, the proposed LTO-ATA algorithm is extremely different from the existing learning-based algorithms.

The LTO-ATA algorithm is presented in Algorithm 1, where the number of learning times is equal to the number of time slots. The reason is that each TaV learns the offloading performance of SeVs and updates the V2V computation offloading decisions per time slot. Algorithm 1 consists of five phases, i.e., the initialization phase (lines 1 and 2), the vehicular role determination phase (lines 3–10), the candidate SeV recognition phase (lines 11–14), the learning while offloading phase (lines 15–26), and the computation resource allocation phase (lines 27–30).

In the initialization phase (lines 1 and 2), the candidate SeVs for task n are defaulted as an empty set at the initial time slot. Let $K_{n,s}^t$ and $\bar{T}_{n,s}^t$ denote the selected times and the average task delay of SeV s for processing task n after t time slots, respectively. Both $K_{n,s}^t$ and $\bar{T}_{n,s}^t$ are set as zero at the initial time slot, i.e., $K_{n,s}^0 = \bar{T}_{n,s}^0 = 0$.

In the vehicular role determination phase (lines 3–10), each vehicle calculates the local task delay. When the local task delay is less than the task deadline, the vehicle is categorized as a SeV; otherwise, the vehicle is categorized as a TaV. In V2V computation offloading, vehicular roles (i.e., SeVs and TaVs) may change across epochs due to diverse requested vehicular tasks.

In the candidate SeV recognition phase (lines 11–14), we obtain the candidate SeV set for each TaV. The candidate SeV needs to satisfy: 1) the physical constraint, i.e., the candidate SeV needs to directly communicate and keeps the same driving heading with the TaV and 2) the service caching constraint, i.e., the SeV is required to configure the computation service requested by the TaV. These two constraints ensure reliable communication links and computation service support. When the constraints are both satisfied, a SeV becomes one of the candidate SeVs for the TaV.

In the learning while offloading phase (lines 15–27), the TaV makes the V2V computation offloading decision based on the history offloading performance of SeVs. To this end, we update the candidate SeV set first. We then define ξ_n^t to represent the selected SeV for processing task n at time slot t.

1) If there exists SeV s not been selected once after $t-1$ time slots, it will be selected at time slot t. This behavior facilitates learning exploration, avoiding local optimum. In this case, $\xi_n^t = s$ and $K_{n,\xi_n^t}^t = 1$.
2) When each candidate SeV has been selected at least once after $t-1$ time slots, we derive the index-based minimal value research to implement V2V computation offloading. We define the index function of SeV s at time slot t as follows:

$$\Psi_s^t = \left(1 - A_s^t\right)\left(1 - H_s^t\right)\left(\bar{T}_{n,s}^{t-1} - \Upsilon_{n,s}^{t-1}\right), t \in \mathcal{T} \quad (15)$$

where $(1 - A_s^t)$ and $(1 - H_s^t)$ reflect the SeV's ability awareness and trustfulness awareness, respectively. For a SeV, if it has higher offloading ability and trustfulness, the SeV contributes to a smaller index value and, thus, has a larger chance to be selected. Besides, $\bar{T}_{n,s}^{t-1}$ denotes the average offloading delay of task n after $(t-1)$ time slots. This demonstrates that the index-based research is based on the SeV history offloading performance (i.e., task delay), without requiring complete offloading information in advance. $\Upsilon_{n,s}^{t-1}$ is the confidence bound, which is used to achieve the balance between exploration and exploitation [49]

$$\Upsilon_{n,s}^{t-1} = \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}, t \in \mathcal{T} \quad (16)$$

where t denotes the current total learning times (i.e., time slots), $t = \sum_{s\in\mathcal{S}^t} K^t_{n,s}$. Besides, $K^{t-1}_{n,s}$ is the selected times of SeV s for processing task n after $t-1$ times learning. Less selected times contribute to a less index value, which facilitates learning exploration. The parameter β is used to adjust the weight of exploration, we will discuss the impact of β on offloading performance in Section VI-E.

On this basis, we find out the target SeV (i.e., SeV ξ^t_n) via the index-based minimal value research

$$\xi^t_n = \arg\min_{s\in\mathcal{S}^t_n} \Psi^t_s, t \in \mathcal{T}. \tag{17}$$

Overall, the index-based minimal value research enables to find out the SeV with minimal task delay without requiring the complete offloading information in advance, meanwhile being aware of SeV's ability and trustfulness.

After that, we update the selected times of SeV ξ^t_n up to the tth time learning, expressed as

$$K^t_{n,\xi^t_n} = K^{t-1}_{n,\xi^t_n} + 1, t \in \mathcal{T}. \tag{18}$$

Furthermore, we obtain the task delay conducted by SeV ξ^t_n based on the (7). The average task delay of SeV ξ^t_n needs to be updated as follows:

$$\bar{T}^t_{n,\xi^t_n} = \frac{\bar{T}^{t-1}_{n,\xi^t_n} K^{t-1}_{n,\xi^t_n} + T^t_{n,\xi^t_n}}{K^t_{n,\xi^t_n}}, t \in \mathcal{T}. \tag{19}$$

In the computation resource allocation phase (lines 27–30), we obtain the task set for each SeV based on the offloading decisions. Then, each SeV determines the computation resource allocation strategies of $\mathcal{P}^t_{n,s}$, and offloaded tasks are processed by the allocated computation resources.

Repeat the learning iterations for phases 2 to phase 5 in the LTO-ATA algorithm until learning times $t > T$.

We emphasize that the proposed LTO-ATA algorithm is easy to implement in real-world V2V computation offloading. The reason is that the intractable offloading information is not required in advance, such as a complicated network topology and varying allocated computing resources. In V2V computation offloading, the proposed LTO-ATA algorithm implements computation offloading based on incomplete offloading information. Furthermore, the proposed algorithm is enhanced by SeV's ability awareness and SeV's trustfulness awareness, facilitating effective and safe V2V computation offloading.

