Learning Based Energy Efficient Task Offloading for Vehicular Collaborative Edge Computing

Peng Qin, *Member, IEEE*, Yang Fu, Guoming Tang, *Member, IEEE*, Xiongwen Zhao, *Senior Member, IEEE*, and Suiyan Geng

***Abstract*—Extensive delay-sensitive and computation-intensive tasks are involved in emerging vehicular applications. These tasks can hardly be all processed by the resource constrained vehicle alone, nor fully offloaded to edge facilities (like road side units) due to their incomplete coverage. To this end, we refer to the new paradigm of vehicular collaborative edge computing (VCEC) and make the best use of vehicles' idle and redundant resources for energy consumption reduction within the VCEC system. To realize this target, we are faced with several nontrivial challenges, including short-term decision making coupled with long-term queue delay constraints, information uncertainty, and task offloading conflicts. Accordingly, we apply Lyapunov optimization to decouple the original problem into three sub-problems and then tackle them one by one: the first sub-problem is resolved by Lagrange multiplier method; the second is handled by UCB learning-matching approach; the third is addressed by a carefully designed greedy method. Scenarios without volatility and real-world road topology with realistic vehicular traffics are utilized to evaluate the proposed solution. Results from extensive numerical simulations demonstrate that our solution can achieve superior performances compared with the benchmark methods, in terms of energy consumption, learning regret, task backlog, and end-to-end delay.**

***Index Terms*—Vehicular collaborative edge computing (VCEC), energy efficient, task offloading, Lyapunov optimization, matching learning.**

I. Introduction

ALONG with the dramatic development of vehicular technology, more stringent requirements at vehicle have been put forward, particularly for timely task execution. A large number of vehicular services and applications have emerged, such as vehicular video streaming, AR/VR-based driving assistance, and remote intelligent control, which generate extensive delay-sensitive and computation-intensive tasks at the vehicle side. [1]. Due to its limited computational capability, however, the vehicle itself can hardly process these arriving tasks promptly. Offloading tasks to cloud servers for mobile cloud computing (MCC) or edge servers for mobile edge computing (MEC) could be two possible solutions [2], [3]. Nevertheless, due to the transmission delay and network congestion, traditional MCC is no longer applicable; MEC is also infeasible considering the high deployment cost of edge facilities (e.g., road side units or RSUs) for complete coverage [4].

To this end, the running vehicles around have great potential to become mobile edge servers and provide vehicular services. Particularly, the equipped computing and storage resources of vehicles are increasing and upgrading rapidly, giving rise to collaborative processing of offloaded tasks [5]. Meanwhile, the fast growth of vehicular networks and widely distributed vehicles on road make it possible for a vehicle to offload tasks to nearby neighbors within its communication range. These nearby vehicles with similar routes could share their idle computing and storage resources for reuse, which leads to a new computing paradigm of vehicle collaborative edge computing (VCEC) [5]–[7]. Specifically, VCEC network is composed of user vehicles (UVs) and collaborative vehicles (CVs). Computing tasks generated by a UV can be offloaded to a suitable CV. The CV that receives offloaded tasks performs collaborative edge computing, and results are finally fed back to the corresponding UV by the same channel or RSUs in case of loss of contact. Unlike vehicular *ad hoc* network [8], we assume that task offloading and result feedback are executed through the direct vehicle-to-vehicle (V2V) manner, which is more appropriate for timely reliable service delivery and flexible resource allocation [9].

To make the best use of local and neighbor vehicle's computing resources, each arriving task is split into two irrelevant parts, i.e., local computing part and offloading part, respectively. After a UV allocates local computing resources, it selects a suitable CV for task offloading considering the dynamics of vehicles. During this process, the energy consumption of involved vehicles is one of the major concerns. On the one hand, the difference of computing resource assignment for (local or offloading) task execution could incur different energy consumption (at UV or CV). On the other hand, the channel condition (given by channel state information or CSI) between UV and CV also bring differences to the consumed energy in task transmission. Moreover, to build environment-harmonic transportation systems and green cities, the energy consumption of vehicles (and other emerging

Manuscript received 20 March 2022; accepted 26 April 2022. Date of publication 29 April 2022; date of current version 15 August 2022. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 2021MS002, and in part by the State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources under Grant LAPS21018. The review of this article was coordinated by Prof. Zhu Han. *(Corresponding authors: Peng Qin; Guoming Tang.)*

Peng Qin, Yang Fu, Xiongwen Zhao, and Suiyan Geng are with the State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources, School of Electrical and Electronic Engineering, North China Electric Power University, Beijing 102206, China, and also with the Hebei Key Laboratory of Power Internet of Things Technology, North China Electric Power University, Baoding, Hebei 071003, China (e-mail: qinpeng@ncepu.edu.cn; 120181070402@ncepu.edu.cn; zhaoxw@ncepu.edu.cn; gsuiyan@ncepu.edu.cn).

Guoming Tang is with the Peng Cheng Laboratory, Shenzhen, 518000, China (e-mail: tanggm@pcl.ac.cn).

Digital Object Identifier 10.1109/TVT.2022.3171344

smart devices) has become a great concern [10]. Thus, how to i) strategically split the task at UV to local and offloading parts and ii) appropriately select CVs for task transmission and offloading part execution are key problems in achieving a minimized energy consumption of the whole VCEC system.

In addition to energy reduction, we also look into other challenges in tackling the joint task offloading and resource allocation of VCEC. (1) Queue delay has an important impact on end-to-end delay, which needs to be considered. Take the AR/VR-based driving assistance as an example, the enhancement of road condition video by integrating driving guidance and environmental awareness information involves long-term computation-intensive tasks. However, the optimization of resource allocation and task offloading require short-term decisions, leading to the joint optimization an NP-hard problem [11]. (2) The available edge computing resources of CV candidates are commonly uncertain to UVs in advance. Furthermore, CSI and availability of CV candidates may vary over time because of vehicles' mobility. Thus, traditional global state information based methods are no longer applicable. (3) Due to the high distributed density of vehicles, the case that multiple UVs prefer the same CV for task offloading often occurs in practice, which leads to the preference conflict. To obtain a stable offloading strategy that can satisfy all UVs' preference is nontrivial [12].

In this work, we aim to cope with the above challenges and solve the energy minimization problem by jointly considering task splitting, task offloading, and computational resource allocation. As far as we know, this is the first work to tackle the energy minimization and resource optimization problem in VCEC under the long-term constraints. Specifically, we leverage Lyapunov optimization to decouple the short-term decision making and long-term queue delay constraints and convert the original problem to three subproblems, i.e., i) task splitting and local resource allocation at UV-side, ii) task offloading selection under information uncertainty, and iii) CV-side edge computing resource assignment. We provide efficient solutions to the sub-problems based on convex optimization, machine learning, matching theory, and greedy approach, respectively. The proposed methods can achieve queue-awareness under long-term queue delay constraints, occurrence-awareness that responds to volatility, and conflict-awareness that resolves preference conflicts.

The main contributions of this paper are summarized as follows.

- We propose a VCEC system model for energy efficient task offloading under information uncertainty in vehicular network, where the tasks generated by each UV can be executed locally or offloaded to a CV for collaborative edge computing. We manage to formulate an energy efficient task offloading and resource allocation optimization problem, which also makes comprehensive and practical considerations, including task splitting and computing resource assignments at both UV and CV, task offloading decision making under information uncertainty, preference conflict of CVs and long-term queue delay.
- To cope with the optimal task offloading and resource allocation problem, we leverage Lyapunov optimization to transform the original long-term problem into three short-term determinable sub-problems. The first one is the jointly task splitting and local resource allocation at UV side, where Lagrange multiplier method is used to obtain the optimal result. The second one is the task offloading selection, for which we develop a UCB learning-matching based approach to estimate the performance of each task offloading candidate. The third one is edge computing resource allocation for CV, which is solved by a carefully designed greedy method.
- We execute extensive experiments to compare the proposed solution with existing benchmark algorithms. Scenarios without volatility and real-world road topology with realistic vehicular traffics are utilized to evaluate the queue-awareness, occurrence-awareness and conflict-awareness. The results demonstrate that our solution can achieve superior performance in terms of energy consumption, learning regret, task backlog, and end-to-end delay. The results also verify that the proposed scheme can achieve good balance between energy consumption and end-to-end delay.

The rest of the paper is organized based on the following arrangement. Related work is introduced in Section II. Section III describes system model of VCEC. Problem formulation and decomposition based on principle of Lyapunov optimization are given in Section IV. Section V provides solution for each subproblem, especially the learning-matching-based task offloading strategy. Section VI includes performance evaluation results, and we finally conclude the paper in Section VII.

II. Related Works

Unprecedented computation demands led by the rapid increasing of delay-sensitive and computation-intensive tasks impose stringent requirements to next-generation vehicular networks [13], [14]. Lots of researches have been conducted on task processing in vehicular network. Paper [15] proposed an effective RSU deployment strategy, which is able to balance data delivery delay and RSU coverage range based on traffic demand. For the sake of reducing RSU cost, authors of [9] leveraged parked cars as P-RSUs and developed a novel reverse auction based incentive mechanism for parked cars recruitment. Literature [16] introduced fog computing to vehicular networks and the fog-based base station was designed to process mobile delay-sensitive tasks, which improved network flexibility and controllability. Paper [17] also studied vehicular fog computing, where computation tasks were offloaded from base station to nearby vehicles to relieve overload, and a delay minimization approach was given. Paper [18] elaborated an ad-hoc vehicular fog scheme allowing offloading intrusion detection tasks to vehicular clusters, and the proposed approach was capable of processing offloaded tasks through vehicular federation. Paper [19] proposed a QoS-based clustering algorithm considering tradeoff between QoS requirements and high speed mobility constraints. Vehicular clusters were achieved by using Ant Colony Optimization for MultiPoints Relay (MPRs) selection, and using MPR recovery algorithm to keep the network connected. Different from above works aiming at delay minimization or other objectives, in this paper, the VCEC model is proposed to make the best use of redundant computing resource at CV side for processing

offloaded tasks from UVs, and our objective is to minimize the overall power consumption bounded by long-term queue delay for building an energy-efficiency low-carbon system.

Task offloading and resource allocation have been widely studied in recent years [20], [21]. The optimal task offloading strategy for multi-user MEC network was studied in [22]. To tackle the offloading decision coupled with resource allocation, authors proposed a Gibbs Sampling algorithm. [23] proposed an offloading-based architecture to augment multi-persona performance and viability on mobile devices, and exacted optimal distribution of multi-persona components generated by a genetic-based approach. Paper [24] investigated the joint task offloading of tasks and energy in fog-enabled Internet of Things networks, and Lyapunov optimization was utilized to obtain the optimal strategies. Lyapunov optimization was also leveraged in [25] to deal with resource allocation in fog computing networks. [26] investigated a cost-effective MEC-based solution for intelligent offloading decision, which minimized computing/storage resources as well as energy consumption while augmenting the virtual mobile instances performance. In the meantime, matching theory is used as an effective tool to handle such high complexity combinatorial task offloading problems [27], [28]. Paper [29] formulated the end-user devices and fog nodes as a matching game, where computational-intensive tasks were offloaded from users to fog nodes. A deferred acceptance based approach was developed to achieve the efficient allocation in a distributed mode ensuring a stable matching result. Vehicular fog computing task offloading under information uncertainty was studied in paper [12]. However, for aforementioned works, either the long term queue delay constraint or the information uncertainty is not considered, leading to unrealistic results in practice.

UCB based machine learning approach is proven to achieve rapid convergence for sequential decision making under information uncertainty [30]–[32]. Numerical literature has focused on application of UCB in wireless network. Paper [33] developed a MAB based task offloading framework, which enabled vehicles to evaluate the potential performance of its neighboring vehicles. In order to achieve interference mitigation in non-orthogonal multiple access networks, literature [34] proposed a novel MAB based solution and the varying channel rewards across different access points was considered. By redesigning the utility function of the classic UCB algorithms, both load-awareness and occurrence-awareness were achieved. An air-ground integrated vehicular edge computing framework was proposed in [35], where authors developed an intent-aware UCB algorithm enabling vehicles to learn task offloading strategy while satisfying the ultra-reliable low-latency communication constraints. However, the long-term queue delay is ignored in all above works. Furthermore, task splitting and resource assignment at UV side are also out of consideration.

