IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received November 11, 2019, accepted November 29, 2019, date of publication December 5, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2957749

Collaborative Vehicular Edge Computing Networks: Architecture Design and Research Challenges

RENCHAO XIE[1,2], QINQIN TANG[1,2], QIUNING WANG[1,2], XU LIU[1,2], FEI RICHARD YU[3], AND TAO HUANG[1,2]
[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Purple Mountain Laboratories, Nanjing 211111, China
[3]Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada

Corresponding author: Renchao Xie (renchao_xie@bupt.edu.cn)

This work was supported by the Development Strategy of Future Network Operating System under Grant 2019-XY-5.

ABSTRACT The emergence of augmented reality (AR), autonomous driving and other new applications have greatly enriched the functionality of the vehicular networks. However, these applications usually require complex calculations and large amounts of storage, which puts tremendous pressure on traditional vehicular networks. Mobile edge computing (MEC) is proposed as a prospective technique to extend computing and storage resources to the edge of the network. Combined with MEC, the computing and storage capabilities of the vehicular network can be further enhanced. Therefore, in this paper, we explore the novel collaborative vehicular edge computing network (CVECN) architecture. We first review the work related to MEC and vehicular networks. Then we discuss the design principles of CVECN. Based on the principles, we present the detailed CVECN architecture, and introduce the corresponding functional modules, communication process, as well as the installation and deployment ideas. Furthermore, the promising technical challenges, including collaborative coalition formation, collaborative task offloading and mobility management, are presented. And some potential research issues for future research are highlighted. Finally, simulation results are verified that the proposed CVECN can significantly improve network performance.

INDEX TERMS Vehicular network, edge computing, computation offloading, architecture design.

I. INTRODUCTION

Vehicular networks have received wide attention since their inception [1], [2]. In vehicular networks, vehicles can communicate with other vehicles via an on-board unit (OBU) or with a road side unit (RSU). And vehicles can implement applications such as accident warning, road traffic information inquiry, workshop communication, Internet access services and so forth. The goal of vehicular networks is to build an inter-vehicle communication network on the road that is open, inexpensive, and easy to deploy. Nowadays, the vehicular networks are a critical part of the transportation system, making the transportation system more intelligent, safe and convenient. However, the emergence of a series of computation-intensive and delay-sensitive new applications pose a severe challenge to existing vehicular networks. How to improve the network computational capabilities and reduce the communication delay has become an important issue for vehicular networks.

The emerging edge computing technology may be a possible direction to address the challenges of vehicular networks [3]–[5]. Driven by the more and more new mobile applications, such as Internet of Things (IoT), augmented reality (AR)/virtual reality (VR), 4K/8K, the mobile network faces exponential growth in data traffic and connections. In order to cope with these challenges, mobile edge computing (MEC) has been proposed as a new promising technology. The main features of the MEC are proximity, low latency and location awareness. By deploying MEC servers at the edge of the network, such as cellular access points and Wi-Fi hotspots, the computation tasks generated by the user application can be processed at the edge of the network, which significantly reduces the pressure on the core network and optimizes the user experience. Therefore, MEC is considered to be an essential technique to improve network performance.

The associate editor coordinating the review of this manuscript and approving it for publication was Changqing Luo.

Therefore, combining MEC and vehicular networks has been considered as a complementary enabling technology and has been attracted much attention in recent years [6]–[8]. In [6], the authors proposed the architecture named vehicular fog computing (VFC) in which vehicles are considered as network infrastructures. Meanwhile, the author also analyzed typical scenes under different vehicle conditions. The authors of [7] investigated content prefetching and distribution in the 5G enabled vehicular ad hoc network. Then they proposed a scheme to meet the rapid topology change and unbalanced traffic. The concept of vehicular cloud is introduced in [8], and the authors provided an overview on the applications, security issues and security solutions. In summary, the combination of MEC and vehicular networks has become a trend to meet the low latency and high bandwidth requirements of service applications.

Although lots of research has been done on the combination of MEC and vehicular networks, the framework of the combination of MEC and vehicular networks mainly focuses on application scenarios, application directions and vehicle task offloading [9], [10], and pays little attention to the cooperation and task scheduling of vehicles. Therefore, in order to improve the task processing efficiency of the RSU, we propose a novel collaborative computation architecture. In this article, we introduce the design principles and architecture. Then we describe the functional modules in detail, followed by communication process and deployment ideas. We also discuss the main challenges for the collaborative edge computing architecture while decentralizing the cloud computing infrastructure to the edge. Simulation results evaluate the performance of the proposed collaborative architecture. The main contribution of this work is listed as follows.

- We first propose the design principles that need to be considered when combining edge computing with vehicle networks. Based on the above principles, we present the detailed architecture and the main functional modules of CVECN. Furthermore, we briefly describe the communication process and the overall implementation in the proposed CVECN.
- Then, we analyze the key challenges of the proposed CVECN, including collaborative coalition formation, collaborative task offloading and mobility management. Moreover, we also discuss some other potential research issues that should be taken into account in the implementation of CVECN.
- Finally, we conduct extensive simulations to evaluate the performance of the proposed CMDP based collaborative task offloading scheme. It is shown that the proposed scheme outperforms reference schemes in terms of making effective task processing decisions. The proposed scheme can achieve the minimum average task processing delay and task loss ratio.

The rest of this paper is organized as follows. Related work and existing developments to MEC and vehicular networks are presented in Section II. The design principles, architecture, functional modules and implementation ideas of CVECN are designed in Section III. Following that, we investigate the challenges in collaborative coalition formation, collaborative computation offloading scheme and mobility management in Section IV. Moreover, we also discuss the basic solutions to these challenges. Then numerical simulation results are then provided to verify the efficiency of the proposed architecture in Section V. Finally, we conclude the article in Section VI.

