Energy-Efficient Edge Computing Service Provisioning for Vehicular Networks: A Consensus ADMM Approach

Zhenyu Zhou, *Senior Member, IEEE,* Junhao Feng, Zheng Chang, *Senior Member, IEEE,* Xuemin (Sherman) Shen, *Fellow, IEEE,*

***Abstract*—In vehicular networks, in-vehicle user equipment (UE) with limited battery capacity can achieve opportunistic energy saving by offloading energy-hungry workloads to vehicular edge computing (VEC) nodes via vehicle-to-infrastructure (V2I) links. However, how to determine the optimal portion of workload to be offloaded based on the dynamic states of energy consumption and latency in local computing, data transmission, workload execution and handover, is still an open issue. In this paper, we study the energy-efficient workload offloading problem and propose a low-complexity distributed solution based on consensus alternating direction method of multipliers (ADMM). By incorporating a set of local variables for each UE, the original problem, in which the optimization variables of UEs are coupled together, is transformed into an equivalent general consensus problem with separable objectives and constraints. The consensus problem can be further decomposed into a bunch of subproblems, which are distributed across UEs and solved in parallel simultaneously. Finally, the proposed solution is validated based on a realistic road topology of Beijing, China. Simulation results have demonstrated that significant energy saving gain can be achieved by the proposed algorithm.**

***Index Terms*—vehicular edge computing, energy efficiency, workload offloading, consensus ADMM, vehicular networks.**

I. Introduction

A. Background and Motivation

THE rapid development of vehicular networks will spur an array of applications in the domains of travel assistance, self-driving, video streaming, and online gaming [1]–[4], which require enormous computation resources to process a large volume of workload data and have strict timeliness requirements [5], [6]. To support the delay-sensitive and multimedia-rich services in vehicular networks, vehicular edge computing (VEC), in which workloads are processed at the network edges to eliminate excessive network hops, has been proposed [7]. VEC not only reduces the computation response time, but also alleviates the traffic congestion problem in capacity-constrained backhaul links [8], [9].

Furthermore, VEC allows opportunistic energy saving for in-vehicle user equipments (UEs) with limited battery capacity such as smart phones and wearable devices. Traditionally, all

TABLE I
NOMENCLATURE

Valuables	Definitions
M	Number of RSUs (VEC nodes)
K	Number of vehicles (in-vehicle UEs)
θ_k	Workload data size of UE U_k
δ	Required computation resource per workload
η	Required computation resource for result
τ_k	The latency constraint of workload execution for UE U_k
p_k^o	Workload offloading portion of UE U_k
λ_k	Average workload arrival rate of UE U_k
τ_k^o	Dwell time of vehicle V_k inside the coverage of RSU R_m
d_k	Distance between the location of V_k and the coverage edge of RSU R_m in the vehicle heading direction
$\bar{v}_k$	Average velocity of V_k
d_m	The coverage diameter of RSU R_m
u_k^l	Local computing capability of UE U_k
S_k^l	Occupancy rate of CPU resources for UE U_k
T_k^l	Local computing latency of UE U_k
β_k	Power consumption of local workload processing for U_k
E_k^l	Energy consumption of local workload processing for U_k
λ_m^e	Sum workload arrival rate of all UEs at VEC node I_m
E_k^t	Energy consumption of in-vehicle UE U_k
T_k^t	Workload transmission latency of UE U_k
T_m^e	Average waiting latency of each workload
T_m^t	Average waiting latency of each computation result
T_k^h	Handover latency of UE U_k
E_k^{total}	Total energy consumption of UE U_k
κ	Time required to deliver the TPBU message
T_{L2}	Time required to deliver the L2 report
$T_{k,PT}^{'}$ ($T_{k,PT}$)	Time required for the sMAG (nMAG) to send the data packets to the nMAG (RSU)
φ	Time required to deliver the HI message
ϖ_k	Time for confirming the received profile and creating a new cache entry

Manuscript received December 8, 2018; revised January 2, 2019; accepted February 21, 2019. Date of publication XXX XX, 2019; date of current version March 11, 2019. This work was partially supported by the National Natural Science Foundation of China (NSFC) under Grant Number 61601181; Fundamental Research Funds for the Central Universities under Grant 2017MS001. *(Corresponding author: Junhao Feng)*

Z. Zhou and J. Feng 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, China. (E-mail: zhenyu_zhou@ncepu.edu.cn, junhao_feng@ncepu.edu.cn).