III. System Model

We consider a VCEC model as shown in Fig. 1, which is composed of UVs and CVs. Delay-sensitive and computation-intensive tasks at UV side can be processed either by local

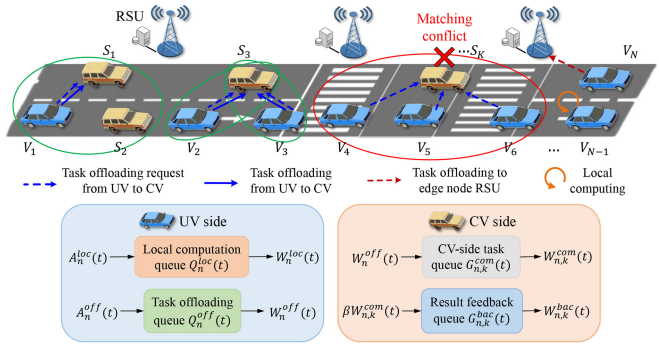


Fig. 1. The system model.

computing or task offloading. N UVs are represented by set $\mathcal{V} = \{V_1, \ldots, V_n, \ldots, V_N\}$. Vehicles with sharable computational resources are called CVs, the number of which is K, represented by set $\mathcal{S} = \{S_1, \ldots, S_k, \ldots, S_K\}$. Note that V and S are not fixed, but dynamically change according to the amount of real-time tasks, computing resources of each vehicle and reliable wireless connectivity among them. Before UV offloads tasks, it prefers to select surrounding CV candidates within the communication range and having the same driving direction to send task offloading request. The CV that receives the request will decide whether to accept based on its computing resources. For UVs that do not have offloading candidates or whose offloading request is rejected, its tasks can be offloaded to the edge server through RSU.

A time-slotted model is adopted to formulate the optimization time period, which is evenly divided into T time slots. Thus, the duration of each time slot is τ and the entire time period can be expressed by $\{1, \ldots, t, \ldots T\}$. According to [36], the model can be regarded as the quasi-static process, which means that system environment, e.g., CSI and the surrounding CVs of each UV, maintains stable in a slot but may vary over multiple slots.

We define the binary factor $a_n^k(t) = \{0, 1\}$ to represent the collaborative relationship between UV V_n and CV S_k, i.e., $a_n^k(t) = 1$ indicates that the S_k is within the communication range and is a candidate of V_n; otherwise $a_n^k(t) = 0$. Note that, there may exist a case that all arriving tasks of V_n are processed locally in the tth slot, regardless of whether CVs are within V_n's communication range, i.e., $a_n^k(t) = 0, k = 1, \ldots, K$. Another binary factor $x_n^k(t) \in \mathbf{X(t)}$is defined to indicate the actual task offloading relationship between $\mathcal{V}$ and $\mathcal{S}$, i.e., $x_n^k(t) = 1$ means V_n offloads task to S_k; otherwise $x_n^k(t) = 0$. We stipulate that arriving tasks of UV can only be offloaded to at most one CV in each time slot, i.e., $\sum_{k=1}^{K} x_n^k(t) \leq 1$. As shown in Fig. 1, there are five types of relationships between UV and CV: (1) More than one CV are available to UV and the optimal candidate is selected. (2) One CV is surrounded by multiple UVs having offloading request, and at most q_k^{max} UVs could connect with it, i.e., $\sum_{n=1}^{N} x_n^k(t) \leq q_k^{max}$. For example, $q_k^{max} = 2$ in Fig. 1. (3) When task offloading requests sent by UVs exceed q_k^{max} for S_k, matching conflict occurs. At this time, the CV has to reject some UVs' request. (4) The UV decides that all arriving tasks are processed locally. (5) When there is no CV candidate or task

offloading requests are all rejected, the UV offloads tasks to the edge server through RSU. We respectively give the queue model, communication and computation model, and the system energy consumption model in the follow subsections.

A. Task Queue Model

In this subsection, the data-partition model is utilized to divide each arriving task of UV into multiple subtasks with data size equal to A_0 [37]. Each subtask can be either executed locally or offloaded to a CV. For each user vehicle V_n, the amount of arriving tasks in the tth slot is $A_n(t)$, which we split into local computing tasks $A_n^{loc}(t)$ and offloading tasks $A_n^{off}(t)$. Therefore, the task splitting of V_n can be expressed as

$$\begin{cases} A_n(t) = A_n^{loc}(t) + A_n^{off}(t) \\ A_n^{loc}(t), A_n^{off}(t) \in \{0, A_0, 2A_0, \ldots\} \end{cases} \tag{1}$$

Assume that there are two buffer queues for storing local computing tasks and offloading tasks at each UV. The backlogs of these two queues in slot t are denoted by $Q_n^{loc}(t)$ and $Q_n^{off}(t)$, which are calculated by formula (2) and (3), respectively.

$$Q_n^{loc}(t+1) = \max\{Q_n^{loc}(t) - W_n^{loc}(t), 0\} + A_n^{loc}(t) \tag{2}$$

$$Q_n^{off}(t+1) = \max\{Q_n^{off}(t) - W_n^{off}(t), 0\} + A_n^{off}(t) \tag{3}$$

where $W_n^{loc}(t)$ and $W_n^{off}(t)$ are the amount of tasks leaving local computing queue and task offloading queue in the tth slot.

For S_k, the backlogs come from offloaded tasks. $G_{n,k}^{com}(t)$ represents the queue storing tasks offloaded from V_n in slot t given by

$$G_{n,k}^{com}(t+1) = \max\{G_{n,k}^{com}(t) - W_{n,k}^{com}(t), 0\} + W_n^{off}(t) \tag{4}$$

where $W_{n,k}^{com}(t)$ is the amount of tasks processed by S_k in slot t.

Processing results need to be fed back to the corresponding UV. Therefore, there is a result feedback queue $G_{n,k}^{bac}(t)$ at CV side expressed as

$$G_{n,k}^{bac}(t+1) = \max\{G_{n,k}^{bac}(t) - W_{n,k}^{bac}(t), 0\} + \beta W_{n,k}^{com}(t) \tag{5}$$

where $W_{n,k}^{bac}(t)$ is the data that leaves the result feedback queue, and $\beta < 1$ is the ratio of the calculation result to the original offloaded data.

B. Model of Communication and Computation

In this subsection, we present model of communication between UV and CV, as well as model of local computing and edge computing.

The data transmission rate from V_n to S_k in the tth slot can be represented by $R_n^k(t)$, while $R_k^n(t)$ denotes the result feedback transmission rate from S_k to V_n. They are calculated by

$$R_n^k(t) = B_n^k \log_2\left(1 + \frac{P_n^k [d_n^k(t)]^{-\alpha}}{N_0 + I_n^k(t)}\right) \tag{6}$$

$$R_k^n(t) = B_k^n \log_2\left(1 + \frac{P_k^n [d_n^k(t)]^{-\alpha}}{N_0 + I_k^n(t)}\right) \tag{7}$$

where B_n^k and B_k^n refer to the bandwidth of the uplink and downlink, respectively. P_n^k and P_k^n are the transmission power of UV and CV, and $d_n^k(t)$ is the distance between V_n and S_k in slot t. α is the path loss factor, and N_0 is the power of the additive white Gaussian noise. $I_n^k(t)$ and $I_k^n(t)$ represent the signal interference to the uplink and downlink communication from other vehicles, respectively. We assume that each UV occupies an orthogonal channel to offload tasks, thus the interference $I_n^k(t)$ and $I_k^n(t)$ are negligible and equal to zero [12], [38]. Therefore, the amount of tasks leaving V_n's task offloading queue in slot t, i.e., $W_n^{off}(t)$, can be calculated by

$$W_n^{off}(t) = \min\{Q_n^{off}(t), \tau R_n^k(t)\} \tag{8}$$

Similarly, the amount of feedback result $W_{n,k}^{bac}(t)$ leaving S_k in slot t can be expressed as

$$W_{n,k}^{bac}(t) = \min\{G_{n,k}^{bac}(t), \tau R_k^n(t)\} \tag{9}$$

The amount of tasks processed by local computing at V_n in the tth slot can be calculated by

$$W_n^{loc}(t) = \min\left\{Q_n^{loc}(t), \frac{\tau f_n(t)}{\lambda_n}\right\} \tag{10}$$

where $f_n(t)$ denotes the assigned CPU frequency at V_n, and λ_n represents required computational density (cycles/bit). Note that λ_n is generally different, while its value is referred to the currently executed task type that is a constant. Thus, this does not affect the problem formulation and solution. The amount of tasks processed by edge computing at S_k in slot t can be calculated by

$$W_{n,k}^{com}(t) = \min\left\{G_{n,k}^{com}(t), \frac{\tau f_{n,k}(t)}{\lambda_n}\right\} \tag{11}$$

where $f_{n,k}(t)$ is the assigned computing resource for executing offloaded tasks within slot t.

C. Energy Consumption Model

The following four types of energy consumption are required to consider in VCEC: (1) Energy consumption for local computing at UV side $E_n^{loc}(t)$. (2) Energy consumption for task offloading $E_n^{off}(t)$. (3) Energy consumption for edge computing at CV side $E_{n,k}^{com}(t)$. (4) Energy consumption for result feedback $E_{n,k}^{bac}(t)$. They are denoted by formulas (12)-(15) [39].

$$E_n^{loc}(t) = \kappa [f_n(t)]^3 \min\left\{\frac{\lambda_n Q_n^{loc}(t)}{f_n(t)}, \tau\right\} \tag{12}$$

$$E_n^{off}(t) = P_n^k \min\left\{\frac{Q_n^{off}(t)}{R_n^k(t)}, \tau\right\} \tag{13}$$

$$E_{n,k}^{com}(t) = \kappa [f_{n,k}(t)]^3 \min \left\{ \frac{\lambda_n G_{n,k}^{com}(t)}{f_{n,k}(t)}, \tau \right\} \quad (14)$$

$$E_{n,k}^{bac}(t) = P_k^n \min \left\{ \frac{G_{n,k}^{bac}(t)}{R_k^n(t)}, \tau \right\} \quad (15)$$

where κ is the computation power parameter.

Therefore, the system energy consumption $E(t)$ in the tth slot can be expressed as

$$E(t) = \sum_{n=1}^{N} \left\{ E_n^{loc}(t) + \sum_{k=1}^{K} x_n^k(t) [E_n^{off}(t) + E_{n,k}^{com}(t) + E_{n,k}^{bac}(t)] \right\} \quad (16)$$

IV. Problem Formulation and Decomposition

In this section, we formulate the energy minimization problem bounded by the long-term queue delay constrains for VCEC system. Because of the coupling between long-term constraints and short-term strategy, the original issue is NP-hard. Thus, Lyapunov optimization is leveraged to decompose it into three short-term determinable sub-problems.

A. Queue Delay Constraints

In order to ensure the time efficiency of task offloading, we introduce constraints of long-term queue delay.