II. RELATED WORK

In recent years, MEC has received extensive attention due to its excellent support for computation-intensive applications and delay-sensitive applications. In [3], authors introduced the definition of edge computing and several case studies, ranging from cloud offloading to smart home and city, as well as collaborative edge to materialize the concept of edge computing. They also presented several challenges and opportunities in the field of edge computing. The article [4] described the main use cases and reference schemes for the applicable MEC. The authors also introduced the existing concepts of integrating MEC functionality into mobile networks and discusses the latest developments in MEC standardization. In [5], authors provided a comprehensive overview of the latest MEC research, especially joint radio and computing resource management. The authors also discussed issues related to MEC research and future research directions, including MEC system deployment, caching, mobility management, energy conservation, and privacy. In [11], the authors studied the latency performance of a large-scale MEC network. To study the tradeoffs between these metrics and constraints, the MEC network is modeled using geometry featuring diversified aspects of wireless access and computing. Based on the model, the latency is analyzed by applying the theories of stochastic geometry, queuing and parallel computing. The MEC program is considered by ETSI to be an important part of unified telecommunications and cloud computing. The goal is to create a standardized open environment that will allow the integration of vendors, service providers and third-party applications for edge computing platforms [12].

Task scheduling is a key technology of MEC and has an essential impact on system efficiency and user experience. In [13], the authors investigated the problem of distributed computation offloading decision making among multiple mobile devices in a cloudlet based mobile cloud computing system. The article [14] presented an integrated framework for computation offloading and interference management in heterogeneous wireless cellular networks with mobile edge computing. And then, the computation offloading decision, physical resource block allocation, and MEC computing resource allocation problems are formulated in this framework. In order to jointly tackle computation offloading and content caching strategies in wireless cellular networks with mobile edge computing, the article [15] formulated the computation offloading decision, resource allocation and content caching strategy as an optimization problem, considering the total revenue of the network. In many existing works, the offloading schemes are designed to optimize the energy consumption, delay performance, or cost for resource usage [16]–[19]. In [16], the authors studied the resource allocation for a multiuser MECO system based on time-division multiple access (TDMA)/orthogonal frequency-division multiple access (OFDMA), accounting for both the cases of infinite and finite cloud computational capacities. It shows that to minimize weighted sum mobile energy consumption, the optimal resource allocation policy should have a threshold-based structure. In [17], authors tackled the computation offloading problem in a mixed fog/cloud system by jointly optimizing the offloading decisions and the allocation of computing resources, transmit power and radio bandwidth, while guaranteeing user fairness and maximum tolerable delay. Different from the aforementioned works, in [18], authors investigated the problem of joint energy consumption, delay, and payment cost (E&D&P) minimization for the MDs in a fog computing heterogeneous network. In [19], the authors investigated a joint radio and computing resource allocation problem to optimize the system performance and improve user satisfaction. Considering the distributive features of the IoT framework, a matching theory, as a semi-distributive solution approach, is proposed to find a stable matching between the users and resources.

Combining MEC and vehicular networks have also become a research hotspot in recent years. Literature [20] proposed a MEC-based multipath transmission workload equalization optimization scheme. The solution is applied to the edge computing architecture by combining edge nodes and cloud computing data centers. Different types of application requests are dynamically assigned to each edge node by considering the response time of different types of vehicles, and then the computing resources of each edge node are optimally allocated to the corresponding VM to reduce the average response time of the vehicle application. In [21], the authors proposed a collaborative computing offloading problem for vehicular networks where MEC and cloud computing coexist. This problem can be identified as constrained optimization by jointly optimizing the calculation of offload decisions and computing resource allocation to maximize system availability. Besides, a CCORAO scheme is proposed to solve the problem, which includes computation offloading strategy game and resource allocation. The DCORA algorithm is developed for the CCORAO solution to reduce system complexity without sacrificing performance. In [22], the authors studied the computational overhead minimization problem by jointly optimizing communication and computing resources in the MEC-enabled vehicle network. First, the authors converted the non-convex problem into an equivalent problem. Then the authors divided the equivalence problem into two levels. In addition, this paper proposes a low complexity algorithm to obtain the optimal solution. Numerical results show that this solution can save a lot of computational overhead.

III. COLLABORATIVE VEHICULAR EDGE COMPUTING NETWORK ARCHITECTURE

In this section, the design principles are first introduced in Subsection III-A, the CVECN architecture is designed later, then the logical functional modules, the communication process and the deployment and implementation are analyzed and discussed in Subsection III-D and Subsection III-E, respectively.

A. DESIGN PRINCIPLES

The network performance of CVECN has been significantly improved by deploying MEC servers in the vehicular networks. However, CVECN involves RSUs, vehicles and MEC servers, which is complex and hierarchical. Therefore, designing the architecture of CVECN is important and challenging. In order to ensure the continuous and stable operation of CVECN, we highlight some design principles below.

Scalability refers to the ability of the system to respond to the vehicles and edge computing application changes. On the one hand, there are often a large number of vehicles registered to enter or leave in CVECN, so the system should be able to configure quickly and stably. On the other hand, CVECN should also be able to support installation and uninstallation of applications. Scalability is an important factor for all systems.

Performance improvement is an important factor for all systems. Performance improvement for CVECN includes low energy consumption and low latency. The energy consumption of the MEC server in the vehicle leads to the consumption of energy, such as gasoline. Lower energy consumption leads to lower carbon emissions. So the CVECN should pay more attention to the limitation of energy consumption. Moreover, latency has a large impact on the user experience and thus should be considered.

Mobility support is critical to system quality of service. In CVECN, the vehicles are moving fast, and the topology of the network is changing rapidly. Therefore, the connection between the RSU and the vehicles is not stable. The computation offloading of the vehicle is susceptible. Therefore, comprehensive support for the mobility of the vehicles is necessary.