Z. Chang is with the Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland. (E-mail: zheng.chang@jyu.fi).

X. (S.) Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada (e-mail: xshen@bbcr.uwaterloo.ca).

Part of this work was presented in the 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW) at Barcelona, Spain.

of the workloads have to be processed locally on the UE, which dramatically reduces the battery endurance time and impedes the service delivery reliability. With the assistance of VEC, the energy-hungry workloads can be offloaded from the UE to nearby VEC nodes with higher computing capability and abundant energy supply via vehicle-to-infrastructure (V2I) links [10]. As a result, the energy expenditure of local computing is saved at the costs of increased latency caused by workload offloading and the additional energy consumption for transmitting the computation workload [11].

There exist some works that have tried to improve energy efficiency of UEs via workload offloading [12]–[15]. You *et al.* studied resource allocation problems under the computation latency constraint for MEC offloading systems in order to minimize the weighted sum energy consumption for mobile UEs [14]. In [15], Li *et al.* introduced MEC into virtualized cellular networks with machine-to-machine communications, where each UE chooses to access virtual networks so as to minimize the energy consumption and execution time. However, some critical challenges have been neglected in previous studies, which are summarized as follows.

First, workload offloading may not always lower energy consumption due to communication costs. To minimize the energy consumption, the tradeoff between energy saving of workload offloading and energy consumption of communication should be optimized dynamically based on a number of factors including channel conditions, workload attributes, vehicle velocity, computing capability, etc., which has not been thoroughly analyzed from the perspective of energy efficiency [12]. Second, the offloading decisions of adjacent UEs are often intertwined with each other via the constraint term of VEC node's computing capability, and the size of the joint optimization problem grows rapidly with the number of UEs. Centralized optimization approaches proposed in [13] faces severe complexity and scalability problems. Last but not least, the intermittent connectivity between vehicles and road side units (RSUs) poses another critical challenge. A vehicle that have moved out of the RSU coverage during workload data transmission will result in frequent offloading failures, which is not considered in previous works [12]–[15].

B. Contributions

In this paper, we investigate how to address the above challenges by exploring consensus alternating direction method of multipliers (ADMM), which is a powerful tool for solving distributed convex optimization problems. It takes a decomposition-coordination procedure, in which the joint optimization problem is firstly decomposed into several tractable subproblems that can be solved in parallel, and then the solutions of all the subproblems are coordinated to obtain the global solution of the original problem [16]. The main contributions of this work are summarized as follows.

- We introduce queuing theory to derive the stochastic traffic models at both UEs and VEC nodes with the consideration of queue heterogeneity. By assuming that the generated workload follows a Poisson process and the service time follows an exponential distribution, the workload traffic models of the UE and the VEC node can be regarded as a $M/M/1$ queue and a $M/M/c$ queue, respectively. Then, the closed-form expressions of computation latency and waiting latency are derived based on Little's law and Erlang's formula.
- An energy-efficient workload offloading problem is formulated to minimize the total energy consumption of all the UEs, with the explicit considerations of the overall energy consumption and latency, including local computing latency and energy consumption, data transmission latency and energy consumption, waiting latency, and handover latency. The formulated problem is NP-hard due to the fractional form of the objective function and that the constraint term of VEC node computing capability couples all the optimization variables.
- We propose a consensus ADMM-based distributed solution, which has less signalling overhead, higher scalability and better flexibility compared to the conventional centralized approach. First, the coupling among optimization variables is decoupled properly by incorporating a set of local variables, which represent the local copies of the same global variables at each UE. Then, the original problem with coupled variables is transformed into an equivalent general consensus problem with separable objectives. Next, the transformed problem is further decomposed into a bunch of subproblems, which are distributed across UEs and solved in parallel.
- A real-world topology based simulation is conducted to validate the proposed algorithm. The relationships between energy consumption and other key parameters, including workload offloading portion, transmission power, RSU coverage radius, and number of UEs are illustrated through numerical results.