1) Queue Delay of Task Offloading: According to the Little's Law, the average data offloading queue delay is proportional to the average queue backlog divided by the average task arriving rate [40]. Denote $\tau_n^{Q,off}$ as the long-term average data offloading queue delay. Thus, we obtain

$$\tau_n^{Q,off} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \frac{Q_n^{off}(t)}{\bar{A}_n^{off}(t)} \le \tau_{n,max}^{Q,off} \quad (17)$$

where $\tau_{n,max}^{Q,off}$ denotes the upper bound for data offloading queue delay, and $\bar{A}_n^{off}(t)$ represents the average task arriving rate calculated by formula (18).

$$\bar{A}_n^{off}(t) = \frac{1}{t} \sum_{j=0}^{t-1} A_n^{off}(j) \quad (18)$$

2) Queue Delay of Task Computing: The queue delay of local computing at UV side is constrained by

$$\tau_n^{Q,loc} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \frac{Q_n^{loc}(t)}{\bar{A}_n^{loc}(t)} \le \tau_{n,max}^{Q,loc} \quad (19)$$

and the queue delay of the edge computing at CV side is constrained by

$$\tau_{n,k}^{G,com} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \frac{G_{n,k}^{com}(t)}{\bar{W}_n^{loc}(t)} \le \tau_{n,k,max}^{G,com} \quad (20)$$

where $\bar{A}_n^{loc}(t)$ and $\bar{W}_n^{loc}(t)$ can be calculated by formula (21) and (22), respectively.

$$\bar{A}_n^{loc}(t) = \frac{1}{t} \sum_{j=0}^{t-1} A_n^{loc}(j) \quad (21)$$

$$\bar{W}_n^{off}(t) = \frac{1}{t} \sum_{j=0}^{t-1} W_n^{off}(j) \quad (22)$$

3) Queue Delay of Result Feedback: The queue delay of result feedback is bounded by

$$\tau_{n,k}^{G,bac} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \frac{G_{n,k}^{bac}(t)}{\beta \bar{W}_{n,k}^{com}(t)} \le \tau_{n,k,max}^{G,bac} \quad (23)$$

where the average feedback task arriving rate of $G_{n,k}^{bac}(t)$ can be calculated as

$$\beta \bar{W}_{n,k}^{com}(t) = \frac{1}{t} \sum_{j=0}^{t-1} \beta W_{n,k}^{com}(j) \quad (24)$$

B. Problem Formulation

In this paper, the objective is to minimize the total power consumption in VCEC by jointly considering the task splitting and local computing resource allocation at UV side, task offloading strategy, and edge computing resource allocation at CV side. The joint optimization problem can be formulated as

$$\begin{aligned}
\mathbf{P1}: \ & \min_{\mathbf{A^{loc}(t)}, \mathbf{A^{off}(t)}, \mathbf{f(t)}, \mathbf{x(t)}} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} E(t) \\
\text{s.t.} \ & C_1: x_n^k(t) = \{0,1\}, \quad \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}, \\
& C_2: \sum_{k=1}^{K} x_n^k(t) < 1, \quad \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}, \\
& C_3: \sum_{n=1}^{N} x_n^k(t) < q_k^{max}, \quad \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}, \\
& C_4: \begin{cases} A_n(t) = A_n^{loc}(t) + A_n^{off}(t) \\ A_n^{loc}(t), A_n^{off}(t) \in \{0, A_0, 2A_0, \ldots\} \end{cases} \\
& C_5: f_n(t) \le f_{n,max}, \quad \forall V_n \in \mathcal{V}, \\
& C_6: \sum_{n=1}^{N} x_n^k(t) f_{n,k}(t) \le f_{k,max}, \quad \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}, \\
& C_7: (17), (19), (20), \text{and} (23), \quad \forall \mathrm{V_n} \in \mathcal{V}, \mathrm{S_k} \in \mathcal{S}.
\end{aligned} \quad (25)$$

where the task splitting at UV side is represented by $\mathbf{A^{loc}(t)} = \{A_n^{loc}(t), \forall V_n \in \mathcal{V}\}$ and $\mathbf{A^{off}(t)} = \{A_n^{off}(t), \forall V_n \in \mathcal{V}\}$. $\mathbf{f(t)} = \{f_n(t), f_{n,k}(t), \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}\}$ denotes the local and edge computing resource allocation, and the task offloading strategy is $\mathbf{x(t)} = \{x_n^k(t), \forall V_n \in \mathcal{V}, S_k \in \mathcal{S}\}$. C_1, C_2 and C_3 are constraints of task offloading factor, which means $x_n^k(t)$ is a binary variable, each UV can offload to one CV mostly, and at most q_k^{max} UVs can connect with one CV. C_4 represents the

task splitting. C_5 and C_6 are computing resource limitations for local and edge computing, respectively. C_7 denotes the long-term queue delay constraints.

C. Lyapunov Optimization

Note that the decision making is coupled with queue delay constraints, making **P1** nontrivial to solve. Thus, we apply Lyapunov optimization to convert the long-term issue to a series of short-term subproblems. We introduce the virtual queue concept [11] and transform C_7 into queue stability constraints. Four new virtual queues corresponding to the local computing queue, the task offloading queue, the edge computing queue, and the result feedback queue are respectively proposed as follows.

$$Z_n^{Q,loc}(t+1) = \max\left\{Z_n^{Q,loc}(t) + \frac{Q_n^{loc}(t+1)}{\bar{A}_n^{loc}(t+1)} - \tau_{n,max}^{Q,loc}, 0\right\} \tag{26}$$

$$Z_n^{Q,off}(t+1) = \max\left\{Z_n^{Q,off}(t) + \frac{Q_n^{off}(t+1)}{\bar{A}_n^{off}(t+1)} - \tau_{n,max}^{Q,off}, 0\right\} \tag{27}$$

$$Z_{n,k}^{G,com}(t+1) = \max\left\{Z_{n,k}^{G,com}(t) + \frac{G_{n,k}^{com}(t+1)}{\bar{W}_n^{off}(t+1)} - \tau_{n,k,max}^{G,com}, 0\right\} \tag{28}$$

$$Z_{n,k}^{G,bac}(t+1) = \max\left\{Z_{n,k}^{G,bac}(t) + \frac{G_{n,k}^{bac}(t+1)}{\beta\bar{W}_{n,k}^{com}(t+1)} - \tau_{n,k,max}^{G,bac}, 0\right\} \tag{29}$$

On the basis of literature [41], we can conclude that if the above four virtual queues are mean rate stable, then constraints C_7 can be automatically satisfied. To solve the original Problem **P1**, we further introduce the set $\Theta(t) = \{Q_n^{loc}(t), Q_n^{off}(t), G_{n,k}^{com}(t), G_{n,k}^{bac}(t), Z_n^{Q,loc}(t), Z_n^{Q,off}(t), Z_{n,k}^{G,com}(t), Z_{n,k}^{G,bac}(t)\}$. The Lyapunov function is defined as half of the sum of all queues' squares in $\Theta(t)$ [37], which is calculated by

$$\begin{aligned} L[\Theta(t)] = \frac{1}{2}\sum_{n=1}^{N}\{&Q_n^{loc}(t)^2 + Q_n^{off}(t)^2 + Z_n^{Q,loc}(t)^2 \\ &+ Z_n^{Q,off}(t)^2 + \sum_{k=1}^{K}[G_{n,k}^{com}(t)^2 + G_{n,k}^{bac}(t)^2 \\ &+ Z_{n,k}^{G,com}(t)^2 + Z_{n,k}^{G,bac}(t)^2]\} \end{aligned} \tag{30}$$

The definition of Lyapunov drift is the expectation of Lyapunov function changes in adjacent slots:

$$\Delta L[\Theta(t)] = \mathbb{E}\{L[\Theta(t+1)] - L[\Theta(t)] | \Theta(t)\} \tag{31}$$

The magnitude of $\Delta L[\Theta(t)]$ determines the change of the queue between two adjacent time slots and is necessary to assure the queue stability. To minimize system total energy consumption while ensuring queue stability, Lyapunov drift plus penalty is defined as Lyapunov drift plus the expectation of system energy consumption multiplied by a weight parameter V [8]:

$$\Delta_V L[\Theta(t)] = \Delta L[\Theta(t)] + V\mathbb{E}\{E(t)|\Theta(t)\} \tag{32}$$

where V is nonnegative and is used to make a tradeoff between "penalty minimization" and "queue stability". We can further relax the upper bound of Lyapunov drift plus penalty and derive the objective function of **P1** as the following theorem.

Theorem 1: Under every potential $\Theta(t)$ and $V \geqslant 0$, the drift plus penalty is "opportunistically" minimized in each time slot and the objective function of **P1** is given by

$$\begin{aligned} obj(t) = &\sum_{n=1}^{N} VE_n^{loc}(t) + Q_n^{loc}(t)(A_n^{loc}(t) - W_n^{loc}(t)) \\ &+ Q_n^{off}(t)A_n^{off}(t) \\ &+ (t+1)Z_n^{Q,loc}(t)\left(\frac{Q_n^{loc}(t) + A_n^{loc}(t) - W_n^{loc}(t)}{A_n^{loc}(t) + \sum_{j=0}^{t-1} A_n^{loc}(j)}\right) \\ &+ (t+1)Z_n^{Q,off}(t)\left(\frac{Q_n^{off}(t) + A_n^{off}(t)}{A_n^{off}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)}\right) \\ &+ \sum_{n=1}^{N}\sum_{k=1}^{K} x_n^k(t)\Bigg[VE_n^{off}(t) + VE_{n,k}^{bac}(t) + VE_{n,k}^{com}(t) \\ &- Q_n^{off}(t)W_n^{off}(t) - \frac{(t+1)Z_n^{Q,off}(t)W_n^{off}(t)}{A_n^{off}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)} \\ &+ G_{n,k}^{com}(t)W_n^{off}(t) - G_{n,k}^{bac}(t)W_{n,k}^{bac}(t) \\ &+ (t+1)Z_{n,k}^{G,com}(t)\frac{G_{n,k}^{com}(t) + W_n^{o}ff(t)}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)} \\ &- \frac{(t+1)Z_{n,k}^{G,com}(t)W_{n,k}^{com}(t)}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)} \\ &+ \frac{(t+1)Z_{n,k}^{G,bac}(t)\beta W_{n,k}^{com}(t)}{\beta[W_{n,k}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)] + G_{n,k}^{bac}(t)} \\ &+ (t+1)Z_{n,k}^{G,bac}(t)\frac{G_{n,k}^{bac}(t) - W_{n,k}^{bac}(t)}{\beta[W_{n,k,max}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)]}\Bigg] \\ &+ G_{n,k}^{bac}(t)\beta W_{n,k}^{com}(t) - G_{n,k}^{com}(t)W_{n,k}^{com}(t) \end{aligned} \tag{33}$$

Proof: See Appendix A. ■

D. Problem Decomposition

According to Lyapunov optimization, the objective function of **P1** is able to be transformed into three sub-problems by distinguishing the optimization variables. The first part that contains variables of task splitting and local computing resource

allocation can be regarded as subproblem of **SP1**, while **SP2** contains task offloading variable, and **SP3** contains edge computing resource allocation variable.