Ability & Trustfulness Awareness: From the index function defined in (15), we find that lower SeV's ability (i.e., A^t_s) and trustfulness (i.e., H^t_s) contribute to a larger index value, thus the SeV has less chance to be selected. In contrast, a SeV with larger A^t_s and H^t_s is beneficial for a small index and, hence, is more likely to be selected. In this way, the proposed LTO-ATA algorithm empowers to dynamically optimize the computation offloading strategies based on the SeV's ability and the SeV's trustfulness and, thus, contributes to effective and safe V2V computation offloading.

Computational Complexity of the LTO-ATA Algorithm: Line 5 calculates the local task delay for overall vehicles in V2V computation offloading. The computational complexity is $\mathcal{O}(S^t + U^t)$, where S^t denotes overall SeVs and U^t is TaVs at time slot t, $S^t = |\mathcal{S}^t|$ and $U^t = |\mathcal{U}^t|$. Line 21 calculates the index value for all candidate SeVs, the computational complexity is $\mathcal{O}(S^t_n)$, where S^t_n denotes the overall candidate SeVs for task n at time slot t, $S^t_n = |\mathcal{S}^t_n|$. Line 22 shows a minimum value-seeking problem, occupying $\mathcal{O}(S^t_n)$ computational complexity. Line 29 determines computation resource allocation strategies. The complexity is $\mathcal{O}(P^t_{n,s})$, where $P^t_{n,s} = |\mathcal{P}^t_{n,s}|$ is the strategy space of computation resource allocation. Additionally, the update behaviors in lines 16, 25, and 26 have the computational complexity of $\mathcal{O}(1)$. Therefore, we conclude that the computational complexity of our proposed algorithm is $\mathcal{O}(S^t + U^t + N^t S^t_n + N^t_s P^t_{n,s})$, where $N^t = |\mathcal{N}^t|$ is the total offloaded tasks, and $N^t_s = |\mathcal{N}^t_s|$ denotes the task set served by SeV s at time slot t.

C. Regret Analysis

In this section, we study the learning regret conducted by the proposed LTO-ATA algorithm. We define the learning regret of task n as: $\Delta_n = \mu_{n,s} - \mu_{n,*}$, where $\mu_{n,s}$ is the expected task delay performed by SeV s, and $\mu_{n,*}$ is the minimal expected task delay conducted by the optimal SeV. The total learning regret after T times learning is defined as

$$R^T = \sum_{t=1}^{T}\sum_{n=1}^{N^t}\sum_{s:\mu^t_{n,s}>\mu^t_{n,*}}^{S^t_n} \Delta_n \mathbb{E}\{K^T_{n,s}\}. \tag{20}$$

Theorem 1: Considering S^t_n candidate SeVs for task n at time slot t, we drive the learning regret of the LOT-ATA algorithm. After T times learning, the total learning regret is upper bounded as

$$R^T \le \sum_{t=1}^{T}\sum_{n=1}^{N^t}\sum_{s:\mu^t_{n,s}>\mu^t_{n,*}}^{S^t_n} \Delta_n \mathbb{E}\left\{(\mathcal{O}(1) + Tf(e)) + \left\lceil \frac{4\beta\Delta_n\left(\left(1-A^t_*\right)\left(1-H^t_*\right)\right)^2 \ln T}{\left(\mu_*\left(1-A^t_*\right)\left(1-H^t_*\right) - \mu_s\left(1-A^t_s\right)\left(1-H^t_s\right)\right)^2} \right\rceil \right\}. \tag{21}$$

Proof: See the Appendix. ∎

Remarks: Based on Theorem 1, we find that more time slots and vehicular tasks cause larger learning regret. The reason is that the time slots and tasks increase the times of task offloading, which raises the number of incorrect offloading decisions. Additionally, when there are many SeVs with poor offloading performance, the selection space of task offloading enlarges, and the SeVs with poor offloading performance become easier to be selected. This leads to larger learning regret. Furthermore, Theorem 1 shows that the proposed LTO-ATA algorithm can provide a performance guarantee under incomplete offloading information, and the performance is also related to SeVs' ability and trustfulness.

VI. Performance Evaluation

In this section, we conduct simulations to evaluate the proposed algorithm and analyze the total learning regret.

TABLE II
PARAMETER SETTINGS

Parameter description	Value
the number of vehicles	500
the V2V communication radius	100 meters
the input task data bits of b_n^t	[0.2, 1] Mbits
the task computational intensity	1000 Cycles/bit
the task deadline	(25, 1500) ms
the maximum computing resources	[1, 5] GHz
the channel bandwidth W	10 MHz
the transmission power P	0.1 W
the channel gain coefficient of A	-17.8 dB

A. Simulation Setup

We use the urban vehicle trajectory data set [51] in Futian District, Shenzhen, from 22°31′N to 22°33′N, from 114°2′E to 114°7′E. It records the GPS coordinates of 500 vehicles, logged approximately every 10 s, from 20 August 2019, to 9 September 2019. These trajectories are used to simulate the vehicle traveling in the urban vehicular networks. The simulations of computation offloading are conducted based on the *MindSpore* framework platform. We set the V2V communication radius as 100 m. Taking the computing-intensive vehicular video streaming tasks as an instance, in which each task is regarded as a one-second video clip. For a video clip with 176 –144 video resolution, 24.8k pixels per frame and 25 frames per second, its data size is 0.62 Mbits [52]. Without loss of generality, we assume that input task data bits of b_n^t follow uniform distribution within [0.2, 1] Mbits, computational intensity is set as 1000 Cycles/bit and the task deadline is evenly distributed with (25, 1500) ms. The maximum computing resources of each vehicle are randomly distributed within [1, 5] GHz. In addition, the channel bandwidth W is set as 10 MHz, and the transmission power P is set as 0.1 W. The channel gain between SeV s and TaV u is modeled as $g_{u,s}^t = A(l_{u,s}^t)^{-2}$, where A is -17.8 dB [18], and $l_{u,s}^t$ denotes the distance between TaV u and SeV s at time slot t. The parameter settings are listed, shown in Table II.