The remaining parts of this paper are organized as follows. A review of related works is presented in Section II. Section III describes the system model. The problem formulation is presented in Section IV. The consensus ADMM-based distributed algorithm is proposed in Section V. Simulation results and related analysis are elaborated in Section VI. Conclusions and future directions are summarized in Section VII.

II. Related works

Mobile edge computing (MEC) is regarded as a promising solution to achieve the performance gain of proximate data processing, short-range transmission, and location awareness [17]. There have been many works investigating MEC in vehicular networks. Feng *et al.* proposed a VEC framework named autonomous vehicular edge (AVE) to increase the computational capabilities of vehicles in a decentralized manner [7]. In [10], Zhang *et al.* designed an offloading scheme to improve the transmission efficiency with considerations of the task execution time and the vehicle mobility. In [18], Taleb *et al.* developed a cloud-based MEC offloading framework and proposed a predictive computation mode transfer scheme to improve task transmission efficiency in vehicular networks. These works mainly focus on low-latency and high-reliability system design, and have not considered the energy saving problems for in-vehicle UEs with limited battery capacity.

There are many studies that investigate the energy efficiency issue in edge computing through workload offloading and system resource allocation. In [12], the workload allocation between fog and cloud is optimized to minimize the system energy consumption under different service delay constraints. In [13], Mao *et al.* proposed an effective computation offloading strategy to construct a green MEC system with energy harvesting devices.

Nevertheless, the above-mentioned works mainly target on static cellular networks, and thus cannot be applied directly for vehicular networks with highly dynamic and unreliable connections. Without considering the fast mobility of vehicles, conventional static decision-making schemes will result in frequent offloading failures when the connectivity between vehicles and the RSU becomes unavailable before the workload data has been fully uploaded. Although there exist some works which have applied MEC for vehicular networks [7], [10], [18], they mainly address the workload offloading problem from a delay minimization perspective, and have not considered the energy efficiency issues of in-vehicle UEs with limited battery capacity. Their results cannot be directly utilized to solve the energy-efficient workload offloading problem investigated in this work. Moreover, most of the previous solutions rely on a centralized optimization approach, the computing complexity of which increases significantly with the number of UEs. It is better to address the problem from a distributed perspective considering the complexity and scalability issues. Therefore, there lacks a unified distributed solution to address the energy saving problems for in-vehicle UEs with the considerations of vehicle mobility.

We next review the related studies about ADMM, which is used to solve the formulation of the joint optimization in this work. ADMM, which is known as a powerful tool for solving distributed convex optimization problems [16], has been widely applied in many aspects. Yin *et al.* considered a fog-assisted data streaming scenario [19], and proposed a hybrid ADMM (H-ADMM) method to solve the social welfare optimization problem and reduce the communication overhead. In [20], Vu *et al.* investigated the energy efficiency optimization problem of small-cell networks with multi-antenna transceivers and base stations. By using Charnes-Cooper's transformation, the original optimization problem was transformed into an equivalent convex program, and an ADMM-based decentralized algorithm was presented to solve the problem and achieve a fast convergence.