*1) UV Side Task Splitting and Resource Allocation (**SP1**):* Firstly, we need to determine the proportion of local computing and task offloading. Then the allocated computing resource at UV side should be decided in slot t. **SP1** is denoted by

$$\begin{aligned}
\mathbf{SP1}: & \min_{A_n^{loc}(t), A_n^{off}(t), f_n(t)} \Phi(A_n^{loc}(t), f_n(t)) \\
= & \; V\kappa[f_n(t)]^3 \min\left\{\frac{\lambda_n Q_n^{loc}(t)}{f_n(t)}, \tau\right\} \\
& + Q_n^{loc}(t)(A_n^{loc}(t) - \frac{\tau f_n(t)}{\lambda_n}) + Q_n^{off}(t)(A_n(t) - A_n^{loc}(t)) \\
& + (t+1)Z_n^{Q,loc}(t)\left(\frac{Q_n^{loc}(t) + A_n^{loc}(t) - \frac{\tau f_n(t)}{\lambda_n}}{A_n^{loc}(t) + \sum_{j=0}^{t-1} A_n^{loc}(j)}\right) \\
& + (t+1)Z_n^{Q,off}(t)\left(\frac{Q_n^{off}(t) + A_n(t) - A_n^{loc}(t)}{A_n(t) - A_n^{loc}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)}\right) \\
& \text{s.t.} \quad C_4, C_5 \\
& C_8: \frac{\tau f_n(t)}{\lambda_n} \le Q_n^{loc}(t)
\end{aligned} \tag{34}$$

*2) Task Offloading Decision (**SP2**):* In **SP2**, each UV selects a CV to send task offloading request under information uncertainty, and CV will resolve the task offloading conflicts according to its preference over the requested UVs in slot t. **SP2** is denoted by

$$\begin{aligned}
\mathbf{SP2}: & \min_{x_n^k(t)} \Gamma(x_n^k(t)) = \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_n^k(t) \\
= & \sum_{n=1}^{N}\sum_{k=1}^{K} x_n^k(t)\left[VP_n^k \min\left\{\frac{Q_n^{off}(t)}{R_n^k(t)}, \tau\right\}\right. \\
& - Q_n^{off}(t)W_n^{off}(t) \\
& \left. + VP_k^n \min\left\{\frac{G_{n,k}^{bac}(t)}{R_k^n(t)}, \tau\right\} + G_{n,k}^{com}(t)W_n^{off}(t)\right. \\
& - \frac{(t+1)Z_n^{Q,off}(t)W_n^{off}(t)}{A_n^{off}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)} - G_{n,k}^{bac}(t)W_{n,k}^{bac}(t) \\
& + (t+1)Z_{n,k}^{G,com}(t)\left(\frac{G_{n,k}^{com}(t) + W_n^{off}(t)}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)}\right) \\
& \left. + (t+1)Z_{n,k}^{G,bac}(t)\left(\frac{G_{n,k}^{bac}(t) - W_{n,k}^{bac}(t)}{\beta[W_{n,k,max}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)]}\right)\right] \\
& \text{s.t.} \quad C_1, C_2, C_3
\end{aligned} \tag{35}$$

*3) CV Side Resource Allocation (**SP3**):* CVs determine the computing resource allocation to process tasks from UVs in slot t. **SP3** is denoted by

$$\begin{aligned}
\mathbf{SP3}: & \max_{f_{n,k}(t)} \Omega(f_{n,k}(t)) \\
= & \sum_{n=1}^{N} V\kappa[f_{n,k}(t)]^3 \min\left\{\frac{\lambda_n G_{n,k}^{com}(t)}{f_{n,k}(t)}, \tau\right\} - G_{n,k}^{com}(t)\frac{\tau f_{n,k}(t)}{\lambda_n} \\
& + G_{n,k}^{bac}(t)\frac{\tau f_{n,k}(t)}{\lambda_n} - \frac{(t+1)Z_{n,k}^{G,com}(t)\frac{\tau}{\tau} f_{n,k}(t)\lambda_n}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)} \\
& + \frac{(t+1)Z_{n,k}^{G,bac}(t)\frac{\tau f_{n,k}(t)}{\lambda_n}}{\beta[W_{n,k}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)]} \\
& \text{s.t.} \quad C_6 \\
& C_9: \frac{\tau f_{n,k}(t)}{\lambda_n} \le G_{n,k}^{com}(t)
\end{aligned} \tag{36}$$

V. Energy-Minimization Task Offloading and Resource Assignment for VCEC System With Queue-Awareness Occurrence-Awareness and Conflict-Awareness

In this section, we provide algorithms to solve the above sub-problems. **SP1** is a convex optimization issue which is able to be handled by Lagrange multiplier method. For **SP2**, to cope with the optimal task offloading decision under information uncertainty, a learning-based UCB approach with queue-awareness and occurrence-awareness is developed. We also leverage price-matching to cope with matching conflict for **SP2**. Last but not least, we give a low-complexity greedy algorithm to optimize the CV side resource assignment for **SP3**. Finally, the optimization result of **P1** could be achieved.

A. Task Splitting and Resource Assignment At UV Side for SP1

Since $\Phi(A_n^{loc}(t), f_n(t))$ of **SP1** is convex, Lagrange multiplier method can be utilized [9]. The Lagrange function is represented by

$$\begin{aligned}
\mathcal{L}_n(A_n^{loc}(t), f_n(t), \theta_n, \xi_n) = & \; \Phi(A_n^{loc}(t), f_n(t)) \\
& + \theta_n\left[\frac{\tau f_n(t)}{\lambda_n} - Q_n^{loc}(t)\right] + \xi_n[f_n(t) - f_{n,max}]
\end{aligned} \tag{37}$$

where θ_n and ξ_n denote Lagrange multipliers. In light of Karush-Kuhn-Tucker (KKT) conditions, the optimal result could be achieved by setting the partial derivatives of $\mathcal{L}_n$ with regard to $A_n^{loc}(y)$, $f_n(t)$, θ_n and ξ_n to zero:

$$\begin{cases}
\frac{\partial \mathcal{L}_n(A_n^{loc}(t), f_n(t), \theta_n, \xi_n)}{\partial A_n^{loc}(t)} = 0 \\
\frac{\partial \mathcal{L}_n(A_n^{loc}(t), f_n(t), \theta_n, \xi_n)}{\partial f_n(t)} = 0 \\
\frac{\partial \mathcal{L}_n(A_n^{loc}(t), f_n(t), \theta_n, \xi_n)}{\partial \theta_n} = 0 \\
\frac{\partial \mathcal{L}_n(A_n^{loc}(t), f_n(t), \theta_n, \xi_n)}{\partial \xi_n} = 0
\end{cases} \tag{38}$$

Note that, the first equation of (38) is a one-variable quartic equation that could be handled by Ferrari's method [41], while the second equation is a quadratic equation whose result can be obtained by (39), shown at the bottom of the next page, where μ is the loop index of Lagrange multipliers updated using gradient

Algorithm 1: Task Splitting and Local Resource Allocation algorithm (**TSLRA**).

Input: $T, A_n(t), \lambda_n, V, \kappa, \tau^{Q,loc}_{n,max}, \tau^{Q,off}_{n,max}, f_{n,max}, a^k_n(t)$.
Output: $A^{loc*}_n(t), f^*_n(t)$.
Initialize: $Q^{loc}_n(1) = 0, Q^{off}_n(1) = 0, Z^{Q,loc}_n(1) = 0, Z^{Q,off}_n(1) = 0$.
while $t = 1 \sim T$ **do**
  while $V_n \in \mathcal{V}$ **do**
    $\mu = 0$
    while $|A^{loc}_n(t, \mu+1) - A^{loc}_y(t, \mu)| \geq \Delta || |f_n(t, \mu+1) - f_n(t, \mu)| \geq \Delta$ **do**
      obtain $A^{loc}_n(t, \mu+1)$ by Ferrari's method
      obtain $f_n(t, \mu+1)$ using (46)
      update $\theta_n(t, \mu)$ and $\xi_n(t, \mu)$ using (47) and (48)
      $\mu = \mu + 1$
    end while
    $A^{loc*}_n(t) = A^{loc}_n(t, \mu_{max})$ and $f^*_n(t) = f_n(t, \mu_{max})$
    update $Q^{loc}_n(t)$, $Q^{off}_n(t)$, $Z^{Q,loc}_n(t)$ and $Z^{Q,off}_n(t)$ using (2), (3), (26), and (27)
    if $A^{loc*}_n(t) == A_n(t)$ **then**
      $a^k_n(t) = 0, k = 1, \ldots, K$
    else
      maintain the value of $a^k_n(t)$
    end if
  end while
end while

method shown by (40) and (41).

$$\theta_n(t, \mu+1) = \max\{\theta_n(t, \mu) + \zeta_{\theta_n}(f_n(t, \mu) - f_{n,max}), 0\} \tag{40}$$

$$\xi_n(t, \mu+1) = \max\left\{\xi_n(t, \mu) + \zeta_{\xi_n}\left(\frac{\tau f_n(t, \mu)}{\lambda_n} - Q^{loc}_n(t)\right), 0\right\} \tag{41}$$

where ζ_{θ_n} and ζ_{ξ_n} are the updating step sizes, which should be chosen charily.

Based on the above analysis, we propose Algorithm 1 (**TSLRA**) to solve **SP1**.

B. Learning-Matching Based Task Offloading Optimization Under Information Uncertainty for *SP2*

Firstly, we utilize many-to-one matching to obtain the optimal task offloading decision for N UVs in **SP2** with global channel state information, where each UV can offload tasks to one CV mostly while one CV can serve at most q^{max}_k UVs.

Definition 1: A many-to-one matching Π satisfies $\Pi(V_n) = S_k || V_n \in \Pi(S_k)$ in set $\mathcal{V} \cup \mathcal{S}$, which indicates that V_n selects S_k as task offloading target, i.e., $x^k_n(t) = 1$. Defining $|\Pi(*)|$ as the cardinality, then $|\Pi(V_n)| = 1$ and $|\Pi(S_k)| \leq q^{max}_k$ correspond to constraints C_2 and C_3. If V_n does not match with any CVs, then $\Pi(V_n) = V_n$ and it can only offload tasks to RSU. Note that full local computing-UVs have been removed by Algorithm 1 and they are not contained in the matching.

However, practically speaking, the available edge computing resources of CV candidates are commonly not known to UVs, and CSI and CV candidates' availability vary over time because of vehicles' mobility. To cope with the dilemma of information uncertainty and asymmetry [33], the UCB-based learning is combined with matching. According to principle of UCB, expected utility of historical observations and the uncertainty degree of these observations are simultaneously considered to obtain utility estimation. Therefore, preference of V_n to S_k under Π in the tth slot can be estimated as follows.

$$U^k_n(t) = \frac{1}{\tilde{\gamma}^k_n(t-1)} + \sqrt{\frac{\eta \ln t}{\rho^k_n(t-1)}} \tag{42}$$

where $\tilde{\gamma}^k_n(t-1)$ represents the average utility estimation before slot t, and $\rho^k_n(t-1)$ is the total number of times that V_n selects S_k for task offloading before slot t, i.e., $\rho^k_n(t-1) = \sum_{i=1}^{t-1} x^k_n(i)$. The first item of the formula is the average reward of V_n when selecting S_k, which represents the exploitation. The second item is the confidence bound, which denotes the exploration. η is a positive constant that tradeoffs exploitation and exploration. The larger η is, the more exploration it trends. The updating of $\tilde{\gamma}^k_n(t)$ and $\rho^k_n(t)$ are shown by (43) and (44).

$$\tilde{\gamma}^k_n(t) = \frac{\tilde{\gamma}^k_n(t-1)\rho^k_n(t-1) + \gamma^k_n(t)x^k_n(t)}{\rho^k_n(t-1) + x^k_n(t)} \tag{43}$$

$$\rho^k_n(t) = \rho^k_n(t-1) + x^k_n(t) \tag{44}$$

where $\gamma^k_n(t)$ is the utility calculated by V_n when it offloads tasks to S_k in the tth slot.

Define the set of CV candidates for V_n in the tth slot as $\mathcal{S}^n(t) = \{\mathcal{S} | a^k_n(t) \neq 0\}$. V_n will choose the most preferred candidate belonging to $\mathcal{S}^n(t)$ for task offloading by estimating utility of each CV. Then, the candidate with the largest utility is selected to send the task offloading request, i.e., the optimal candidate $\Pi(V_n)^*$ expressed as

$$\Pi(V_n)^* = \arg \max_{\forall S_k \in \mathcal{S}^n} U^k_n(t) \tag{45}$$

For each S_k that receives offloading requests, it firstly checks whether the number of requests exceeds its service capacity q^{max}_k. If it is no more than q^{max}_k, then S_k is matched with all

$$f_n(t, \mu+1) = \sqrt{\frac{1}{3V\kappa \min\{\frac{\lambda_n Q^{loc}_n(t)}{f_n(t,\mu)}, \tau\}} \left\{\frac{\tau Q^{loc}_n(t)}{\lambda_n} + \frac{\tau(t+1)Z^{Q,loc}_n(t)}{\lambda_n[A^{loc}_n(t) + \sum_{j=0}^{t-1} A^{loc}_n(j)]} - \theta_n(t, \mu)\frac{\tau}{\lambda_n} - \xi_n(t, \mu)\right\}} \tag{39}$$

those UVs. Otherwise, matching conflict occurs and those UVs are put into a set $\Lambda(t)$. To cope with the matching conflict issue, we introduce the price-matching and the utility function can be modified to

$$\widehat{U}_n^k(t) = U_n^k(t) - B_k \tag{46}$$

where B_k is the matching cost of S_k, whose initial value is set zero. However, when matching conflict occurs, B_k of each $S_k \in \Lambda(t)$, is increased by a fixed step size ΔB_k according to the following formula.

$$B_k = B_k + \Delta B_k \tag{47}$$

Each UV requesting to connect with $S_k \in \Lambda(t)$ needs to recalculate its utility according to formula (46), and reselects the optimal candidate. The above price-matching process continues until requestors' total number is equal to q_k^{max}, then S_k is removed from $\Lambda(t)$. When there is no matching conflict in the system, it means that a stable matching result has been achieved. For the UV that does not successfully match with any CV, let $\Pi(V_n) = V_n$.