We compare the proposed LTO-ATA algorithm with the following algorithms.

1) *Computation Offloading With the Complete Offloading Information (CI):* This offloading assumes that accurate transmission rate information and computing resource information are known prior to TaVs. Guided by the complete offloading information, TaVs make offloading decisions without requiring information learning [23].
2) *UCB-Enabled Computation Offloading With SeV's Ability Awareness (UCB-AA):* This algorithm considers SeVs' ability awareness in a UCB-enabled learning-based offloading paradigm. The index function is defined as $\Psi_s^t = (1 - A_s^t)(\bar{T}_{n,s}^{t-1} - \sqrt{\beta \ln t / K_{n,s}^{t-1}})$ [53]. When implementing V2V computation offloading, TaVs incline to select SeVs with large abilities to maintain QoS of SeVs.
3) *UCB-Enabled Computation Offloading With SeV's Trustfulness Awareness (UCB-TA):* This algorithm concerns SeVs' trustfulness awareness in a UCB-enabled learning-based offloading paradigm. The index function is defined as $\Psi_s^t = (1 - H_s^t)(\bar{T}_{n,s}^{t-1} - \sqrt{\beta \ln t / K_{n,s}^{t-1}})$ [11]. In V2V computation offloading, TaVs are prone to selecting SeVs with high trustfulness to reduce privacy leakage risks.
4) *UCB1:* This is a traditional UCB algorithm, and the index function is defined as $\Psi_s^t = \bar{T}_{n,s}^{t-1} - \sqrt{\beta \ln t / K_{n,s}^{t-1}}$ [50]. In the nonadjustable learning-based computation offloading algorithm, neither the SeV's ability nor trustfulness awareness is taken into consideration.

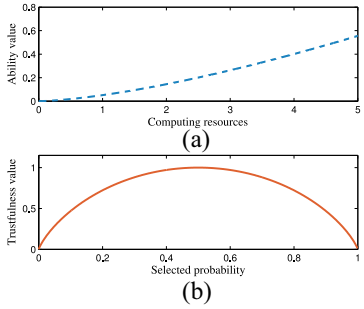


Fig. 2. Illustration of (a) ability and (b) trustfulness of a SeV.

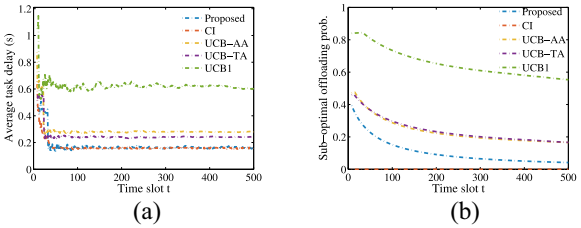


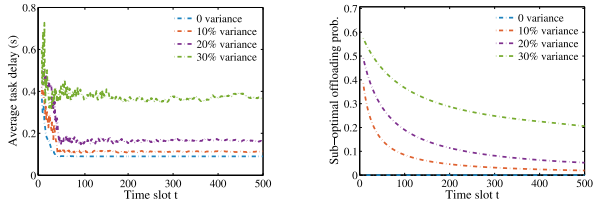
Fig. 3. Comparison of different algorithms in (a) average task delay and (b) suboptimal offloading probability.

B. Ability and Trustfulness Awareness

Fig. 2 illustrates the ability and trustfulness of a SeV. From the figure, we find that more computing resources contribute to a larger ability. Furthermore, we observe that a SeV has the largest trustfulness value when the selected probability is 0.5 (i.e., the largest uncertainty). The reason is that large uncertainty effectively decreases the probability of the attacker's correct inference. In V2V computation offloading, TaVs are prone to selecting SeVs with high abilities and trustfulness to maintain the QoS of SeVs and reduce privacy leakage risks.

C. Performance Comparison

Fig. 3(a) shows the impact of the time slots (i.e., learning times) on average task delay under different algorithms. As the learning times increase, the optimal SeV can be found and, thus, the average task delay decreases. The proposed algorithm achieves smaller task delay than other learning-based algorithms. In particular, UCB1 suffers from excessive delay without awareness of SeV's ability and trustfulness. Fig. 3(b) presents the impact of learning times on the probability of suboptimal offloading under various algorithms. Small learning times lead to a large probability of suboptimal offloading.



(a) (b)

Fig. 4. Comparison of different variances in (a) average task delay and (b) suboptimal offloading probability.

TABLE III
CANDIDATE SEV SET IN V2V COMPUTATION OFFLOADING

SeV	1	2	3	4	5	6	7	8
F^s (GHZ)	2	1.5	1.8	2.3	2.8	3.6	3.0	4.2
Epoch 1 (1 $\sim$ 300)	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\times$
Epoch 2 (301 $\sim$ 600)	$\surd$	$\times$	$\times$	$\surd$	$\surd$	$\times$	$\surd$	$\times$
Epoch 3 (601 $\sim$ 900)	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\surd$	$\times$	$\surd$

As learning times increase, the suboptimal probabilities of the learning-based algorithms reduce. As the ideal case, CI always maintains the optimal offloading since the complete offloading information is available. The results in Fig. 3 indicate that the overall performance is tightly related to the learning times. For example, at least 50 times learning is essential for the proposed algorithm to realize satisfactory delay performance.

Fig. 4 presents the comparison between different variances in the average task delay and suboptimal offloading probability. Large observation variance hampers the deviation of optimal SeV and, thus, prolongs the task delay and increases the probability of suboptimal offloading. When there are no observation variances, each TaV enables to find out the optimal SeV after once selection of each candidate SeV. However, observation variance is commonly inevitable due to vehicular mobility, complicated network environment, and diverse requested tasks. In our simulations, we set the observation variance as 20%.