This work is an extension of our previous work [1]. Different from the previous studies, we employ the consensus ADMM approach to address the energy saving problem. The differences between ADMM and the consensus ADMM are summarized as follows. In ADMM, the primal variables are updated in an alternating or sequential fashion, which can be regarded as a modified version of the conventional method of multipliers based on the Gauss-Seidel approach [16]. On the other hand, the consensus ADMM employs a series of local variables, based on which the primal variables no longer need to be updated sequentially. Instead, the coupled objectives and constraints of the joint optimization problem can be separated and distributed across UEs, where each UE only has to deal with its own objective and constraint term. In other words, the consensus ADMM is actually the extension of ADMM for solving the consensus problems, which aims to achieve a consensus between the local variables and the global variables in a dynamically changing environment [21]. Hence, the consensus ADMM approach not only reduces the amount of information that needs to be exchanged, but also enables parallel decision making [22]. This is of significant importance for vehicular networks with fast mobility, capacity-constrained communication links, and strict timeliness requirements. Convergence to the global solution is guaranteed as long as the convergence requirements can be satisfied [23]. Furthermore, a more realistic handover model based on the IFP-NEMO is utilized, which takes into account both the link reestablishment and the data forwarding latency. Last but not least, we provide a comprehensive analysis regarding the convergence and complexity properties. We also validate the proposed scheme by using a real-world road topology.

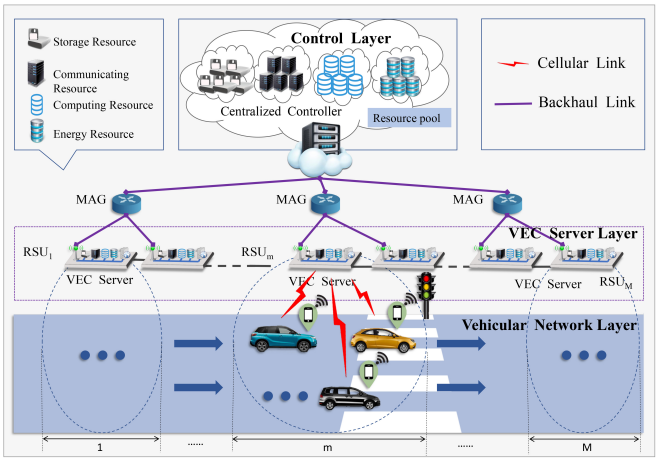


Fig. 1. The three-layer hierarchical architecture of VEC.

III. System Model

In this section, we elaborate the overall system model of VEC, the data transmission model, and the computation workload offloading model in details.

A. The Overall System Model

The hierarchical computing framework for vehicular networks is shown in Fig. 1, which is composed of three layers, i.e., the control layer, the VEC server layer, and the vehicular network layer. In the control layer, a centralized controller is responsible for the inter-cell resource coordination and handover management [24]. To maintain Internet connectivity for moving vehicles, the improved fast proxy mobile IPv6 based network mobility basic support (IFP-NEMO) mechanism is adopted [25]. In IFP-NEMO, the mobility management of vehicles is performed by a mobile access gateway (MAG), which acts as a proxy mobility agent. In the distributed VEC server layer, M RSUs are deployed uniformly along an unidirectional lane and connected to the MAGs via Ethernet

connections. For each RSU, there exists a co-located VEC node with c homogeneous servers. The m-th RSU and the co-located VEC node are denoted as R_m and I_m, respectively.

In the vehicular network layer, we can divide the road into corresponding M segments based on the coverage areas of M RSUs, e.g., segment m corresponds to the coverage of RSU R_m. We assume that there exist K vehicles in the m-th segment traveling towards the same direction, which is shown in Fig. 1. The k-th vehicle is denoted as V_k. The communication device mounted on each vehicle has two-folded functions. On the one hand, it allows the vehicle to transmit data to the RSU and offload workloads to the VEC node via dedicated V2I links. On the other hand, it acts as an access point and provides free connections for in-vehicle UEs via short-range communication technologies such as Wi-Fi [26]. The UE inside vehicle V_k is denoted as U_k. The set of in-vehicle UEs is defined as $\mathcal{U} = \{U_1, \cdots, U_k, \cdots, U_K\}$.