Moreover, a CV may occur or vanish when it enters or leaves a UV's communication range. Thus, we take CVs' volatility into consideration and combine the occurrence-awareness to our solution. The concept of occurrence time is proposed, and we let $t_{n,k}^{fir}$ represent the slot when S_k appears in the communication range of V_n for the first time, i.e., $\{t_{n,k}^{fir} = t | a_n^k(t) = 1, a_n^k(l) = 0, l = 1, \ldots, t-1\}$. Therefore, the estimated utility function is further modified to

$$\widetilde{U}_n^k(t) = \begin{cases} \frac{1}{\widetilde{\gamma}_n^k(t-1)} + \sqrt{\frac{\eta \ln(t - t_{n,k}^{fir})}{\rho_n^k(t-1)}} - B_k, & t > t_{n,k}^{fir} \\ +\infty, & t = t_{n,k}^{fir} \end{cases} \tag{48}$$

The utility of S_k is infinite large when $t = t_{n,k}^{fir}$, which indicates that V_n can match S_k at least once. The earlier S_k appears, the more frequently it will be selected. According to the Chernoff-Hoeffding inequality, the empirical performance of those CVs which appear earlier can be estimated more precisely due to larger historical observations.

Based on the above analysis, we propose **QOCA-UCB** of Algorithm 2 to solve the subproblem of optimal task offloading with queue-awareness, occurrence-awareness, and conflict-awareness.

Algorithm 2: Queue-Occurrence-Conflict Awareness-UCB based Task offloading under Information Uncertainty (**QOCA-UCB**).

Input: $\eta, \Delta B_k, q_k^{max}, a_n^k(t), Q_n^{off}(t), G_{n,k}^{com}(t), G_{n,k}^{bac}(t), Z_n^{Q,off}(t), Z_{n,k}^{G,com}(t), Z_{n,k}^{G,bac}(t), A_n^{off}(t)$.
Output: A stable Matching Π and task offloading strategy $\mathbf{X(t)}$.
Initialize: Each $V_n \in \mathcal{V}$ establishes $\mathcal{S}^n(t)$ according to $a_n^k(t)$ and randomly matches with $S_k \in \mathcal{S}^n(t), B_k = 0$.
Phase 1: Matching
while $V_n \in \mathcal{V}$ **do**
  obtain $t_{n,k}^{fir}$ by $\{t_{n,k}^{fir} = t | a_n^k(t) = 1, a_n^k(l) = 0, l = 1, \ldots, t-1\}$
  calculate $\widetilde{U}_n^k(t)$ for all $S_k \in \mathcal{S}^n$ using (55)
  select the optimal candidate $\Pi(V_n)^*$ and send task offloading request
end while
Phase 2: Conflict elimination
$\Lambda(t) = \varnothing$
repeat
while $S_k \in \mathcal{S}$ **do**
  if the total number of received requests is no more than q_k^{max} **then**
    $x_n^k(t) = 1, \Pi(V_n) = S_k$
  else
    $S_k \Rightarrow \Lambda(t)$
  end if
end while
while $S_k \in \Lambda(t)$ **do**
  repeat
  update B_k using (54)
  for each UV sending request to S_k, recalculate $\widetilde{U}_n^k(t)$ using (55)
  until the total number of received requests is equal to q_k^{max}
  update $x_n^k(t)$ and Π
end while
until there is no matching conflict
for V_n that does not successfully match with any CV, set $\Pi(V_n) = V_n$
execute task offloading based on Π
Phase 3: Learning
observe $\gamma_n^k(t)$ using (42)
update $\widetilde{\gamma}_n^k(t)$ and $\rho_n^k(t)$ using (50) and (51)
update $Q_n^{off}(t), G_{n,k}^{com}(t), G_{n,k}^{bac}(t), Z_n^{Q,off}(t), Z_{n,k}^{G,com}(t)$, and $Z_{n,k}^{G,bac}(t)$

C. Edge Resource Allocation At CV Side for *SP3*

For **SP3**, we propose a greedy-based edge resource allocation algorithm. Given task offloading decision $\mathbf{X(t)}$, each S_k can obtain its associated UV set $\mathcal{V}^k$ in the tth slot, i.e., $\mathcal{V}^k = \{V_n | x_n^k(t) = 1, n = 1, \ldots, N\}$. Let $F_k(t)$ denote the available computing resource of S_k in slot t, whose initial value is equal to $f_{k,max}$ and decreases as resource allocation proceeds. For $V_n \in \mathcal{V}^k$, the maximum computing resource $\sigma_{n,k}(t)$ that can be allocated is calculated by

$$\sigma_{n,k}(t) = \min\left\{F_k(t), \frac{\lambda_n G_{n,k}^{com}(t)}{\tau}\right\} \tag{49}$$

Algorithm 3: Greedy Based Edge Computing Resource Allocation at CV Side (**GECRA**).

Input: $G_{n,k}^{com}(t), G_{n,k}^{bac}(t), Z_{n,k}^{G,com}(t), Z_{n,k}^{G,bac}(t), f_{k,max}, \lambda_n$.
Output: $f_{n,k}^*(t)$.
while $S_k \in \mathcal{S}$ **do**
  $\mathcal{V}^k(t) = \{\mathcal{V} | x_n^k(t) = 1, n = 1, \ldots, N\}$ and $F_k(t) = f_{k,max}$
  while $\mathcal{V}^k(t) \neq \varnothing$ &&$F_k(t) > 0$ **do**
    $\sigma_{n,k}(t) = \min\{F_k(t), \frac{\lambda_n G_{n,k}^{com}(t)}{\tau}\}$
    calculate $\Omega_{n,k}(\sigma_{n,k}(t))$ using (57)
    $V_n^* = \arg[\max \Omega_{y,k}(\sigma_{n,k}(t))]$
    $f_{n,k}^*(t) = \sigma_{n,k}(t)$
    $\mathcal{V}^k(t) = \mathcal{V}^k(t) \setminus V_n^*$
    $F_k(t) = F_k(t) - f_{n,k}^*(t)$
  end while
end while

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Total time period T	1000
Duration of a time slots τ	100 ms
CV service capacity q_k^{max}	2
Pathloss factor α	3.4
Transmission power P_n^k, P_k^n	23 dBm
Bandwidth B_n^k, B_k^n	10 MHz
Power of the additive white Gaussian noise N_0	-144 dBm
Required computational density λ_n	1000 cycles/bit
Queuing delay constraints $\tau_{n,max}^{Q,loc}, \tau_{n,max}^{Q,off}, \tau_{n,k,max}^{G,com}, \tau_{n,k,max}^{G,bac}$	0.4 s, 0.15 s, 0.2 s, 0.15 s
Computation power parameter κ	5×10^{-26} J/Hz^3/s
Ratio of result to original task β	0.167
Tradeoff between exploitation and exploration η	2
Weight parameter V	10^6

Substituting $\sigma_{n,k}(t)$ into the objective function of **SP3**, we can obtain

$$\begin{aligned}&\Omega_{n,k}(\sigma_{n,k}(t))\\&= V\kappa[\sigma_{n,k}(t)]^3 \min\left\{\frac{\lambda_n G_{n,k}^{com}(t)}{\sigma_{n,k}(t)}, \tau\right\} - G_{n,k}^{com}(t)\frac{\tau\sigma_{n,k}(t)}{\lambda_n}\\&+ G_{n,k}^{bac}(t)\frac{\tau\sigma_{n,k}(t)}{\lambda_n} - \frac{(t+1)Z_{n,k}^{G,com}(t)\frac{\tau\sigma_{n,k}(t)}{\lambda_n}}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)}\\&+ \frac{(t+1)Z_{n,k}^{G,bac}(t)\frac{\tau\sigma_{n,k}(t)}{\lambda_n}}{\beta[\frac{\tau\sigma_{n,k}(t)}{\lambda_n} + \sum_{j=0}^{t-1} W_n^{com}(j)]}\end{aligned} \tag{50}$$

Based on principle of the greedy algorithm, S_k preferentially allocates computing resource to the UV with the largest $\Omega_{n,k}(\sigma_{n,k}(t))$. The process will not end until all UVs' offloaded queues have been allocated computing resource, or the available resource $F_k(t)$ becomes zero. More details can be seen in **GECRA** of Algorithm 3.

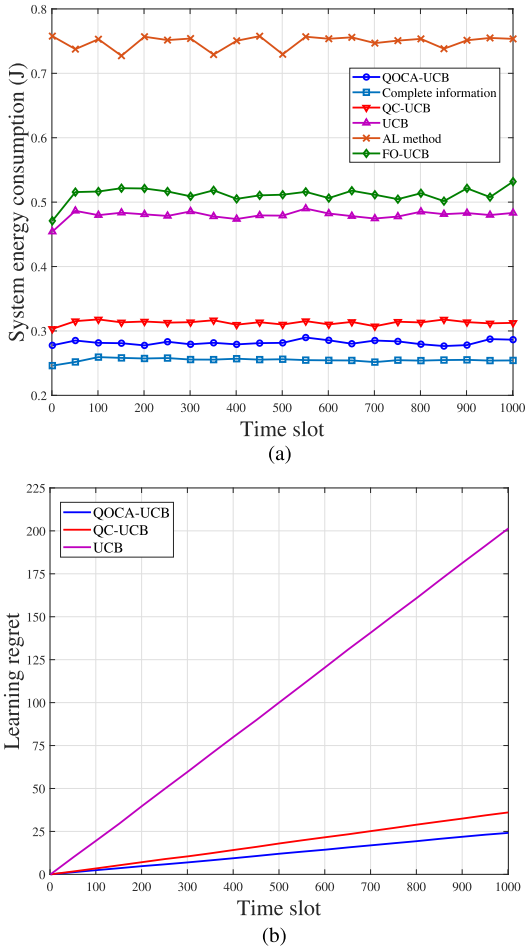


Fig. 2. Energy consumption and learning regret under scenario 1. (a) System energy consumption versus time slot. (b) Learning regret.

D. Properties Analyses

In this subsection, we theoretically analyze the optimality of Algorithms 1 and 3, discuss the awareness and learning regret of Algorithm 2, and explain the complexity of all three algorithms.

1) Optimality: **SP1** is a strict convex optimization problem, thus KKT condition is utilized to calculate the optimal result for each iteration. The task splitting and local computing resource allocation at UV side can be optimized by **TSLRA**, and the resource allocation at CV side can be optimized by **GECRA** based on the greedy algorithm. Since we always preferentially allocate resource to UV with the maximum objective value and obtain local optimal solution, hence, the near optimal result of **SP3** can be achieved while avoiding the complexity of exhausting global optimal method. Because of the finite number of UVs and CVs, both Algorithms 1 and 3 can converge.