D. Dynamic SeV Sets

We discuss the impact of dynamic SeV sets on average task delay under different algorithms. As shown in Table III, we focus on a representative TaV within three epochs, and each epoch occupies 300 time slots in our simulation. For the TaV, its candidate SeVs remain fixed at each epoch, while the candidate SeVs change across epochs. As such, at the beginning of each epoch, the TaV needs to explore the newly appeared SeVs for probably less task delay. If a SeV is available for the TaV during a specified epoch, it is marked as "$\surd$"; otherwise, the SeV is marked as "$\times$." We present the candidate SeV set in Table III. In epochs 2 and 3, these learning-based algorithms need to restart for seeking more suitable SeVs at time slots 201 and 601, respectively. For example, the SeV 8 has the maximum computing resources and exactly appears in epoch 3. At the beginning of time slot 601, the TaV updates its candidate SeVs and restarts to learn the offloading performance of the SeVs. After several times learning, the TaV offloads computing-intensive tasks to the SeV 8 within epoch 3 and, thus, achieves less task delay.

Fig. 5 illustrates the performance comparison of the proposed algorithm with the existing ones under dynamic SeV sets. From Fig. 5, we find that the proposed algorithm shows superiority compared with other learning-based algorithms in dynamic SeV sets. The reason is that both SeV's ability and SeV's trustfulness are taken into account in the proposed learning-based V2V computation offloading algorithm. In this way, the offloading efficiency and safety are significantly improved and, hence, contributing to better delay performance.

E. Learning Regret

Fig. 6(a) shows the comparison of different algorithms in the learning regret. Specifically, both UCB-AA and UCB-TA achieve lower learning regret compared with UCB1. By jointly considering SeV's ability and SeV's trustfulness, the proposed LTO-ATA algorithm achieves better performance compared with other learning-based algorithms. More explicitly, the proposed algorithm decreases the learning regret by 72.92%, 42%, and 39.81% from that of UCB1, UCB-AA, and UCB-TA, respectively. Since CI has complete offloading information, it enables to make the optimal offloading decision directly without learning. Furthermore, we observed that the regret curves sublinearly rise with the increasing learning times. This observation demonstrates that the simulation results are consistent with the proposed theorem (Theorem 1) (detailed in Section V-C).

Fig. 6(b) presents the impact of observation variance on the learning regret. Clearly, 30% variance incurs larger learning regret than that of 20% and 10%. The reason is that a larger observation variance requires more learning times to find the optimal SeV. These features slow down the learning convergence and incur excessive learning regret. When the observation variance is zero, the TaV enables the optimal V2V computation offloading after one connection between the TaV and candidate SeVs. It is noted that no observation variance is impractical in the real world.

Fig. 6(c) shows the impact of the weight factor β on the learning regret. The parameter β denotes the exploration weight in the learning-based algorithms, which is essential to achieve the tradeoff between exploration and exploitation. When $\beta = 0$, there is no exploration in learning. As a result, the learning-based algorithms are easily trapped by suboptimal SeVs and incur large learning regret. When $\beta > 0$, the learning regret is less than that of $\beta = 0$. Furthermore, $\beta = 0.5$ conducts a larger learning regret than that of $\beta = 0.1$. The reason is that $\beta = 0.5$ pays excessive attention to the exploration in learning, leading to degraded delay performance. In our settings, only a small number of explorations are beneficial to find the optimal SeV.

VII. CONCLUSION

In this article, we investigated a learning-based approach for V2V computation offloading. Specifically, we proposed a learning-based V2V computation offloading algorithm, which is enhanced through the ability awareness and trustfulness

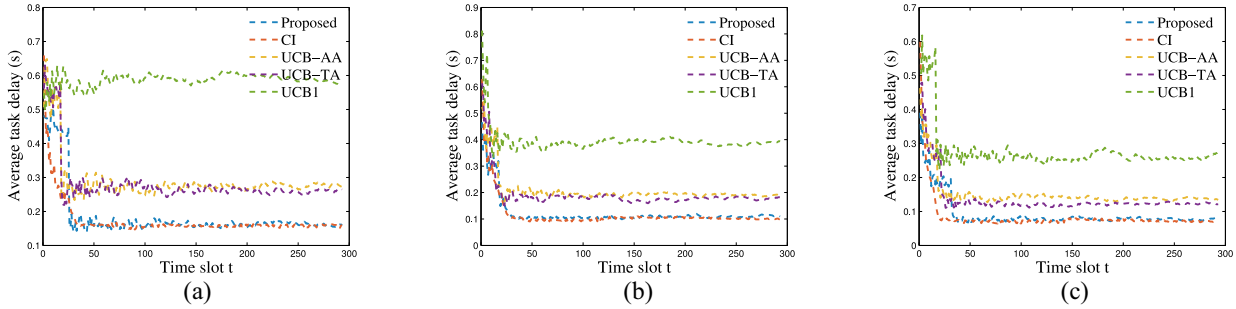


Fig. 5. Comparison of different algorithms in the average task delay under the dynamic SeV sets. (a) Epoch 1. (b) Epoch 2. (b) Epoch 3.