For any UE $U_k \in \mathcal{U}$, an array of applications are executed, which accordingly generate a series of computation workloads. Without loss of generality, the workload generated at UE U_k is assumed to follow a Poisson process with an average arrival rate λ_k [27]–[29], which can be either processed locally by the UE itself or offloaded to VEC node I_m. The key attributes of the workloads generated by UE U_k can be described by a triplet $\{\theta_k, \delta_k, \tau_k\}$, where θ_k represents the data size of workloads, δ_k is the required computation resource for processing the workloads, and τ_k represents the delay constraint. We assume that each workload has the same computation complexity, which is defined as δ. This assumption is valid since a higher complexity workload is equivalent to several several basic workloads with the same computation complexity. Thus, we have $\delta_k = \delta\lambda_k$. By further assuming that the service time follows an exponential distribution, the workload traffic models of UE U_k and VEC node I_m can be regarded as a $M/M/1$ queue and a $M/M/c$ queue, respectively.

The workload offloading and execution are implemented as the following three steps: (i.) each UE $U_k \in \mathcal{U}$ determines the portion of workload offloaded to VEC node I_m, i.e., $0 \leq p_k^o \leq 1$, and transmits the workload related data to RSU R_m; (ii.) the offloaded workload is processed at the VEC node; (iii.) the obtained computation results are fed back to UE U_k.

Remark 1. In this work, we only consider the simplified single-segment case in order to derive a tractable solution. The more complicated multi-segment case is beyond the scope of this paper and will be investigated in future works. Nevertheless, the proposed solution can be easily extended to the multi-segment scenario by adopting a time-slot model. That is, the number of vehicles in each segment remains constant within a slot and varies across different slots. Hence, the proposed solution can be applied for the optimization of workload offloading within each segment in a slot-by-slot fashion.

Remark 2. A justification for the $M/M/1$ and $M/M/c$ queuing models is that the same traffic and service time models have been adopted in a number of previous works such as [27]–[29]. Moreover, the solution structure does not depend on the specific traffic models. The proposed solution can be extended to other traffic models.

B. The Transmission Model

We assume that each vehicle is allocated with an orthogonal spectrum resource block so that the co-channel interference among vehicles can be ignored. In the offloading mode, data are actually transmitted from UE U_k to RSU R_m in a two-hop fashion, i.e., data are firstly sent from UE U_k to vehicle V_k in the first hop, and then are forwarded from vehicle V_k to RSU R_m in the second hop. The signal to noise ratio (SNR) expressions of the first-hop link and the second-hop link are calculated as

$$\gamma_k^U = \frac{P_k^U g_k^U}{N_0}, \tag{1}$$

$$\gamma_k^V = \frac{P_k^V g_k^V}{N_0}, \tag{2}$$

where P_k^U and P_k^V are the transmission power of UE U_k and vehicle V_k, respectively. g_k^U and g_k^V are the channel gain between U_k and V_k, and the channel gain between V_k and RSU R_m, respectively. N_0 is the additive white Gaussian noise (AWGN).

The effective SNR of the two-hop link, i.e., ($U_k \rightarrow V_k \rightarrow$ RSU R_m) [30], is expressed as

$$\gamma_k = \frac{\gamma_k^U \gamma_k^V}{\gamma_k^U + \gamma_k^V + 1}. \tag{3}$$

Hence, the transmission time required by UE U_k for uploading workload data with size $p_k^o\theta_k$, i.e., T_k^t, can be obtained as

$$T_k^t(p_k^o) = \frac{p_k^o \theta_k}{B_k \log_2(1+\gamma_k)}, \tag{4}$$

where B_k refers to the channel bandwidth.

Due to the fast vehicle mobility, vehicle V_k might move out of the communication range of RSU R_m during data transmission, which results in an offloading failure. Denote the dwell time of V_k inside the coverage of RSU R_m as τ_k^o. An offloading failure occurs if $\tau_k^o < T_k^t$. Therefore, τ_k^o also represents the delay constraint of data transmission because V_k can only transmit data to RSU R_m when it remains within segment m. That is, an offloading request is admissible if and only if $T_k^t \leq \tau_k^o$. τ_k^o can be calculated as

$$\tau_k^o = d_k / \bar{v}_k, \tag{5}$$

where d_k denotes the distance between the location of V_k and the coverage edge of RSU R_m in the vehicle heading direction, and $\bar{v}_k$ denotes the average velocity of V_k within segment m.