2) Awareness: Firstly of all, our approach is queue-aware since it can alter task offloading decisions according to the dynamic real-time queue information. For example, when queue backlog grows too large, the queue delay may exceed its bound which gives feedback to the utility function. This will impact UVs' preference and task offloading strategy and motivate them to choose better CVs for task offloading. Secondly, our method is conflict-awareness because of the pricing-matching strategy

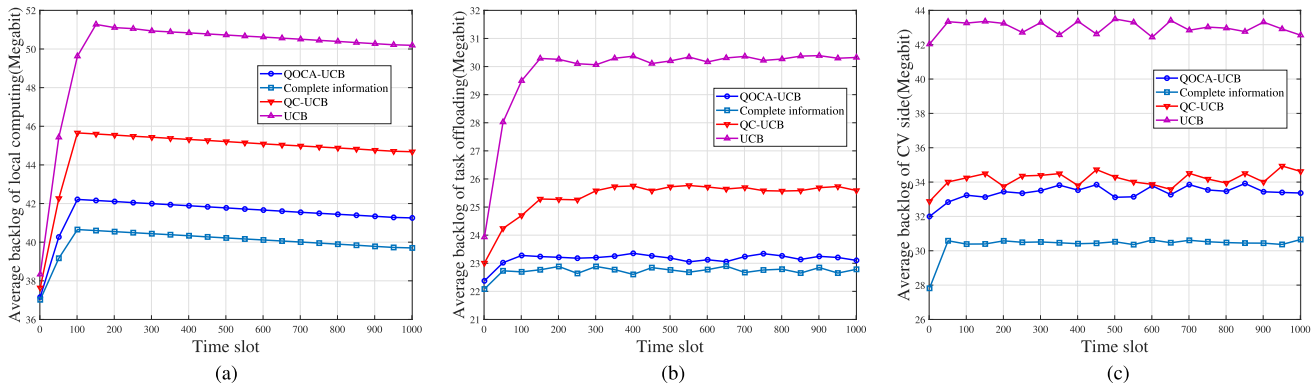


Fig. 3. Average backlog over time slot under scenario 1. (a) Local computing $Q_n^{loc}(t)$. (b) Task offloading $Q_n^{off}(t)$. (c) Server side $G_{n,k}^{com}(t)$.

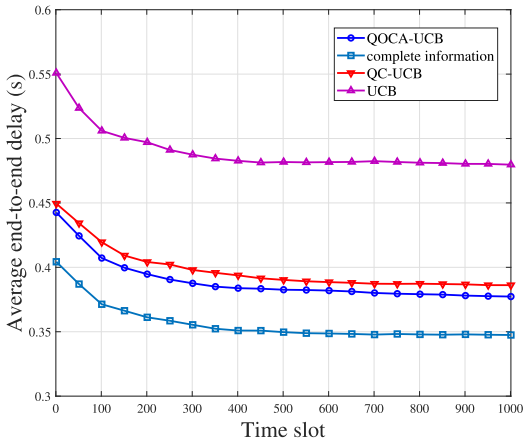
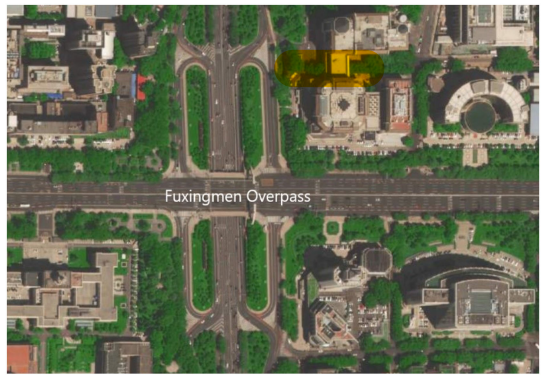
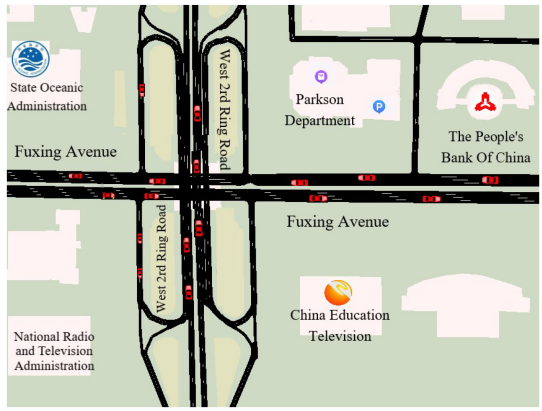


Fig. 4. Average end-to-end delay versus time slot under scenario 1.



(a)



(b)

Fig. 5. Real-world road. (a) Fuxingmen overpass topology. (b) Snapshot of SUMO simulation scenario.

for conflict elimination as illustrated in previous subsection. Last but not least, according to $\widetilde{U}_n^k(t)$ in (55),our approach is occurrence-awareness due to that the performance of those CVs which appear earlier can be estimated more precisely owing to larger historical observations.

3) Learning Regret: In this paper, learning regret is defined as the expectation of the difference between the optimal $\Gamma^*(x_n^k(t))$ obtained under global complete information and $\Gamma(x_n^k(t))$ achieved under information uncertainty given by

$$LR = \mathbb{E}\left\{\sum_{t=1}^{T}\Gamma(x_n^k(t)) - \Gamma^*(x_n^k(t))\right\} = \mathbb{E}\left\{\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{k=1}^{K} x_n^k(t)\gamma_n^k(t) - x_n^{k^*}(t)\gamma_n^{k^*}(t)\right\} \quad (51)$$

where $k^* = \Pi^*(V_n)$ refers to the global optimal result of matching. We further define $\Delta\gamma_n^k$ as follows

$$\Delta\gamma_n^k = \mathbb{E}\{\gamma_n^k(t) - \gamma_n^{k^*}(t)\} \quad (52)$$

Then, we are able to prove that **QOCA-UCB** can obtain a guaranteed task offloading result under information uncertainty, i.e., achieving a reliable performance and a bounded difference from the optimal performance without global complete information.

Theorem 2: The upper confidence bound of learning regret for Algorithm 2 can be expressed as

$$LR \le \sum_{t=1}^{T}\sum_{k=1,k\neq k^*}^{K}\left[2\eta^2\frac{\ln(T - t_{n,k}^{fir})}{\Delta\gamma_n^k} + \left(1 + \frac{\pi^2}{3}\right)\Delta\gamma_n^k\right] \quad (53)$$

Proof: See Appendix B. ■

4) Complexity: Algorithm 1 jointly optimizes task splitting and computing resource assignment within T slots and N UVs. Its complexity is $o(TN\mu_{max})$, where μ_{max} is the number of

iterations required. The complxity of Algorithm 2 consists of three phases. In phase 1, N UVs calculate the utility of K CVs and select the optimal CV by sorting all candidates in a descending order with complexity of $o(NK + NK\log_2 K)$. Assuming that matching conflicts are resolved within ϵ iterations in phase 2, the complexity of conflict elimination is $o(N\epsilon)$ when $N \geq K$. In phase 3, each UV observes its task offloading performance and updates the historicial experience with complexity of $o(N)$. Therefore, the complexity of Algorithm 2 in T time slots is $o(NKT + NKT\log_2 K + N\epsilon T + NT)$. Algorithm 3 optimizes the allocation of K CVs' computing resources for T slots, with a complexity of $o(TKL_3)$, where L_3 is iteration number for CV to find its preferential users.

VI. Simulations Results

In this section, we evaluate the proposed algorithms by carrying out extensive simulations. We consider two scenarios where vehicles move dynamically. Scenario 1: scenario without volatility, i.e., vehicles' speeds are almost the same and distance between two vehicles can be assumed unchanged. Scenario 2: real-world road topology with volatility, i.e., vehicles' speeds are different and vary over time, resulting in constantly variation of neighbor relationship. These two scenarios are utilized to evaluate the queue-awareness, occurrence-awareness, and conflict-awareness characteristics. The communication range of each UV is 200m [17]. The number of arriving tasks $A_n(t)$ follows uniform distribution within $[16.8, 25.2] \times 10^6$ bits, and the maximum available computing resources $f_{n,max}$, $f_{k,max}$ are uniformly distributed within [5,18] and [160,240] GHz, respectively. Other parameters are presented in Table I.

Our matching-learning approach is compared with five benchmark methods.

1) *UCB method:* Matching conflict is ignored and $U_n^k(t)$ is the utility function for machine learning in Algorithm 2 without conflict-awareness and occurrence-awareness. For the UV that fails to connect with a CV, its tasks are offloaded to a RSU, in which case the channel condition is generally worse than that of offloading tasks to a nearby CV.
2) *QC-UCB method:* $\widehat{U}_n^k(t)$ is the utility function for machine learning in Algorithm 2, which does not consider the volatility of vehicles. This method has inferior performance for finding a better CV through "exploration" owing to that there is no occurrence-awareness.
3) *Optimal method with global information:* UV selects CV by the brute-force search. The utility function $\Gamma(x_n^k(t))$ can be directly calculated under global information of CV, and the CV with the best utility value is selected to perform task offloading. The optimization of local and edge computing resource allocation are still carried out in accordance with Algorithms 1 and 3. Note that, it is only feasible for small-scale problems.
4) *All local computing method (AL method):* All tasks of each UV are processed at local side, and local resource allocation is optimized by our method.

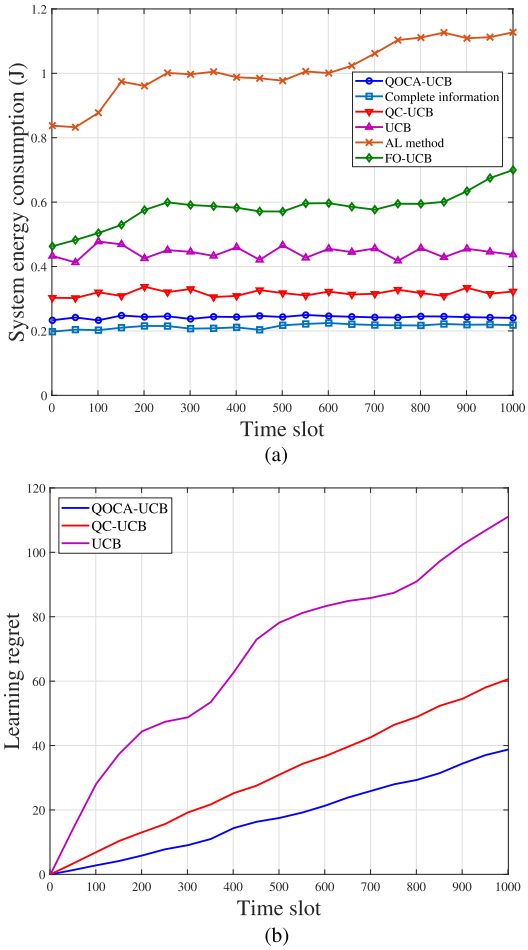


Fig. 6. Energy consumption and learning regret in scenario 2. (a) System energy consumption versus time slot. (b) Learning regret.

5) *Full offloading-UCB method (FO-UCB):* All tasks of each UV are offloaded to CV, and task offloading strategy and resource allocation is optimized by our method.

We optimize the joint task offloading and resource allocation every J slots. When $t = J+1, 2J+1, 3J+1, \ldots$ Algorithms 1, 2, and 3 are executed, but for the rest of the time the task offloading strategy maintains constant, i.e., $x_n^k(t+1) = x_n^k(t)$. In our simulation, $J = 5$ and we find that the actual calculation time is much less than $J \times \tau$. Therefore, the calculation cost of our proposed method is acceptable and is feasible in practice.

A. Scenario 1: Scenario Without Volatility

In this scenario, we consider a 30 km road containing 10 UVs and 5 CVs. The initial position of the UV is 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000 m, and the CV is 100, 300, 500, 700, 900 m. All vehicles are moving at the same speed of 20 m/s, so they can keep constant distance.