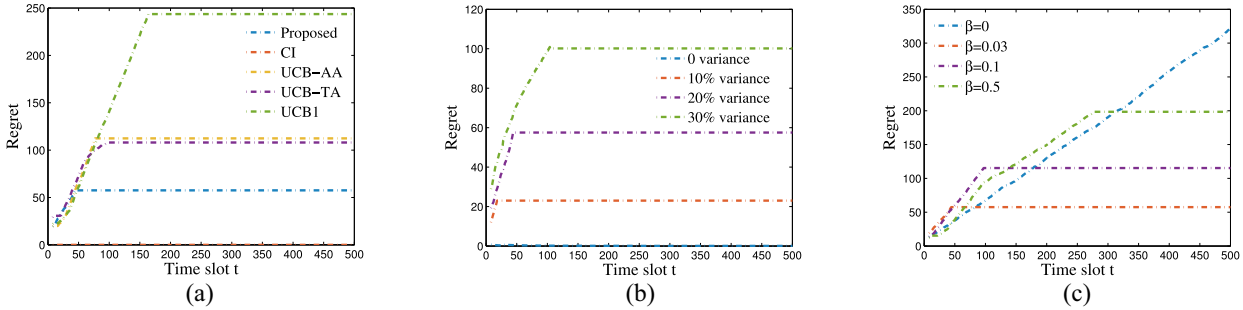


Fig. 6. Comparison of the learning regret under different (a) algorithms, (b) variances, and (c) parameter β.

awareness of SeVs. Within the learning-based V2V computation offloading algorithm, TaVs are able to learn the offloading performance from the candidate SeVs and make computation offloading decisions without requiring the complete offloading information in advance. Besides, both ability awareness and trustfulness awareness of SeVs were emphasized in the proposed algorithm, which facilitates effective and safe V2V computation offloading. After that, we carried out extensive simulations and the results showed the superiority of the proposed algorithm compared with other algorithms. One limitation of our work is that computing resources are randomly allocated from a SeV to the TaVs. To address this issue, we will extend our work by combining resource allocation and offloading decisions into V2V computation offloading problem in our future work.

APPENDIX
PROOF OF THEOREM 1

We analyze the total learning regret by bounding $\mathbb{E}\{K_{n,s}^T\}$. To this end, we introduce an indicator function as $\mathbb{I}\{x\} \in \{0, 1\}$. When the event x is true, $\mathbb{I}\{x\} = 1$; otherwise, $\mathbb{I}\{x\} = 0$. Let ϵ be a positive integer, denoting the selected times of SeV s. Based on [49], we derive the following inequalities:

$$\begin{aligned} K_{n,s}^T &= \sum_{t=S_n^t+1}^{T} \mathbb{I}\{\xi_n^t = s\} + 1 \\ &\le \sum_{t=S_n^t+1}^{T} \mathbb{I}\left\{\xi_n^t = s, K_{n,s}^{t-1} \ge \epsilon\right\} + \epsilon \\ &\le \sum_{t=1}^{\infty} \sum_{K_{n,*}^t=1}^{t-1} \sum_{K_{n,s}^t=\epsilon}^{t-1} \mathbb{I}\Big\{\big(1-A_*^t\big)\big(1-H_*^t\big) \\ &\qquad \left(\bar{T}_{n-1,*}^t - \sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}}\right) \le \big(1-A_s^t\big)\big(1-H_s^t\big) \\ &\qquad \left(\bar{T}_{n-1,s}^t - \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}\right)\Big\} + \epsilon. \end{aligned} \tag{22}$$

Then, we introduce the following two events:

$$V1 \triangleq \left\{\bar{T}_{n,s}^{t-1} \in \left[\mu_s - \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}, \mu_s + \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}\right]\right\} \tag{23}$$

$$\begin{aligned} V2 \triangleq \Bigg\{ &\big(1-A_*^t\big)\big(1-H_*^t\big)\left(\bar{T}_{n,*}^{t-1} - \sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}}\right) \\ &\le \big(1-A_s^t\big)\big(1-H_s^t\big)\left(\bar{T}_{n,s}^{t-1} - \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}\right)\Bigg\}. \end{aligned} \tag{24}$$

Based on the Chernoff–Hoeffding bound, we obtain the occurrence possibility of event V_1

$$\mathbb{P}\left\{\mu_s - \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}} \le \bar{T}_{n,s}^{t-1} \le \mu_s + \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}\right\} = 1 - \frac{2}{t^{2\beta}}. \tag{25}$$

Correspondingly, we obtain the nonoccurrence possibility of event V_1, expressed as: $P(\mathring{V}_1) = 2/t^{2\beta}$.

Following that, we analyze the event V2. When one of the following cases holds, we recognize that event V2 occurs.

Case i): Apart from the optimal SeV, the performance of other SeVs is underestimated. We define the event as

$$C_1 \triangleq \left\{\big(1-A_s^t\big)\big(1-H_s^t\big)\bar{T}_{n,s}^{t-1} > \mu_s + \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}\right\}. \tag{26}$$

In this case, we obtain $P(C1|V_1) = 0$.

Case ii): The performance of the optimal SeV is overestimated. The event is expressed as

$$C_{2,1} \triangleq \left\{ \left(1-A_*^t\right)\left(1-H_*^t\right)\bar{T}_{n,*}^{t-1} < \mu_* - \sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}} \right\}. \quad (27)$$

If $A_*^t = H_*^t = 0$, we derive $P(C_{2,1}|V1) = 0$. When both $A_*^t > 0$ and $B_*^t > 0$, we introduce the following event:

$$C_{2,2} \triangleq \left\{ \left(1-A_*^t\right)\left(1-H_*^t\right) < \frac{\mu_* - \sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}}}{\bar{T}_{n,*}^{t-1}} \right\}. \quad (28)$$

Besides, an single event $C_{2,2}$ is not sufficient for the learning regret, we introduce the following inequality:

$$C_{2,3} \triangleq \left\{ \left(1-A_s^t\right)\left(1-H_s^t\right) > \frac{\mu_s - \sqrt{\frac{\beta \ln t}{K_{n,s}^{t-1}}}}{\bar{T}_{n,s}^{t-1}} \right\}. \quad (29)$$

After that, we define the event $C_{2,4} \triangleq C_{2,2} \cap C_{2,3}$.