Remark 3. Both d_k and $\bar{v}_k$ can be estimated from the GPS data [31], which are generally available for latest vehicles. For example, if V_k moves in the centrifugal direction to leave the coverage area of RSU R_m with radius d_m, d_k is calculated as $d_k = d_m - d_{k,m}$, where $d_{k,m}$ is the distance between V_k and RSU R_m. Otherwise, if V_k moves in the centripetal direction, we have $d_k = d_m + d_{k,m}$.

The energy consumed for transmitting the workload data to the in-vehicle access point is calculated as

$$E_k^t(p_k^o) = P_k^U T_k^t(p_k^o) = \frac{P_k^U p_k^o \theta_k}{B_k \log_2(1+\gamma_k)}. \tag{6}$$

C. The Computation-Offloading Model

Based on the Poisson splitting property [32], if the workload of UE U_k follows a Poisson process with an average rate λ_k, then the workload that is processed locally on UE U_k follows a Poisson process with an average rate $(1-p_k^o)\lambda_k$. Furthermore, the workload offloaded from UE U_k to VEC node I_m also follows a Poisson process with an average rate $p_k^o\lambda_k$. Next, by using Little's law, the local computing latency T_k^l of UE U_k is calculated as

$$T_k^l(p_k^o) = \frac{1}{\frac{u_k^l}{\delta}(1-S_k^l) - \lambda_k(1-p_k^o)}, \tag{7}$$

where u_k^l is the local computing capability of UE U_k. S_k^l denotes the normalized workload of other on-going applications, which reflects the occupancy rate of CPU resources, i.e., $0 \le S_k^l \le 1$. For example, $S_k^l = 1$ represents that the CPU is completely occupied by other applications.

The energy consumption of local workload execution is given by

$$E_k^l(p_k^o) = \beta_k T_k^l(p_k^o) = \frac{\beta_k}{\frac{u_k^l}{\delta}(1-S_k^l) - \lambda_k(1-p_k^o)}, \tag{8}$$

where β_k represents the local power consumption per unit workload execution.

The energy consumption of UE U_k, which contains the energy consumed for local workload execution and workload data uploading, is expressed as

$$E_k^{total}(p_k^o) = E_k^l(p_k^o) + E_k^t(p_k^o). \tag{9}$$

Taking (6) and (8) into (9), the expression of $E_k^{total}(p_k^o)$ is written as (10).

Remark 4. u_k^l, β_k and S_k^l depend on the intrinsic nature of CPU, workload complexity, and other ongoing applications. To simplify the problem, the values of u_k^l, β_k and S_k^l are assumed as constants during the decision making process, and may vary across different decision making processes. It is noted that the values of u_k^l, β_k and S_k^l are privacy information of UE U_k, which are generally unknown for VEC node I_m. Hence, conventional centralized optimization algorithms which require perfect knowledge of UE's private information cannot be directly applied.

Due to the limited computation resources, the VEC node cannot execute a massive number of workloads simultaneously. In VEC node I_m, the workloads offloaded from different UEs are pooled together and wait to be processed by VEC servers. Since the combination of independent Poisson processes is also Poisson [32], the sum rate λ_m^e is calculated as

$$\lambda_m^e = \sum_{U_k \in \mathcal{U}} p_k^o \lambda_k. \tag{11}$$

Considering the c homogeneous servers deployed in VEC node I_m, the computing capability of each server is defined as u_m^e. Based on the $M/M/c$ queuing model and Erlang's formula [33], the average waiting latency of each workload at VEC node I_m can be calculated as

$$T_m^e(p_k^o) = \frac{\varphi(c, \rho_m^e)}{\frac{cu_m^e}{\delta} - \lambda_m^e} + \frac{\delta}{u_m^e}, \tag{12}$$

where ρ_m^e is the server occupancy, and $\varphi(c, \rho_m^e)$ is the Erlang C formula which represents the waiting probability. ρ_m^e and $\varphi(c, \rho_m^e)$ are calculated as

$$\rho_m^e = \frac{\lambda_m^e \delta}{cu_m^e}, \tag{13}$$

$$\varphi(c, \rho_m^e) = \frac{\frac{(c\rho_m^e)^c}{c!(1-\rho_m^e)}}{\sum_{l=0}^{c-1} \frac{(c\rho_m^e)^l}{l!} + \frac{(c\rho_m^e)^c}{c!(1-\rho_m^e)}}. \tag{14}$$