B. ARCHITECTURE DESIGN

Based on the design principles, we design the CVECN architecture, as illustrated in Fig. 1, which consists of a variety of MEC servers, vehicles and RSUs. MEC servers have computing and storage capabilities. They install edge computing platforms that can run edge computing applications and handle computation tasks. There are two types of edge computing servers in CVECN: MEC servers at vehicles and MEC servers at RSUs. As for the former, some applications can be installed on these servers, such as map applications or navigation programs. Due to the limited computing resources of the

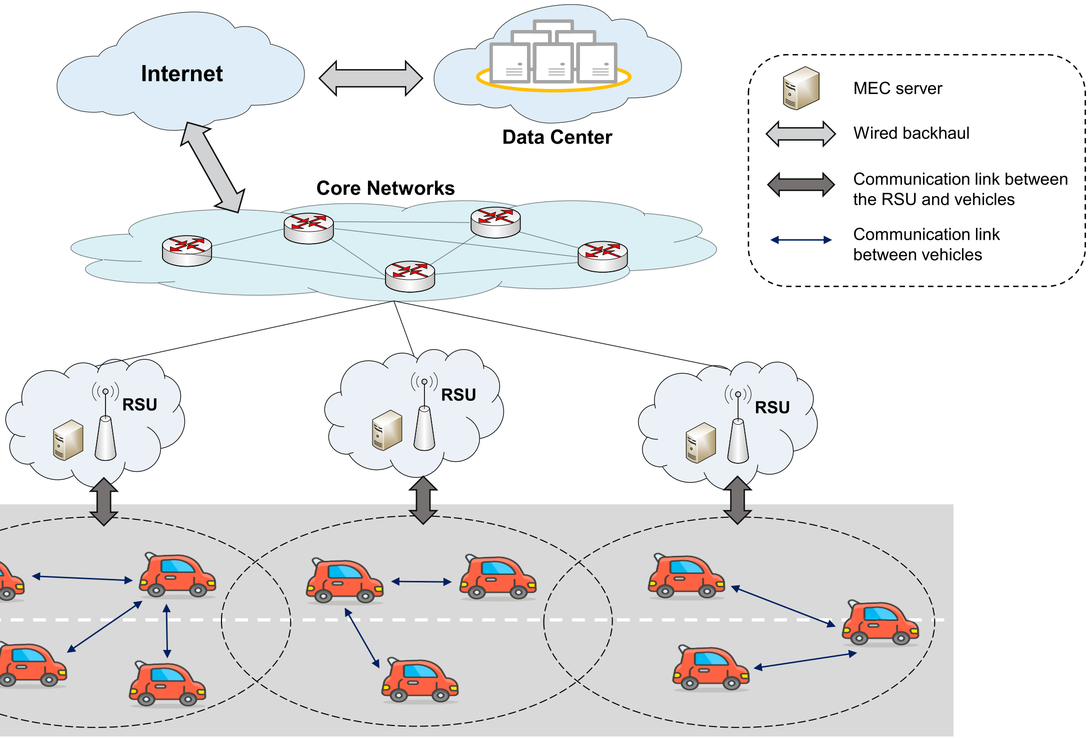

FIGURE 1. System architecture of CVECN.

vehicle, it may offload some computation tasks generated by applications to surrounding MEC servers at the vehicles and RSUs for processing. The vehicle that needs offload computation tasks to other location for processing is called the mission-initiated vehicle. For the latter, the RSUs have a broad range of services, so the MEC servers have more resources. They can provide edge computing services for vehicles in the area, as well as handle the computation tasks that the vehicles offload. When all the MEC servers in the range are busy, RSUs will send the computation tasks to the data center in the core network for processing.

In CVECN, the vehicles can communicate with each other over the wireless link to form a cluster and collaborate on computation tasks. The SDN-based unified coordination plane in the RSUs can manage all vehicles and clusters within range. More specifically, it can query resource status information of the MEC servers at vehicles and control the servers through the southbound interface. The unified coordination plane determines the collaboration strategy based on the state of the vehicles. The participating collaborative vehicles can then communicate with each other to exchange information and computation tasks in a timely manner. Through collaborative processing tasks, CVECN further speeds up task processing and reduces the mission-initiated vehicle waiting time.

C. THE LOGICAL FUNCTIONAL MODULE DESIGN

According to the above-proposed CVECN architecture, details of each functional module of the architecture designed above are shown in Fig. 2. The CVECN architecture consists of three major functional components, including the service and application plane, CVECN platform and the infrastructure resources plane, as elaborated in the following. Under this architecture, the mission-initiated vehicle's computation tasks can be processed faster, and various applications and the management functions of services and platforms can run smoothly.

1) SERVICE AND APPLICATION PLANE

Service and application plane provides standardized application service interfaces and management interfaces. The application service interface is open to the mission-initiated vehicles, which receive application registration requests, and exposes some of the control plane functionality to the applications. The management interface provides system management and control information to enable the release of control commands and the supervision of system status. The standardized interface provides excellent scalability and efficient system management, simplifying the expansion, use, and maintenance of the entire system.

2) CVECN PLATFORM

This platform is the core of the CVECN architecture, which guarantees the regular and efficient operation of cooperation task processing, resource allocation, orchestration, and unified management. The platform consists of an RSU unified coordination plane and computing platforms.