Case iii): Based on the definitions of event 1 and event 2, we introduce the following event:

$$C_3 \triangleq \left\{ \mu_s > \frac{\left(1-A_*^t\right)\left(1-H_*^t\right)}{\left(1-A_s^t\right)\left(1-H_s^t\right)} \left(\mu_* - 2\sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}}\right) \right\}. \quad (30)$$

In order to find the optimal SeV, the following constraint of the confidence interval needs to be satisfied:

$$\sqrt{\frac{\beta \ln t}{K_{n,*}^{t-1}}} < \frac{\mu_*}{2} - \frac{\mu_s\left(1-A_s^t\right)\left(1-H_s^t\right)}{2\left(1-A_*^t\right)\left(1-H_*^t\right)}. \quad (31)$$

It is noted that C_3 will never happen when the confidence intervals are too small. Based on (31), we derive the selected times of SeV s for processing task n up to time slot T

$$\epsilon > \left\lceil \frac{4\beta\left(\left(1-A_*^t\right)\left(1-H_*^t\right)\right)^2 \ln T}{\left(\mu_*\left(1-A_*^t\right)\left(1-H_*^t\right) - \mu_s\left(1-A_s^t\right)\left(1-H_s^t\right)\right)^2} \right\rceil. \quad (32)$$

Based on the above analysis, we rewrite (22) as follows:

$$\begin{aligned} K_{n,s}^T &\le \epsilon + \sum_{t=1}^{\infty} \sum_{K_{n,*}^{t-1}=1}^{t-1} \sum_{K_{n,s}^{t-1}=\epsilon}^{t-1} \mathbb{I}\Big\{P(C1|V_1) + P\left(C_{2,1}|V_1\right) \\ &\quad + P\left(C_{2,4}|V_1\right) + P(C_3|V_1) + P\left(\mathring{V}_1\right)\Big\} \\ &\le \left\lceil \frac{4\beta\left(\left(1-A_*^t\right)\left(1-H_*^t\right)\right)^2 \ln T}{\left(\mu_*\left(1-A_*^t\right)\left(1-H_*^t\right) - \mu_s\left(1-A_s^t\right)\left(1-H_s^t\right)\right)^2} \right\rceil \\ &\quad + \sum_{K_{n,*}^t=1}^{t-1} \sum_{K_{n,s}^t=\epsilon}^{t-1} \mathbb{I}\left\{P\left(C_{2,4}\right)\right\} + \mathcal{O}(1). \end{aligned} \quad (33)$$

Owing to the changing of the average task delay and confidence interval, it is challenging to derive the closed-form expression for the event $C_{2,4}$. Based on the analysis from [11], we conclude that the occurrence times of the event $C_{2,4}$ is a liner function of $Tf(e)$, where $f(e)$ is a prediction error function. When $e = 1$, $f(e) = 0$. As such, we obtain the upper bound

$$\begin{aligned} K_{n,s}^T \le & \left\lceil \frac{4\beta\left(\left(1-A_*^t\right)\left(1-H_*^t\right)\right)^2 \ln t}{\left(\mu_*\left(1-A_*^t\right)\left(1-H_*^t\right) - \mu_s\left(1-A_s^t\right)\left(1-H_s^t\right)\right)^2} \right\rceil \\ & + \mathcal{O}(1) + Tf(e). \end{aligned} \quad (34)$$

Finally, we use the upper bound in (34) to substitute $K_{n,s}^T$ in (20) and, thus, we obtain the total learning regret as shown in Theorem 1.

REFERENCES

[1] J. Liu et al., "RL/DRL meets vehicular task offloading using edge and vehicular cloudlet: A survey," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8315–8338, Jun. 2022.

[2] L. Chen and J. Xu, "Task replication for vehicular cloud: Contextual combinatorial bandit with delayed feedback," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 748–756.

[3] X. Zhang, M. Peng, S. Yan, and Y. Sun, "Joint communication and computation resource allocation in fog-based vehicular networks," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13195–13208, Aug. 2022.

[4] A. Bozorgchenani, S. Maghsudi, D. Tarchi, and E. Hossain, "Computation offloading in heterogeneous vehicular edge networks: Online and off-policy bandit solutions," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4233–4248, Dec. 2022.

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

[6] Z. Zhou, H. Liao, X. Zhao, B. Ai, and M. Guizani, "Reliable task offloading for vehicular fog computing under information asymmetry and information uncertainty," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8322–8335, Sep. 2019.

[7] Y. Qi, Y. Zhou, Y.-F. Liu, L. Liu, and Z. Pan, "Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17762–17777, Dec. 2021.

[8] X. Zhu, Y. Luo, A. Liu, M. Z. A. Bhuiyan, and S. Zhang, "Multiagent deep reinforcement learning for vehicular computation offloading in IoT," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9763–9773, Jun. 2021.

[9] S. Xu, C. Guo, R. Q. Hu, and Y. Qian, "BlockChain-inspired secure computation offloading in a vehicular cloud network," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14723–14740, Aug. 2022.

[10] A. Chopra, A. U. Rahman, A. W. Malik, and S. D. Ravana, "Adaptive-learning-based vehicle-to-vehicle opportunistic resource-sharing framework," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12497–12504, Jul. 2022.

[11] H. Liao, Y. Mu, Z. Zhou, M. Sun, Z. Wang, and C. Pan, "Blockchain and learning-based secure and intelligent task offloading for vehicular fog computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4051–4063, Jul. 2021.

[12] Y. Liu, Y. Li, Y. Niu, and D. Jin, "Joint optimization of path planning and resource allocation in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 9, pp. 2129–2144, Sep. 2020.

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

[14] T. Bahreini, M. Brocanelli, and D. Grosu, "VECMAN: A framework for energy-aware resource management in vehicular edge computing systems," *IEEE Trans. Mobile Comput.*, early access, Jun. 15, 2021, doi: 10.1109/TMC.2021.3089338.

[15] J. Shi, J. Du, Y. Shen, J. Wang, J. Yuan, and Z. Han, "DRL-based V2V computation offloading for blockchain-enabled vehicular networks," *IEEE Trans. Mobile Comput.*, early access, Feb. 23, 2022, doi: 10.1109/TMC.2022.3153346.