In RSU R_m, the computation results also have to wait in a queue before they can be processed and delivered back to UE U_k. Hence, the average waiting latency of each computation result at RSU R_m, i. e. , $T_m^t(p_k^o)$, can be expressed as

$$T_m^t(p_k^o) = \frac{1}{\frac{u_m^t}{\eta} - \lambda_m^e}, \tag{15}$$

where u_m^t denotes the transmission processing rate of RSU R_m, and η denotes the computation resouce required to process each result. The transmission latency from RSU R_m to U_k is ignored, due to the fact that the size of computation results is usually negligible compared to that of the input data.

If vehicle V_k has already moved out of the coverage of RSU R_m when the results are ready for transmission, i.e., a handover occurs when $T_k^t + T_m^e + T_m^t > \tau_k^o$, then the results have to be forwarded firstly from the serving MAG (sMAG) to the centralized controller, and then sent from the centralized controller to the next MAG (nMAG) with which V_k will be attached. The handover process of the IFP-NEMO is carried out from two perspectives in parallel: link reestablishment and data forwarding [25]. The procedure of link reestablishment is illustrated as follows.

- **Step 1:** The previous wireless layer 2 (L2) link between R_m and V_K is disconnected, which requires a time of T_{off}.
- **Step 2:** A new L2 link between RSU $R_{m'}$ ($R_{m'} \neq R_m$), with which V_k is reconnected, and V_k is established, which takes a time of T_{on}.

Hence, the total latency of link reestablishment is calculated as

$$T_{k,link} = T_{off} + T_{on}. \tag{16}$$

The procedure of data forwarding is illustrated as follows:

- **Step 1:** The predictive mode of IFP-NEMO is activated when V_k sends a L2 report to the sMAG. The time required to deliver the L2 report is denoted as t_{L2}.
- **Step 2:** Upon receiving the L2 report, the sMAG sends a handover initiate (HI) message to the nMAG, which contains a number of key information including vehicle ID, home network prefix, mobile network prefix, and centralized controller address. The time required to deliver the HI message is denoted as φ.

$$E_k^{total}(p_k^o) = \frac{\beta_k}{\frac{u_k^l}{\delta}\left(1-S_k^l\right)-\lambda_k\left(1-p_k^o\right)} + P_k^U T_k^t(p_k^o) = \frac{\overbrace{\beta_k B_k \log_2(1+\gamma_k) + P_k^U p_k^o \theta_k \left[\frac{u_k^l}{\delta}\left(1-S_k^l\right)-\lambda_k\left(1-p_k^o\right)\right]}^{F_{k,1}(p_k^o)}}{\underbrace{\left[\frac{u_k^l}{\delta}\left(1-S_k^l\right)-\lambda_k\left(1-p_k^o\right)\right] B_k \log_2(1+\gamma_k)}_{F_{k,2}(p_k^o)}}. \quad (10)$$

- **Step 3:** Upon receiving the HI message, the nMAG confirms the received profile of V_k and creates a new cache entry, which takes a time of ϖ_k.
- **Step 4:** The nMAG sends a tentative proxy binding update (TPBU) message to the centralized controller. The time required to deliver the TPBU message is denoted as κ.
- **Step 5:** The centralized controller confirms the received profile of V_k and creates a new cache entry, which takes a time of ϖ_k.
- **Step 6:** The sMAG sends the data packets to the nMAG via the centralized controller, which takes a time of $T_{k,PT}^{'}$.