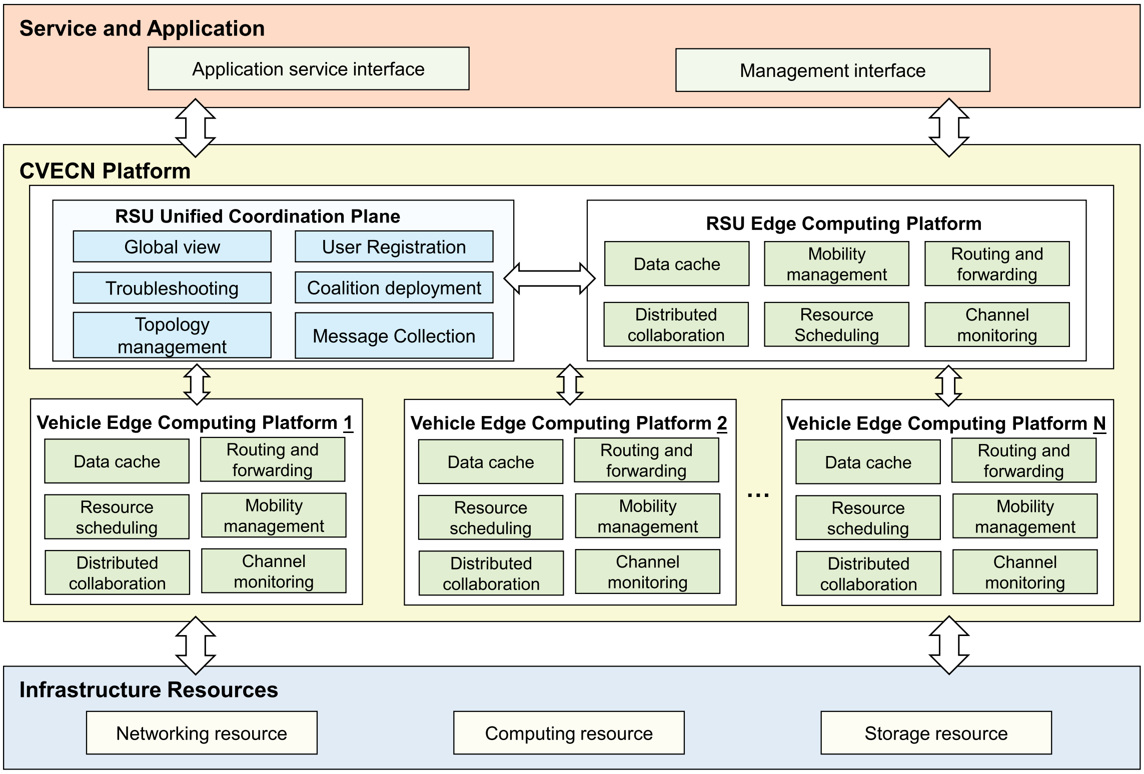

FIGURE 2. The logical functional module of CVECN architecture.

The RSU unified coordination plane: The RSU unified coordination plane performs is unified and coordinated on computing platforms, mainly to complete the scheduling and distribution of the mission-initiated vehicle's computation tasks, and to ensure the rapid processing of computation tasks. Moreover, through the virtualization technology, the RSU unified coordination plane performs can provide the corresponding information delivery interface to ensure accurate, fast transmission and processing of services.

The unified coordination plane mainly implements global view, user registration, coalition deployment, topology management, troubleshooting, message collection and other functions. The global view provides global topology information based on the information sent by the vehicles, dynamically displays vehicle network information within RSU coverage in real-time; the user registration allows the mission-initiated vehicles to dynamically access and complete the registration, all mission-initiated vehicles need to register with the RSU before offloading the computation task, and then the RSU will offload the computation task to different locations according to different situations; coalition deployment module finds the vehicle collaboration coalition according to the corresponding algorithm by receiving the state information of the vehicle network, and issues the instruction of coalition information; topology management can issue instructions according to business requirements, and control whether the vehicle is involved in the network; troubleshooting mainly completes tasks such as judgment of faults, alarms, etc.; message collection is a relatively basic function, mainly responsible for information interaction, and information collection with the mission-initiated vehicles and collaborative vehicles and so on.

Edge computing platform: The edge computing platform, including the RSU edge computing platform and vehicle edge computing platform, ensures the normal and effective operation of task processing, resource allocation, and scheduling. As a task processing platform, the edge computing platform needs to deploy corresponding functional modules to meet the service function requirements of the platform, including resource scheduling, channel monitoring, routing and forwarding, collaboration, mobility management, and data caching. The resource scheduling module allows the edge computing platform to call the underlying infrastructure resources to meet the computation task requirements; the channel monitoring can acquire the channel information at any time, it mainly includes channel changes between the mission-initiated vehicles, collaborative vehicles, and RSUs due to the high-speed movement of the mission-initiated vehicles; the routing and forwarding, the decentralized collaboration and the mobility management module implement the functional scheduling, task allocation, mobility support, and other functions between the vehicles; data caching can effectively ensure the data security of edge computing platform. If necessary, users can continue to add the corresponding function modules to achieve the required functions.

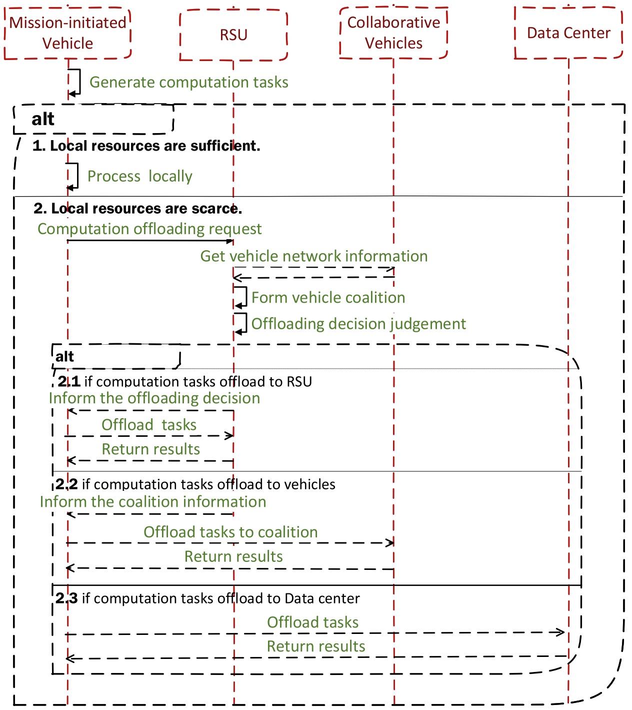

FIGURE 3. The typical task processing procedure.

3) INFRASTRUCTURE RESOURCES

Infrastructure resources refer to the underlying physical resources of CVECN architecture, including computing resources, storage resources, and network resources.

The cloudlet provides computing and storage resources to the upper layer for unified deployment through virtualization technology. Network resources mainly include communication links between vehicles and RSUs. The application of virtualization technology can effectively solve the problem of heterogeneous resources of the underlying resources, and provide standardized interfaces for use by the upper control plane.

D. COMMUNICATION PROCESS

In CVECN, the typical task processing procedure is shown in Fig. 3. According to different scenarios and requirements, the mission-initiated vehicle has computation tasks that need to be processed. If the local computing resource is sufficient, mission-initiated vehicle processes the computation tasks locally. Otherwise, the mission-initiated vehicle will offload these tasks to RSUs, vehicles, or data centers for remote computation processing.

Offloading to the RSU: when mission-initiated vehicle requests to offload its tasks, it first communicates with the RSU. As the main body of the computation offloading decision, after receiving the mission-initiated vehicle's computation task request, the RSU first collects the resource usage of the RSU itself and the information of nearby vehicles. The RSU then directs the collaborative vehicles to form a vehicle coalition and compares the latency and energy consumption of the computation task between the RSU and the vehicle coalition. If the latency and power consumption are smaller in the RSU processing, the computation task will be processed at the RSU. The RSU returns an offloading decision to the mission-initiated vehicle, it offloads the computation task to the RSU, and returns the result after the RSU processing is completed.