Fig. 2(a) shows the system energy consumption versus time slot. Results of power consumption suggest that there is almost no difference among QC-UCB, optimal and QOCA-UCB methods, while QOCA-UCB outperforms UCB by 72.7%. The reason is that queue-awareness, occurrence-awareness and conflict-awareness are all taken into considered in QOCA-UCB. The

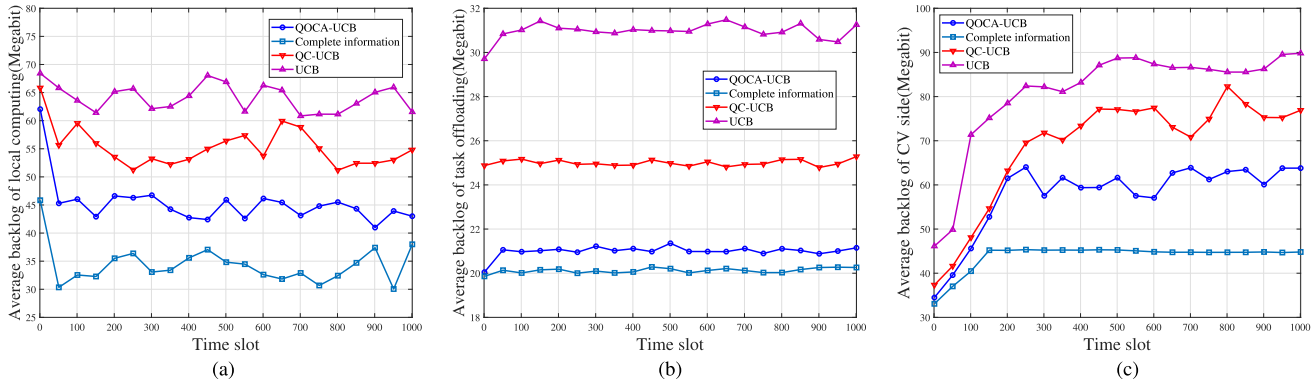


Fig. 7. Average backlog over time slot under scenario 2. (a) Local computing $Q_n^{loc}(t)$. (b) Task offloading $Q_n^{off}(t)$. (c) CV side $G_{n.k}^{com}(t)$.

energy consumption of AL method and OF-UCB are quite high due to lack of task splitting. Thus, the computational resource of UV or CV are not fully used. It can be seen in Fig. 2(b) that QOCA-UCB has superior performance in terms of both learning regret and convergence speed compared with UCB and QC-UCB.

Fig. 3(a), (b), and (c) show the backlogs of queue $Q_n^{loc}(t)$, $Q_n^{off}(t)$ and $G_{n,k}^{com}(t)$ over time. We can see the task backlog of the QOCA-UCB approach is nearly the same as that of the optimal method, but is significantly less than that of the QC-UCB and UCB methods. The reason is that QOCA-UCB is easier to select better offloading CVs and more arriving tasks are added to task offloading queues due to queue-awareness, thus UV's local queue backlog is much smaller. For queues $Q_n^{off}(t)$ and $G_{n,k}^{com}(t)$, the UCB and QC-UCB methods have large backlogs due to no conflict awareness or occurrence awareness.

Fig. 4 shows the average end-to-end delay versus time slot, which is defined as sum of task offloading delay, edge computing delay, and result feedback delay. QOCA-UCB also has superior performance. For example, when $t = 800$, QOCA-UCB can reduce the end-to-end delay by 27.1% compared with UCB.

B. Scenario 2: Real-World Road Topology With Volatility

We use SUMO to investigate the representative real-world scenario of Fuxingmen overpass in Beijing, China. Fig. 5(a) shows its road topology located at 39.91°N, 116.36°E. the NETCONVERT tool is used to acquire XML file containing road information from OpenStreetMap, and the RANDOMTRIPS tool is utilized to generate virtual vehicles based on real road topologies. Parameters of vehicle's number, speed, acceleration and traffic congestion can be set separately. We run SUMO and MATLAB simultaneously, thus the mobility-related information of vehicles can be delivered to MATLAB.

Fig. 5(b) is the snapshot of Fuxingmen overpass for SUMO simulation, where there are four parallel lanes in the same direction. Therefore, vehicles' overtaking may occur, leading to changing of CV candidates for each UV. This is the volatility characteristic of real-world scenario.

Fig. 6(a) shows the system power consumption performance in real-world topology scenario. The performance of QOCA-UCB and QC-UCB decreases due to volatility compared to

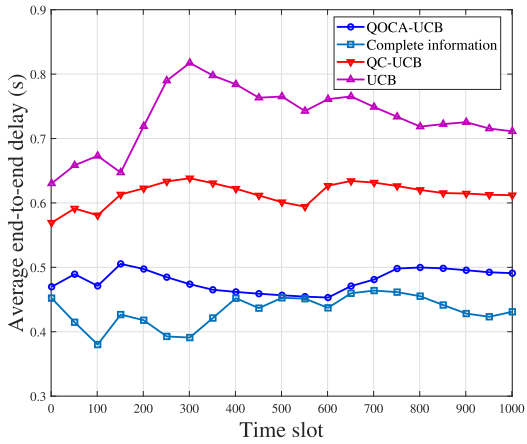


Fig. 8. Average end-to-end delay versus time slot under scenario 2.

that of scenario 1. However, the performance of QOCA-UCB is significantly better than that of UCB and QC-UCB. We can see that the energy consumption from AL method and FO-UCB is gradually increasing. Due to lack of task splitting, the task backlog grows accordingly, which in turn results in a growing computing power consumption. In addition, the fluctuation of the curve in real-world topology is obvious, because the distance between UVs and CVs is constantly changing. Fig. 6(b) also indicates that QOCA-UCB outperforms the other two UCB algorithms and converges more rapidly in terms of learning regret, which is more suitable for real-world implementation.

Fig. 7 shows the average backlog over time slot. It can be seen that QOCA-UCB can also perform the best backlog of local computing, task offloading, and edge computing queue, indicating that occurrence-awareness and conflict-awareness combined with queue-awareness in volatile scenario can achieve superior system performance.

End-to-end delay versus time slot under scenario 2 is shown in Fig. 8. The reason for curve fluctuation is still the continuous changing of vehicles' real-time positions. QOCA-UCB is near optimal and outperforms QC-UCB and UCB. This is because occurrence-awareness enhances UVs' ability of finding the best task offloading CVs, thereby reducing end-to-end delay.

Fig. 9 shows the impact of V. The system power consumption decreases and the end-to-end delay increases as V increases, indicating that our method laies more emphasis on power

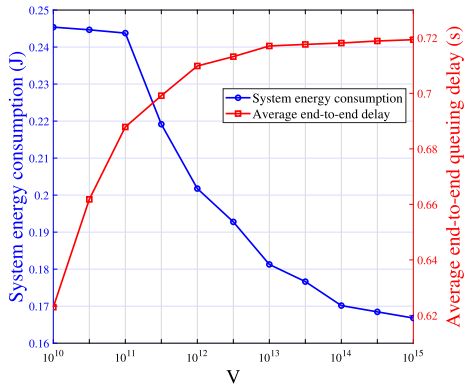


Fig. 9. Impact of V.

consumption reduction than the end-to-end delay decrement. Specifical 31.8%, and the queue delay increases by 13.5% when V increases from 10^{10} to 10^{15}. It demonstrates that our method can obtain good balance between power consumption and end-to-end delay.

VII. Conclusion

In this paper, we investigated the energy efficient task offloading problem under information uncertainty for VCEC system. We utilized Lyapunov optimization to decompose the original issue into three subproblems, i.e., task splitting and resource allocation at UV side, task offloading optimization, edge resource assignment for CV. Accordingly, we developed **TSLRA**, **QOCA-UCB** and **GECRA** algorithms to solve the three subproblems step by step. Our proposed approach can achieve queue-awareness, occurrence-awareness, and conflict-awareness by dynamically changing offloading and allocation strategies in light of real-time queue information, vehicle positions and matching results, respectively. We also theoretically analyze optimality, awareness property, learning regret, and complexity of our approach. Simulation results indicate that our approach can achieve superior performance in terms of energy consumption, learning regret, task backlog, and end-to-end delay.

Appendix A
Proof of Theorem 1

Based on the definition of Lyapunov drift plus penalty in (32), we can further derive that with every potential $\Theta(t)$ and $V \geqslant 0$, $\Delta_V L[\Theta(t)]$ is upper bounded by

$$\begin{aligned}
&\Delta_V L[\Theta(t)] \\
&\leqslant C + \sum_{n=1}^{N} \mathbb{E}[Q_n^{loc}(t)(A_n^{loc}(t) - W_n^{loc}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[Q_n^{off}(t)(A_n^{off}(t) - W_n^{off}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[Z_n^{Q,loc}(t)\left(\frac{Q_n^{loc}(t+1)}{\bar{A}_n^{loc}(t+1)} - \tau_{n,max}^{Q,loc}\right)|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[Z_n^{Q,off}(t)\left(\frac{Q_n^{off}(t+1)}{\bar{A}_n^{off}(t+1)} - \tau_{n,max}^{Q,off}\right)|\Theta(t)] \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{E}[G_{n,k}^{com}(t)(W_n^{off}(t) - W_{n,k}^{com}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{E}[G_{n,k}^{bac}(t)(\beta W_{n,k}^{com}(t) - W_{n,k}^{bac}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{E}[Z_{n,k}^{G,com}(t)\left(\frac{G_{n,k}^{com}(t+1)}{\bar{W}_n^{off}(t+1)} - \tau_{n,k,max}^{G,com}\right)|\Theta(t)] \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{E}[Z_{n,k}^{G,bac}(t)\left(\frac{G_{n,k}^{bac}(t+1)}{\beta\bar{W}_{n,k}^{com}(t+1)} - \tau_{n,k,max}^{G,bac}\right)|\Theta(t)] \\
&\quad + V\mathbb{E}[E(t)|\Theta(t)] \qquad (54)
\end{aligned}$$

where C is a constant. Substituting the queue expression (2), (3), (4), (5) and the following five formulas into the above formula, formula (60) is obtained.

$$\bar{A}_n^{loc}(t+1) = \frac{1}{t+1}\left[A_n^{loc}(t) + \sum_{j=0}^{t-1} A_n^{loc}(j)\right] \qquad (55)$$

$$\bar{A}_n^{off}(t+1) = \frac{1}{t+1}\left[A_n^{off}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)\right] \qquad (56)$$

$$\bar{W}_n^{off}(t+1) = \frac{1}{t+1}\left[W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)\right] \qquad (57)$$

$$\beta\bar{W}_{n,k}^{com}(t+1) = \frac{\beta}{t+1}\left[W_{n,k}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)\right] \qquad (58)$$

$$W_{n,k,max}^{com}(t) = \frac{\tau f_{k,max}}{\lambda_n} \qquad (59)$$

$$\begin{aligned}
&\Delta_V L[\Theta(t)] \\
&\leqslant C' + \sum_{n=1}^{N} \mathbb{E}[Q_n^{loc}(t)(A_n^{loc}(t) - W_n^{loc}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[Q_n^{off}(t)(A_n^{off}(t) - W_n^{off}(t))|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[(t+1)Z_n^{Q,loc}(t) \\
&\quad \times \left(\frac{Q_n^{loc}(t) + A_n^{loc}(t) - W_n^{loc}(t)}{A_n^{loc}(t) + \sum_{j=0}^{t-1} A_n^{loc}(j)}\right)|\Theta(t)] \\
&\quad + \sum_{n=1}^{N} \mathbb{E}[(t+1)Z_n^{Q,off}(t)
\end{aligned}$$

$$\times \left(\frac{Q_n^{off}(t) + A_n^{off}(t) - W_n^{off}(t)}{A_n^{off}(t) + \sum_{j=0}^{t-1} A_n^{off}(j)} \right) |\Theta(t)]$$
$$+ \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[G_{n,k}^{com}(t)(W_n^{off}(t) - W_{n,k}^{com}(t))|\Theta(t)]$$
$$+ \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[G_{n,k}^{bac}(t)(\beta W_{n,k}^{com}(t) - W_{n,k}^{bac}(t))|\Theta(t)]$$
$$+ \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}\left[(t+1) Z_{n,k}^{G,com}(t) \right.$$
$$\left. \times \left(\frac{G_{n,k}^{com}(t) + W_n^{off}(t) - W_{n,k}^{com}(t)}{W_n^{off}(t) + \sum_{j=0}^{t-1} W_n^{off}(j)} \right) |\Theta(t) \right]$$
$$+ \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[(t+1) Z_{n,k}^{G,bac}(t)$$
$$\times \left(\frac{G_{n,k}^{bac}(t) + \beta W_{n,k}^{com}(t) - W_{n,k}^{bac}(t)}{\beta[W_{n,k}^{com}(t) + \sum_{j=0}^{t-1} W_{n,k}^{com}(j)]} \right) |\Theta(t)]$$
$$+ V\mathbb{E}[E(t)|\Theta(t)] \tag{60}$$

where $C' = C - \sum_{n=1}^{N} Z_n^{Q,loc}(t)\tau_{n,max}^{Q,loc} - \sum_{n=1}^{N} Z_n^{Q,off}(t)\tau_{n,max}^{Q,off} - \sum_{n=1}^{N}\sum_{k=1}^{K} Z_{n,k}^{G,com}(t)\tau_{n,k,max}^{G,com} - \sum_{n=1}^{N}\sum_{k=1}^{K} Z_{n,k}^{G,bac}(t)\tau_{n,k,max}^{G,bac}$ is also a constant that does not impact Lyapunov optimization.