[16] Y.-J. Ku, S. Baidya, and S. Dey, "Adaptive computation partitioning and offloading in real-time sustainable vehicular edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13221–13237, Dec. 2021.

[17] T. Ouyang, R. Li, X. Chen, Z. Zhou, and X. Tang, "Adaptive user-managed service placement for mobile edge computing: An online learning approach," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 1468–1476.

[18] 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.
[19] Z. Zhou et al., "Learning-based URLLC-aware task offloading for Internet of Health Things," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 2, pp. 396–410, Feb. 2021.
[20] J. Yan, S. Bi, Y. J. Zhang, and M. Tao, "Optimal task offloading and resource allocation in mobile-edge computing with inter-user task dependency," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 235–250, Jan. 2020.
[21] W. Chang, Y. Xiao, W. Lou, and G. Shou, "Offloading decision in edge computing for continuous applications under uncertainty," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 6196–6209, Sep. 2020.
[22] R. Lin et al., "Distributed optimization for computation offloading in edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8179–8194, Dec. 2020.
[23] C. Liu, K. Liu, S. Guo, R. Xie, V. C. S. Lee, and S. H. Son, "Adaptive offloading for time-critical tasks in heterogeneous Internet of Vehicles," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7999–8011, Sep. 2020.
[24] H. Tang, H. Wu, G. Qu, and R. Li, "Double deep Q-network based dynamic framing offloading in vehicular edge computing," *IEEE Trans. Netw. Sci. Eng.*, early access, May 5, 2022, doi: 10.1109/TNSE.2022.3172794.
[25] C. Tang, C. Zhu, H. Wu, Q. Li, and J. J. P. C. Rodrigues, "Toward response time minimization considering energy consumption in caching-assisted vehicular edge computing," *IEEE Internet Things J.*, vol. 9, no. 7, pp. 5051–5064, Apr. 2022.
[26] C. Tang and H. Wu, "Joint optimization of task caching and computation offloading in vehicular edge computing," *Peer-to-Peer Netw. Appl.*, vol. 15, no. 2, pp. 854–869, 2022.
[27] Y. Liu et al., "Dependency-aware task scheduling in vehicular edge computing," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4961–4971, Jun. 2020.
[28] Q. Luo, C. Li, T. H. Luan, W. Shi, and W. Wu, "Self-learning based computation offloading for Internet of Vehicles: Model and algorithm," *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 5913–5925, Sep. 2021.
[29] B. Shang, L. Liu, and Z. Tian, "Deep learning-assisted energy-efficient task offloading in vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9619–9624, Sep. 2021.
[30] Y. Lin, Y. Zhang, J. Li, F. Shu, and C. Li, "Popularity-aware online task offloading for heterogeneous vehicular edge computing using contextual clustering of bandits," *IEEE Internet Things J.*, vol. 9, no. 7, pp. 5422–5433, Apr. 2022.
[31] M. Yang, H. Zhu, H. Wang, Y. Koucheryavy, K. Samouylov, and H. Qian, "An online learning approach to computation offloading in dynamic fog networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1572–1584, Feb. 2021.
[32] C. Tekin and M. Liu, "Online learning of rested and restless bandits," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5588–5611, Aug. 2012.
[33] H. Liu, K. Liu, and Q. Zhao, "Learning in a changing world: Restless multiarmed bandit with unknown dynamics," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1902–1916, Mar. 2013.
[34] C. Wang, S. Zhang, Y. Chen, Z. Qian, J. Wu, and M. Xiao, "Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2020, pp. 257–266.
[35] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.
[36] M. Sheng, Y. Dai, J. Liu, N. Cheng, X. Shen, and Q. Yang, "Delay-aware computation offloading in NOMA MEC under differentiated uploading delay," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2813–2826, Apr. 2020.
[37] J. Peng, H. Qiu, J. Cai, W. Xu, and J. Wang, "D2D-assisted multi-user cooperative partial offloading, transmission scheduling and computation allocating for MEC," *IEEE Trans. Wireless Commun.*, vol. 20, no. 8, pp. 4858–4873, Aug. 2021.
[38] N. Eshraghi and B. Liang, "Joint offloading decision and resource allocation with uncertain task computing requirement," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 1414–1422.
[39] T. Li, D. Baumberger, and S. Hahn, "Efficient and scalable multiprocessor fair scheduling using distributed weighted round-robin," *ACM SIGPLAN Notices*, vol. 44, no. 4, pp. 65–74, 2009.
[40] Y. Sun, S. Zhou, and Z. Niu, "Distributed task replication for vehicular edge computing: Performance analysis and learning-based algorithm," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1138–1151, Feb. 2021.
[41] P. Dai, K. Hu, X. Wu, H. Xing, and Z. Yu, "Asynchronous deep reinforcement learning for data-driven task offloading in MEC-empowered vehicular networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2021, pp. 1–10.
[42] L. Liu and M. Gruteser, "EdgeSharing: Edge assisted real-time localization and object sharing in urban streets," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2021, pp. 1–10.
[43] Y. Li, X. Wang, X. Gan, H. Jin, L. Fu, and X. Wang, "Learning-aided computation offloading for trusted collaborative mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2833–2849, Dec. 2020.
[44] W. Kong, X. Li, L. Hou, J. Yuan, Y. Gao, and S. Yu, "A reliable and efficient task offloading strategy based on multifeedback trust mechanism for IoT edge computing," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13927–13941, Aug. 2022.
[45] H. Chen et al., "Practical membership inference attack against collaborative inference in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 477–487, Jan. 2022.
[46] J. Li et al., "Drive2friends: Inferring social relationships from individual vehicle mobility data," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5116–5127, Jun. 2020.
[47] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, and L. Liu, "Edge computing in VANETs-an efficient and privacy-preserving cooperative downloading scheme," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1191–1204, Jun. 2020.
[48] X. Han et al., "Reliability-aware joint optimization for cooperative vehicular communication and computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5437–5446, Aug. 2021.
[49] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, nos. 2–3, pp. 235–256, 2002.
[50] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Nov. 2017.
[51] Z. Xiao et al., "TrajData: On vehicle trajectory collection with commodity plug-and-play OBU devices," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 9066–9079, Sep. 2020.
[52] A. Anjum, T. Abdullah, M. F. Tariq, Y. Baltaci, and N. Antonopoulos, "Video stream analysis in clouds: An object detection and classification framework for high performance video analytics," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1152–1167, Oct.–Dec. 2019.
[53] Z. Su, Y. Hui, and T. H. Luan, "Distributed task allocation to enable collaborative autonomous driving with network softwarization," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2175–2189, Oct. 2018.