Offloading to the collaborative vehicles: whether the computation task will be offloaded to the collaborative vehicles depends on the real-time link states, vehicle resource usage and the requirements of the computation task. RSU performs coordinated scheduling of vehicles within its service range. According to the computing resources of vehicles and the channel link information, a vehicle coalition is constituted to provide the mission-initiated vehicle with computation offloading services. If the RSU judges that the vehicle coalition can better meet the task requirements, and the delay or energy consumption is lower than that of RSU, the computation task will be offloaded to the vehicle coalition for processing.

Specifically, in vehicle coalition offloading mode, the RSU will first send information about vehicle coalition to mission-initiated vehicle. Then mission-initiated vehicle communicates with the vehicles in the vehicle coalition and offloads the computation tasks.

Offloading to the data center: if the vehicles and RSU cannot meet the computation requirements of the mission-initiated vehicle, the computation tasks need to be forwarded to the remote data center through the RSU. The remote data center will return the result.

E. DEPLOYMENT AND IMPLEMENTATION

In the CVECN architecture, edge computing platforms mainly deployed by cloudlets. The cloudlet is based on OpenStack and runs edge computing applications in MEC servers. These servers can be installed in a single or aggregated manner, which is up to the network needs. The architecture adopts the idea of separating the SDN control plane from the data plane. The unified coordination plane can manage edge computing platforms in the network, which acts as the control plane. The unified coordination plane located in the RSUs manages and performs encrypted transmission with the edge computing platforms through the southbound interface based on the OpenFlow protocol. The edge computing platforms use the southbound interface to upload a status dictionary, which contains three types of information, including mobile device information, computation task information, and edge server operation information. The controller uses the southbound interface to deliver collaboration policies, topology commands, and other control information.

IV. THE CHALLENGES AND OPEN ISSUES FOR CVECN ARCHITECTURE

The designed CVECN architecture can bring various benefits to improve network performance and vehicle users' QoE. And at the same time, the CVECN architecture also

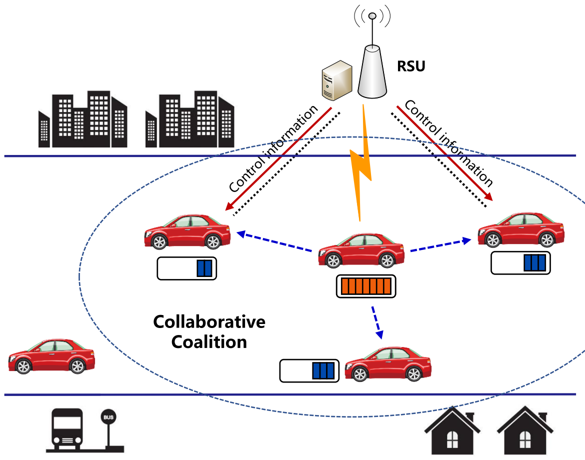

FIGURE 4. **Illustration of collaborative task processing in CVECN.**

introduces new challenges and open research issues, such as the establishing collaborative coalition, the design of collaborative task offloading scheme and mobility management and so forth, which will be discussed in this section.

A. COLLABORATIVE COALITION FORMATION

In CVECN architecture, mission-initiated vehicle's local resources are limited and cannot meet the latency requirements and computing resource requirements of intensive computation tasks. Therefore, the computation tasks need to be offloaded to the collaborative vehicles, RSU, or data center for processing. When mission-initiated vehicle's computation tasks are offloaded to collaborative vehicles, considering that the computing power and power consumption of the vehicle itself are limited, if tasks are offloaded to a single vehicle, too many computation tasks will not only affect the vehicle's processing of its own computation tasks, but also cause a large delay in the processing of the offloading tasks. Therefore, the mission-initiated vehicle needs to distribute computation tasks to multiple collaborative vehicles in the vicinity. In general, there are a large number of vehicles around the mission-initiated vehicle that can be offloaded. How to select suitable collaborative vehicles for offloading is a problem that needs to be paid first attention.

Therefore, in order to solve the above problems, how to form a vehicle collaboration coalition is very important. As shown in Fig. 4, the formation of the vehicle coalition determines the scope of computation offloading collaboration, which ensures that the vehicle participating in the collaboration process can provide enough computing resources and ensure that the computation offloading collaboration normally works according to mission-initiated vehicle's own needs. In addition, the vehicle can also refuse to access the vehicle collaboration coalition according to the needs of the service and the predicted state of the network link, and guarantee the resource requirements of the vehicle.

For the CVECN architecture, the formation of a vehicle coalition is completed in RSU unified coordination plane. Vehicles within the coverage of the RSU can send their own resource usage information, link information, etc. to the unified coordination plane. After receiving the information, the coordination plane through the coalition deployment module selects the corresponding vehicle to form a coalition and sends the information to the collaborative vehicle and mission-initiated vehicle. The coalition deployment module of the unified coordination plane not only needs to complete the formation of the coalition but also needs to be responsible for issuing corresponding service instructions to maintain and control the formed coalition. In order to form alliances in the CVECN architecture, game theory models (e.g., repeated games, random games, etc.) can be used to express such problems [23]–[25].

B. COLLABORATIVE TASK OFFLOADING SCHEME

Based on the discussion in Subsection IV-A, another essential issue is after the formation of the vehicle coalition, how the vehicles in the coalition collaboratively process the computation tasks of the mission-initiated vehicle. In CVECN, when the local computing resources of the RSU cannot meet its current task computational requirements, the RSU will coordinate vehicles within its service range to form a coalition for the mission-initiated vehicle. The mission-initiated vehicle then communicates with the vehicles in the coalition and offloads its computation tasks to them.

As shown in Fig. 4, in the process of the mission-initiated vehicle communication with the collaborative vehicle coalition, the state, topology, and connection characteristics of the coalitions change dynamically. Therefore, when designing the collaborative task offloading scheme, it is necessary to take into account the states of the coalition to make use of computing resources of vehicular coalition effectively and guarantee the QoE of vehicle users. Nowadays, many methods such as Markov decision process, stochastic game and Lyapunov, etc. [26], [27]. can be utilized to formulate collaborative task offloading optimization problem in the time-varying CVECN. And some algorithms, such as evaluate learning algorithms, deep learning algorithms and reinforcement learning algorithms, can be used to obtain the solution of the optimization offloading problem.