We need to "opportunistically" minimize the drift minus penalty in each time slot. Thus, we get the objective funtion in (33). This completes the proof of Theorem 1.

Appendix B
Proof of Theorem 2

We denote the learning regret in slot t as LR_t, which can be calculated by

$$LR_t = \mathbb{E}\left\{ \sum_{n=1}^{N} \sum_{k=1}^{K} x_n^k(t)\gamma_n^k(t) - x_n^{k^*}(t)\gamma_n^{k^*}(t) \right\} \tag{61}$$

According to Theorem 1 in [42], we can obtain that LR_t is upper bounded by

$$LR_t \leq 2\eta^2 \sum_{k=1,k\neq k^*}^{K} \frac{\ln(t - t_{n,k}^{fir})}{\Delta\gamma_n^k} + \left(1 + \frac{\pi^2}{3}\right) \sum_{k=1,k\neq k^*}^{K} \Delta\gamma_n^k \tag{62}$$

which can be further expressed as

$$LR_t \leq 2\eta^2 \sum_{k=1,k\neq k^*}^{K} \frac{\ln(T - t_{n,k}^{fir})}{\Delta\gamma_n^k} + \left(1 + \frac{\pi^2}{3}\right) \sum_{k=1,k\neq k^*}^{K} \Delta\gamma_n^k \tag{63}$$

Therefore, by summing the learning regret of $t = 1, 2, \ldots, T$ we can get the total learning regret

$$LR = \sum_{t=1}^{T} LR_t \leq \sum_{t=1}^{T} \sum_{k=1,k\neq k^*}^{K} \left[2\eta^2 \frac{\ln(T - t_{n,k}^{fir})}{\Delta\gamma_n^k} + \left(1 + \frac{\pi^2}{3}\right) \Delta\gamma_n^k \right] \tag{64}$$

The proof finishes.

References

[1] M. Bute, P. Fan, G. Liu, F. Abbas, and Z. Ding, "A collaborative task offloading scheme in vehicular edge computing," in *Proc. IEEE 93rd Veh. Technol. Conf.*, 2021, pp. 1–5.

[2] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. Foy, and Y. Zhang, "Mobile edge cloud system: Architectures, challenges, and ap-proaches," *IEEE Syst. J.*, vol. 12, no. 3, pp. 2495–2508, Sep. 2018.

[3] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[4] A. B. Reis, S. Sargento, and O. K. Tonguz, "Parked cars are excellent roadside units," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2490–2502, Sep. 2017.

[5] 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.

[6] N. Cha, C. Wu, T. Yoshinaga, Y. Ji, and K.-L. A. Yau, "Virtual edge: Exploring computation offloading in collaborative vehicular edge computing," *IEEE Access*, vol. 9, pp. 37739–37751, 2021.

[7] A. Boukerche and V. Soto, "An efficient mobility-oriented retrieval protocol for computation offloading in vehicular edge multi-access network," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 6, pp. 2675–2688, Jun. 2020.

[8] M. Ma, D. He, H. Wang, N. Kumar, and K.-K. R. Choo, "An efficient and provably secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8065–8075, Oct. 2019.

[9] P. Qin, Y. Fu, X. Feng, X. Zhao, S. Wang, and Z. Zhou, "Energy-efficient resource allocation for parked-cars-based cellular-V2V heterogeneous networks," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 3046–3061, Feb. 2022.

[10] "Watch: Is Tesla Sentry Mode Worth The Energy It Consumes?," [Online]. Available: https://insideevs.com/features/410525/is-tesla-sentry-mode-worth-the-energy

[11] W. Bao, H. Chen, Y. Li, and B. Vucetic, "Joint rate control and power allocation for non-orthogonal multiple access systems," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 12, pp. 2798–2811, Dec. 2017.

[12] H. Liao, Z. Zhou, X. Zhao, B. Ai, and S. Mumtaz, "Task offloading for vehicular fog computing under information uncertainty: A matching-learning approach," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf.*, 2019, pp. 2001–2006.

[13] S. Kazmi *et al.*, "A novel contract theory-based incentive mechanism for cooperative task-offloading in electrical vehicular net-works," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 8380–8395, Jul. 2022.

[14] J. Du, F. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1079–1092, Feb. 2019.

[15] H. Yu, R. Liu, Z. Li, Y. Ren, and H. Jiang, "An RSU deployment strategy based on traffic demand in vehicular ad hoc networks (VANETs)," *IEEE Internet Things J.*, vol. 9, no. 9, pp. 6496–6505, May 2022.

[16] C. Lin, G. Han, X. Qi, M. Guizani, and L. Shu, "A distributed mobile fog computing scheme for mobile delay-sensitive applications in sdn-enabled vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5481–5493, May 2020.

[17] 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.

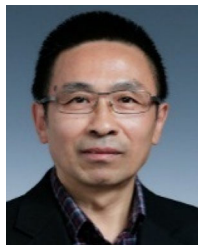
[18] A. Mourad, H. Tout, O. Wahab, H. Otrok, and T. Dbouk, "Ad hoc vehicular fog enabling cooperative low-latency intrusion detection," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 829–843, Jan. 2021.

[19] O. Wahab, H. Otrok, and A. Mourad, "Vanet QoS-OLSR: QoS-based clustering protocol for vehicular ad hoc networks," *Comput. Commun.*, vol. 36, no. 13, pp. 1422–1435, 2013.
[20] J. Xue and Y. An, "Joint task offloading and resource allocation for multi-task multi-server NOMA-MEC networks," *IEEE Access*, vol. 9, pp. 16152–16163, 2021.
[21] M. Qin *et al.*, "Service-oriented energy-latency tradeoff for IoT task partial offloading in MEC-enhanced multi-RAT networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1896–1907, Feb. 2021.
[22] J. Yan, S. Bi, Y. 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.
[23] H. Tout, C. Talhi, N. Kara, and A. Mourad, "Selective mobile cloud offloading to augment multi-persona performance and viability," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 314–328, Apr.–Jun. 2019.
[24] P. Cai, F. Yang, J. Wang, X. Wu, Y. Yang, and X. Luo, "JOTE: Joint offloading of tasks and energy in fog-enabled IoT networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3067–3082, Apr. 2020.
[25] X. Huang, W. Fan, Q. Chen, and J. Zhang, "Energy-efficient resource allocation in fog computing networks with the candidate mecha-nism," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8502–8512, Sep. 2020.
[26] H. Tout, A. Mourad, N. Kara, and C. Talhi, "Multi-persona mobility: Joint cost-effective and resource-aware mobile-edge computation offloading," *IEEE/ACM Trans. Netw.*, vol. 29, no. 3, pp. 1408–1421, Jun. 2021.
[27] P. Qin, Y. Zhu, X. Zhao, X. Feng, J. Liu, and Z. Zhou, "Joint 3D-location planning and resource allocation for XAPS-enabled c-noma in 6G heterogeneous Internet of Things," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10594–10609, Oct. 2021.
[28] J. Liu, X. Zhao, P. Qin, S. Geng, and S. Meng, "Joint dynamic task offloading and resource scheduling for WPT enabled space-air-ground power Internet of Things," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 2, pp. 660–677, Mar. 2022.
[29] F. Chiti, R. Fantacci, and B. Picano, "A matching theory framework for tasks offloading in fog computing for IoT systems," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5089–5096, Dec. 2018.
[30] C. Shi and C. Shen, "Multi-player multi-armed bandits with collision-dependent reward distributions," *IEEE Trans. Signal Process.*, vol. 69, pp. 4385–4402, 2021.
[31] S. Yang and Y. Gao, "An optimal algorithm for the stochastic bandits while knowing the near-optimal mean reward," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 2285–2291, May 2021.
[32] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[33] Y. Sun, X. Guo, S. Zhou, Z. Jiang, X. Liu, and Z. Niu, "Learning-based task offloading for vehicular cloud computing systems," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–7.
[34] M. Youssef, V. Veeravalli, J. Farah, C. Nour, and C. Douillard, "Resource allocation in NOMA-based self-organizing networks using stochastic multi-armed bandits," *IEEE Trans. Commun.*, vol. 69, no. 9, pp. 6003–6017, Sep. 2021.
[35] H. Liao *et al.*, "Learning-based intent-aware task offloading for air-ground integrated vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5127–5139, Aug. 2021.
[36] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.
[37] C. Liu, M. Bennis, M. Debbah, and H. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.
[38] C. Xu, G. Zheng, and X. Zhao, "Energy-minimization task offloading and resource allocation for mobile edge computing in NOMA heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16001–16016, Dec. 2020.
[39] Y. Mao, J. Zhang, S. Song, and K. Letaief, "Power-delay tradeoff in multi-user mobile-edge computing systems," in *Proc. IEEE Glob. Commun. Conf.*, 2016, pp. 1–6.
[40] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan and Claypool, 2010.
[41] S. Jung, J. Hong, and K. Nam, "Current minimizing torque control of the IPMSM using ferrari's method," *IEEE Trans. Power Electron.*, vol. 28, no. 12, pp. 5603–5617, Dec. 2013.
[42] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.

Peng Qin (Member, IEEE) received the B.S. and Ph.D. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 2009 and 2014, respectively. From 2012 to 2013, he was a Visiting Scholar with the University of Victoria, Victoria, BC, Canada. He is currently an Associate Professor with the Department of Electrical and Electronic Engineering, North China Electric Power University, Beijing, China. His research interests include resource allocation in Internet of Things, smart grid communications, space air ground integrated networks, and vehicular networks. He was the recipient of the International Communications Signal Processing and Systems Conference Best Paper Award, and International Conference on Artificial Intelligence in China Best Paper Award in 2019, 2020, and 2021, respectively.

Yang Fu is currently working toward the B.S. degree with the School of Electrical and Electronic Engineering, North China Electric Power University, Beijing, China. His research interests include resource allocation in smart grid communications, space air ground integrated networks, vehicular networks, and the Internet of Things.

Guoming Tang (Member, IEEE) received the B.S. and M.S. degrees from the National University of Defense Technology, Changsha, China, in 2010 and 2012, respectively, and the Ph.D. degree in computer science from the University of Victoria, Victoria, BC, Canada, in 2017. He is currently a Research Fellow with Peng Cheng Laboratory, Shenzhen, China. He was also a Visiting Research Scholar with the University of Waterloo, Waterloo, ON, Canada, in 2016. His research interests include green computing and computing for green.



Xiongwen Zhao (Senior Member, IEEE) received the Ph.D. degree from the Helsinki University of Technology, Helsinki, Finland, in 2002. From 2004 to 2011, he was with Elektrobit Corporation. He is currently a Chair Professor of information and communications engineering with North China Electric Power University, Beijing, China, and responsible for the projects supported by the National Science Foundation of China and the Double First Class construction project by Ministry of Education. He was the recipient of the IEEE VTS Neal Shepherd Memorial Best Propagation Paper Award, IEEE ISAPE and IWCMC best paper awards in 2014, 2018, and 2019, respectively. He was the TPC Co-Chair, and a Keynote Speaker for numerous international conferences. He is an Associate Editor for the *IET Communications* and a Fellow of Chinese Institute of Electronics.

Suiyan Geng received the M.S. and Ph.D. degrees from the Helsinki University of Technology, Helsink, Finland, in 2003 and 2011, respectively. She is currently an Associate Professor with North China Electric Power University, Beijing, China. She was the recipient of the IEEE VTS Neal Shepherd Memorial Best Propagation Paper Award in 2014. Her research interests include millimeter-wave and ultra-wideband radio wave propagation and stochastic channel modeling for future-generation radio systems and technologies.