Xingxia Dai received the B.S. degree in communication engineering from Xiangtan University, Xiangtan, China, in 2018. She is currently pursuing the Ph.D. degree in computer science and technology with Hunan University, Changsha, China.

Her current research interests include Internet of Vehicles and mobile-edge computing.

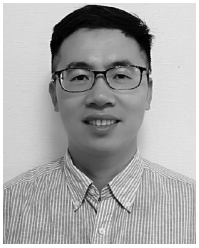
Zhu Xiao (Senior Member, IEEE) received the M.S. and Ph.D. degrees in communication and information system from Xidian University, Xi'an, China, in 2007 and 2009, respectively.

He was a Research Fellow with the Department of Computer Science and Technology, University of Bedfordshire, Luton, U.K., from 2010 to 2012. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include mobile communications, wireless localization, Internet of Vehicles, and trajectory data mining.

Hongbo Jiang (Senior Member, IEEE) received the Ph.D. degree from Case Western Reserve University, Cleveland, OH, USA, in 2008.

He is currently a Full Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. He was a Professor with Huazhong University of Science and Technology, Wuhan, China. His research concerns computer networking, especially algorithms and protocols for wireless and mobile networks.

Prof. Jiang was the Editor of the IEEE/ACM TRANSACTIONS ON NETWORKING, the Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, and the Associate Technical Editor for the *IEEE Communications Magazine*. He is an Elected Member of the Academia Europaea, Fellow IET, BCS, and AAIA.



Hongyang Chen (Senior Member, IEEE) received the B.S. and M.S. degrees from Southwest Jiaotong University, Chengdu, China, in 2003 and 2006, respectively, and the Ph.D. degree from The University of Tokyo, Tokyo, Japan, in 2011.

From 2011 to 2020, he was a Researcher with Fujitsu Ltd., Tokyo. He is currently a Senior Research Expert with Zhejiang Lab, Hangzhou, China. He is an Adjunct Professor with Hangzhou Institute for Advanced Study, The University of Chinese Academy of Sciences, Hangzhou, and Zhejiang University, Hangzhou. He has authored or coauthored 100+ refereed journal and conference papers in the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON SIGNAL PROCESSING, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON COMMUNICATIONS, CIKM, and has been granted 20+ PCT patents. His research interests include data-driven intelligent systems, graph machine learning, big data mining, and intelligent computing.

Dr. Chen was the Editor of the IEEE Journals and the symposium chair or a special session organizer for some flagship conferences. He was a leading Guest Editor of the IEEE JOURNAL ON SELECTED TOPICS OF SIGNAL PROCESSING on tensor decomposition. He is currently an Associate Editor of the IEEE INTERNET OF THINGS JOURNAL. He has been selected as the Distinguished Lecturer of the IEEE Communication Society from 2021 to 2022.

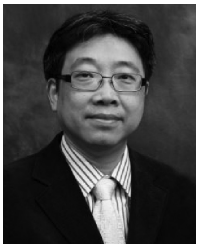
Geyong Min (Member, IEEE) received the B.Sc. degree in computer science from Huazhong University of Science and Technology, Wuhan, China, in 1995, and the Ph.D. degree in computing science from the University of Glasgow, Glasgow, U.K., in 2003.

He is a Professor of High Performance Computing and Networking with the Department of Computer Science, College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, U.K. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high-performance computing, ubiquitous computing, modeling, and performance engineering.

Schahram Dustdar (Fellow, IEEE) received the Ph.D. degree in business informatics from the University of Linz, Linz, Austria, in 1992.

He is currently a Full Professor of Computer Science (Informatics) with a focus on Internet technologies heading the Distributed Systems Group, TU Wien, Wein, Austria.

Prof. Dustdar was a recipient of the ACM Distinguished Scientist Award in 2009 and the IBM Faculty Award in 2012. He has been a member of the IEEE Conference Activities Committee since 2016, the Section Committee of Informatics of the Academia Europaea since 2015, and the Academia Europaea: The Academy of Europe, Informatics Section since 2013. He has been the Chairman of the Informatics Section of the Academia Europaea since December 2016. He is an Associate Editor of the IEEE TRANSACTIONS ON SERVICES COMPUTING, *ACM Transactions on the Web*, and *ACM Transactions on Internet Technology*. He is on the Editorial Board of IEEE.



Jiannong Cao (Fellow, IEEE) received the M.Sc. and Ph.D. degrees in computer science from Washington State University, Pullman, WA, USA, in 1986 and 1990, respectively.

He is currently a Chair Professor with the Department of Computing, The Hong Kong Polytechnic University (PolyU), Hong Kong. He is also the Dean of the Graduate School, the Director of the Research Institute of Artificial Intelligent of Things and the Internet and Mobile Computing Laboratory, and the Vice Director of the University's Research Facility in Big Data Analytics, PolyU. He has coauthored five books, coedited nine books, and published over 500 papers in major international journals and conference proceedings. His research interests include distributed systems and blockchain, wireless sensing and networking, big data and machine learning, and mobile cloud and edge computing.

Dr. Cao is a member of the Academia Europaea and an ACM Distinguished Member.