In the following, we will briefly introduce an available collaborative computation offloading scheme. Considering that in CVECN, the computation data queue state and the channel link state change dynamically, we adopt the constrained Markov decision process (CMDP) method [28], [29] to model the dynamic computation offloading process. The objective of CMDP formulation is to find the optimal policy, which minimizes the average delay of the task under the constraint of the task loss probability. In the CMDP model, time is logically divided into several intervals, each of which is defined as a decision period. The system state of the proposed CMDP formulation for the task offloading decision of the mission-initiated vehicle includes the mission-initiated vehicle's data queue state (i.e., the number of remaining tasks in its data queue), the vehicles' data queue state and the channel link state between the mission-initiated vehicle and the collaborative vehicles. The channel link is assumed to be independent and identically distributed (i.i.d.) block fading in each time slot, i.e., the channel remains static within each time slot, but varies among different time slots. In each time slot, the mission-initiated vehicle can make an action (i.e., the number of tasks to be locally processed or offloaded to the vehicles) through observing solely on its local data queue state information, the collaborative vehicles' data queue state information, and the channel link state information. According to the adopted action $\boldsymbol{a}$, the old state $\boldsymbol{s}$ transfer to a new state $\boldsymbol{s'}$. And meanwhile, an immediate global cost $g(\boldsymbol{s}, \boldsymbol{a})$ is generated, which can be defined as the average delay of each task at the current time step.

All vehicles in the coalition are collaborative to achieve an optimal stationary control policy $\boldsymbol{\pi}$ to minimize the obtained average processing delay while maintaining the average task loss ratio below the thresholds L^{th}. Accordingly, the optimization problem can be defined as follows:

$$\text{Maximize: } \mathcal{C}(\boldsymbol{\pi}) = \lim_{T\to\infty} \sup \frac{1}{T}\mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=1}^{T} g(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

$$\text{Subject to: } \mathcal{L}(\boldsymbol{\pi}) \le L^{th}$$

where $\mathcal{L}(\boldsymbol{\pi})$ is the average task loss ratio under the control policy $\boldsymbol{\pi}$. Moreover, $\boldsymbol{s}_t$ and $\boldsymbol{a}_t$ are the state and action at the time slot t, respectively. And $\mathbb{E}[\cdot]$ is the expected cost at the long run.

To solve the CMDP formulation problem, we can fist use the Lagrangian approach to relax the original optimization into an unconstrained MDP problem. Then, the Q-learning algorithm can be adopted to find the approximate optimal solution for the relaxed MDP problem.

C. MOBILITY MANAGEMENT

In CVECN, the RSU controls the clusters in the range. The vehicle can perform collaborative processing of tasks in the cluster. This means that at some point, the computation task of one vehicle may exist in other vehicles, or that one vehicle is processing tasks of other vehicles. However, the vehicle is mobile, and the cluster is basically in the RSU range, and the location is relatively fixed. Therefore, when a vehicle moves between different RSUs, how to deal with the remaining computation tasks in the mission-initiated vehicle and cluster is an important issue for mobility management considerations [30].

Application migration is the current solution [31]. Computational tasks are generated by specific applications, so when collaborating, vehicles in the cluster also need to install the corresponding applications to handle these computation tasks. Application migration refers to the fact that vehicle packages and stores unfinished computation tasks and sends them to another vehicle. The vehicle installs the application after receiving the data and then has the ability to continue processing tasks, as illustrated in Fig. 5. There are two cases depending on the role of the vehicle in the cluster:

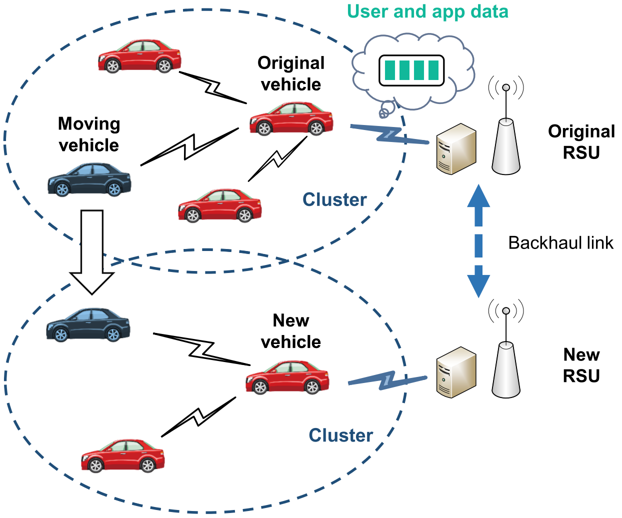

FIGURE 5. Illustration of mobility management in CVECN.

Cluster consumer refers to the mission-initiated vehicle that requires cluster collaboration to handle computation tasks. When the new RSU is not far from the original RSU, communication between the vehicle and the original cluster can be maintained by the backhaul link between the RSUs. If the two RSUs are far apart, the new RSU should join the vehicle to the new cluster. At the same time, application migration will occur between all vehicles in the original cluster and all vehicles in the new cluster. The new cluster will then handle unfinished computation tasks. It is worth noting that the cost of application migration between clusters is high.

Cluster producer refers to the collaborative vehicle that assists in the processing of other vehicle computation tasks in a cluster. When the vehicle enters the new RSU range from the original RSU, it may not have completed the computation tasks of other vehicles. If there are fewer remaining computation tasks, the vehicle can be sent to the cluster over the backhaul link after processing is complete. If there are more remaining computation tasks, the original RSU should select a new vehicle in the cluster. These two vehicles can perform application migration. The new vehicle can continue to process the remaining computation tasks in the cluster.

During the vehicle's movement, the key issue is when to implement application migration. The migration decision mainly depends on the balance between service quality and system energy consumption. The quality of service is mainly related to the task transmission delay. If the cluster consumer gradually moves away from the original cluster, the task transmission delay will increase. The system energy consumption mainly includes the application migration energy consumption, which is related to the size of the application data and the distance between the vehicle and the cluster. To improve the quality of service, application migration needs to be performed more frequently, and this will lead to an increase in system energy consumption. So how to optimize system energy consumption while guaranteeing service quality is the main focus of current research. The solution to this problem is to establish a more scientific application migration model. Analyzing and predicting the moving path of a single vehicle or summarizing the movement rules of a large number of vehicles can significantly improve the efficiency of application migration. Adopting more advanced application management and migration virtualization technology is also an important idea.

D. OTHER RESEARCH ISSUES

Besides the above challenges, there still some other issues that should be taken into account in the implementation of CVECN, which can be summarized as follows.

Security and privacy: Security and privacy are key issues when mobile devices share data (such as video, photos, sensor data, etc.) through the proposed CVECN architecture. To prevent and defend against erroneous data injection and flash attacks from external attackers, edge computing nodes in the CVECN architecture are required to be able to identify and mitigate attacks at the network edge. Regarding the CVECN architecture, there are interactions with various access technologies (such as WiFi, Bluetooth, etc.); different edge applications and multi-tenant infrastructure make the deployment of security and privacy mechanisms a technical challenge problem. As a result, improved data security can be provided because client data is aggregated at certain access points that are close to the mobile user. Furthermore, security features with enhanced robustness should be implemented on the edge computing nodes.

Reliability: Reliability is especially important in the CVECN. The designed information system must implement the specified functions and tasks within the specified time interval and under specified conditions. CVECN required to be able to provide reliable service for mobile devices when the system is destroyed artificially or randomly. When designing the architecture of CVECN system, attention should be paid to avoiding single-point faults in the network, preventing the whole network from being completely paralyzed due to local faults, and ensuring that mobile terminals can switch quickly and transparently.

Standard protocols and interfaces: In the proposed CVECN, different mobile devices and sensors are connected to each other through a communication protocol and communicate with an edge platform. These devices have their own interfaces and therefore require a specific communication protocol. When equipment vendors develop devices in the CVECN environment, standard protocols and interfaces should be developed to support communication between these heterogeneous devices. Due to the rapid development of new devices, developing standard protocols and interfaces in a CVECN environment is challenging.

V. PERFORMANCE EVALUATION

In this section, we conduct some simulation experiments to demonstrate the advantage of our proposed CVECN architecture and collaborative task offloading scheme.

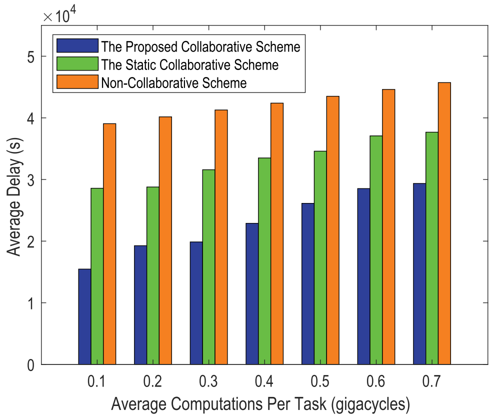

FIGURE 6. The average delay vs. average computations per task.

The specific simulation parameters are set as follows. The number of vehicles in a collaborative coalition is 3. And the number of queue state and channel states are set to 3 and 2, respectively. Meanwhile, the number of queue state are set to 6. For the computation task model, the required number of CPU cycles per task is uniformly selected in the range [50, 100] gcycles while the input data size is randomly distributed between [0.5, 0.7] Gb. Moreover, we assume that the computational capabilities of the vehicles are 1 GHz, and the consumed power per CPU cycle is set to 500 mJ/gcycle. Moreover, the bandwidth, the transmit power between the vehicles, and the background noise power are set to 20 MHz, 0.1 W and 2×10^{-13}, respectively. The decision period is set as 60 s.

For the Q learning-based offloading algorithm, the initial Q vector and the Lagrange multiplier is set at 0, respectively. The learning rate is set to 0.9 and parameter value of greedy algorithm is 0.1. Moreover, the performance results of the adopted Q learning-based offloading algorithm are collected after 500 learning episodes, each of which comprises 2000 steps, i.e., 2000 the number of decision periods.

To better verify the superiority of our proposed scheme, we compare our scheme with two reference schemes, i.e., a static collaborative scheme and a local non-collaborative scheme. We analyze the performance of three offloading schemes in various simulation scenarios, where the average computations per task and the task arrival rate are varied.

In Fig. 6, we verify the considered offloading scheme under different average computations per task. We can observe that as the average computations per task increases, the average delay of the considered schemes increase. This is due to the fact that the increase average computations per task will result in increasing processing delay of each task. Moreover, the delay of the collaborative schemes is significantly lower than that of the non-collaborative scheme, and our proposed collaborative scheme performs better than the static collaborative scheme.

Fig. 7 shows the relationship between the task loss ratio and task arrival rate. When more new tasks arrive, the task loss ratio will increase due to the lack of edge computing node's

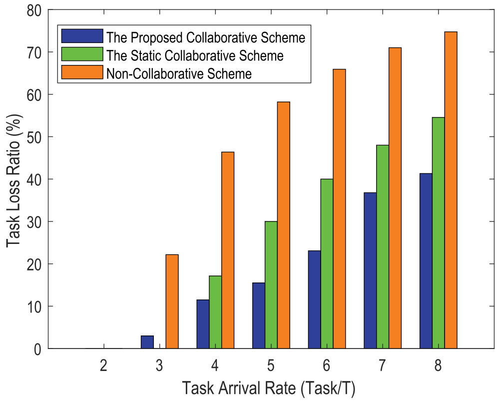

FIGURE 7. **The task loss ratio vs. task arrive rate.**

queue space. Because higher task arrival rates will result in more dropped tasks, which will be removed from the edge compute node's queue. In addition, the proposed scheme can achieve a minimum task loss ratio since it can determine the optimal decision for both local processing and offloading.

VI. CONCLUSION

In this article, we have introduced the collaborative architecture in CVECN, which can further improve the task processing capability and resource utilization of the vehicular network. We first discussed the design principles of CVECN, which is the premise of system operation. Next, we discussed the specific architecture and functional modules of CVECN. Then we introduced the communication process of CVECN in detail, and the implementation idea based on SDN technology was given. After that, the challenges have been presented and illustrated, mainly including edge coalition formation, collaborative scheme, and mobility management. We also sorted out some other research issues that are necessary for the efficient operation of the system. Numerical results have been confirmed that our proposed architecture, which allows computation task collaboration among vehicles, can achieve better performance in an average processing delay and task loss ratio.

ACKNOWLEDGMENT

We thank the reviewers for their detailed reviews and constructive comments, which have helped to improve the quality of this article.

REFERENCES

[1] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014.

[2] Y. Fangchun, W. Shangguang, L. Jinglin, L. Zhihan, and S. Qibo, "An overview of Internet of vehicles," *China Commun.*, vol. 11, no. 10, pp. 1–15, Oct. 2014.

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[4] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[5] 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.

[6] 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.

[7] G. Luo, Q. Yuan, H. Zhou, N. Cheng, Z. Liu, F. Yang, and X. S. Shen, "Cooperative vehicular content distribution in edge computing assisted 5G-VANET," *China Commun.*, vol. 15, no. 7, pp. 1–17, 2018.

[8] M. K. Sharma and A. Kaur, "A survey on vehicular cloud computing and its security," in *Proc. 1st Int. Conf. Next Gener. Comput. Technol. (NGCT)*, Sep. 2015, pp. 67–71.

[9] F. Ahmad, M. Kazim, A. Adnane, and A. Awad, "Vehicular cloud networks: Architecture, applications and security issues," in *Proc. IEEE/ACM 8th Int. Conf. Utility Cloud Comput. (UCC)*, Dec. 2015, pp. 571–576.

[10] H. A. Khattak, S. U. Islam, I. U. Din, and M. Guizani, "Integrating fog computing with VANETs: A consumer perspective," *IEEE Commun. Standards Mag.*, vol. 3, no. 1, pp. 19–25, Mar. 2019.

[11] S. Ko, K. Han, and K. Huang, "Wireless networks for mobile edge computing: Spatial modeling and latency analysis," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5225–5240, Aug. 2018.

[12] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing-a key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[13] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 752–764, Jan. 2017.

[14] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017.

[15] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, Aug. 2017.

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

[17] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1594–1608, Apr. 2018.

[18] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multi-objective optimization for computation offloading in fog computing," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 283–294, Jan. 2018.

[19] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, "Joint radio and computational resource allocation in IoT fog computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7475–7484, Aug. 2018.

[20] Z. Haitao, D. Yi, Z. Mengkang, W. Qin, S. Xinyue, and Z. Hongbo, "Multipath transmission workload balancing optimization scheme based on mobile edge computing in vehicular heterogeneous network," *IEEE Access*, vol. 7, pp. 116047–116055, 2019.

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

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

[23] L. Zhang and B. Cao, "Stochastic programming method for offloading in mobile edge computing based Internet of vehicle," in *Proc. ICC IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[24] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 771–786, Apr. 2019.

[25] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," *J. Mach. Learn. Res.*, vol. 4, pp. 1039–1069, Nov. 2003.

[26] M. Abu Alsheikh, D. T. Hoang, D. Niyato, H.-P. Tan, and S. Lin, "Markov decision processes with applications in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1239–1267, Apr. 2015.

[27] Y. Hong, L. Gao, D. Cheng, and J. Hu, "Lyapunov-based approach to multiagent systems with switching jointly connected interconnection," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 943–948, May 2007.

[28] D. V. Le and C.-K. Tham, "Quality of service aware computation offloading in an Ad-Hoc mobile cloud," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8890–8904, Sep. 2018.
[29] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *Proc. IEEE GLOBECOM*, Singapore, Dec. 2017, pp. 1–6.
[30] 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.
[31] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.

QIUNING WANG received the B.S. degree in communication engineering from the Beijing University of Posts and Telecommunications (BUPT), China, in 2017, where he is currently pursuing the master's degree with the State Key Laboratory of Networking and Switching Technology. His current research interests include mobility management and resource allocation for mobile edge computing.

XU LIU received the B.S. degree in communication engineering from the Beijing University of Posts and Telecommunications (BUPT), China, in 2018, where he is currently pursuing the master's degree with the State Key Laboratory of Networking and Switching Technology. His current research interests include mobile edge computing, computation offloading, edge collaboration, and resource management and allocation.

FEI RICHARD YU is currently a Professor with Carleton University, Canada. His research interests include blockchain, security, and green ICT. He serves on the Editorial Boards of several journals and the Co-Editor-in-Chief for ad hoc and sensor wireless networks. He is also a Lead Series Editor for the IEEE Transactions on Vehicular Technology and the IEEE Communications Surveys and Tutorials. He has served as a Technical Program Committee (TPC) and a Co-Chair of numerous conferences.

HUANG TAO received the B.S. degree in communication engineering from Nankai University, Tianjin, China, in 2002, and the M.S. and Ph.D. degrees in communication and information system from the Beijing University of Posts and Telecommunications, Beijing, China, in 2004 and 2007, respectively. He is currently a Professor with the Beijing University of Posts and Telecommunications. His current research interests include network architecture, routing and forwarding, and network virtualization.

RENCHAO XIE received the Ph.D. degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, in 2012. He is currently an Associate Professor with the Beijing University of Posts and Telecommunications (BUPT). From July 2012 to September 2014, he held a postdoctoral position at China United Network Communications Group Company, Ltd. From November 2010 to November 2011, he visited Carleton University, Ottawa, ON, Canada, as a Visiting Scholar. His current research interests include 5G network and edge computing, information centric networking, and future network architecture. He has published more than 80 journal and conference papers. He has served as a Technical Program Committee (TPC) Member of numerous conferences, including the IEEE Globecom, the IEEE ICC, the EAI Chinacom, and the IEEE VTC-Spring. He has served for several journals and conferences as a Reviewer, including the IEEE Transactions on Communications, *Wireless Networks* (ACM/Springer), IEEE Access, *EURASIP Journal on Wireless Communications and Networking*, and *Wireless Communications and Mobile Computing* (Wiley).

QINQIN TANG received the B.S. degree in information engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2016. She is currently pursuing the Ph.D. degree in network engineering with the State Key Laboratory of Network and Switching Technology, BUPT. Her research interests include mobile edge networks, traffic offloading, and resource management and allocation.

• • •