The total time required for data forwarding is calculated as

$$T_{k,data} = T_{L2} + \varphi + 2\varpi_k + \kappa + T_{k,PT}^{'}. \quad (17)$$

Once both the link reestablishment and the data forwarding processes are completed, the nMAG sends the data packet to RSU $R_{m'}$ ($R_{m'} \neq R_m$), which takes a time of $T_{k,PT}$. The handover latency T_k^h is defined as the total duration during which V_k cannot send or receive any data packet due to either link reestablishment latency or data forwarding latency. To calculate the handover latency, the following four cases are considered, which are shown in Fig. 2.

- **Case A** ($T_{L2} + \varphi + 2\varpi_k + \kappa > T_{off}$ **and** $T_{k,data} > T_{k,link}$)**:** If $T_{k,data} > T_{k,link}$, the link reestablishment process is finished earlier than the data forwarding process. Therefore, the data can be directly sent to V_k from the nMAG without buffering. Furthermore, since $T_{L2} + \varphi + 2\varpi_k + \kappa > T_{off}$, the handover latency should be calculated from the moment that the previous L2 link has been disconnected, which is given by

$$T_{k,A}^h = T_{L2} + \varphi + 2\varpi_k + \kappa - T_{off} + T_{k,PT}^{'} + T_{k,PT}. \quad (18)$$

- **Case B** ($T_{L2} + \varphi + 2\varpi_k + \kappa < T_{off}$ **and** $T_{k,data} > T_{k,link}$)**:** In this case, the sMAG starts to transfer data to the nMAG even though the L2 link has not been disconnected. Therefore, the handover latency is calculated from the moment when the sMAG starts to transfer data packets to the nMAG. T_k^h is calculated as

$$T_{k,B}^h = T_{k,PT}^{'} + T_{k,PT}. \quad (19)$$

- **Case C** ($T_{L2} + \varphi + 2\varpi_k + \kappa > T_{off}$ **and** $T_{k,data} < T_{k,link}$)**:** Since $T_{k,data} < T_{k,link}$, V_k has not reconnected with the nMAG when the data forwarding process is completed, and the delivered data have to be buffered in nMAG. T_k^h is calculated as

$$T_{k,C}^h = T_{on} + T_{k,PT}. \quad (20)$$

- **Case D** ($T_{L2} + \varphi + 2\varpi_k + \kappa < T_{off}$ **and** $T_{k,data} < T_{k,link}$)**:** This case is similar to case B. The only difference is that the data forwarding process is finished earlier, and the nMAG has to wait for the link reestablishment process to be finished. Therefore, T_k^h is calculated as

$$T_{k,D}^h = T_{off} - (T_{L2} + \varphi + 2\varpi_k + \kappa) + T_{on} + T_{k,PT}. \quad (21)$$

A robust approach is to consider the worst-case scenario, i.e., $T_k^h = \max\{T_{k,A}^h, T_{k,B}^h, T_{k,C}^h, T_{k,D}^h\}$. Hence, the latency caused by workload offloading is the sum of the workload transmission latency, the waiting latency at VEC node I_m, the remote workload execution latency, the waiting latency at RSU R_m, and the handover latency, which is given by

$$T_k^o(p_k^o) = T_k^t(p_k^o) + T_m^e(p_k^o) + T_m^t(p_k^o) + T_k^h(p_k^o). \quad (22)$$

IV. Problem Formulation

The objective is to minimize the total energy consumption of K UEs within the coverage of RSU R_m. The formulated energy-efficient workload offloading problem is given as follows:

$$\begin{aligned}
\mathbf{P1}: & \min_{\{p_k^o\}} \sum_{U_k \in \mathcal{U}} E_k^{total}(p_k^o) \\
\text{s.t.} \quad & C_1: \lambda_k(1-p_k^o) \leq \frac{u_k^l}{\delta}(1-S_k^l), \forall U_k \in \mathcal{U}, \\
& C_2: \sum_{U_k \in \mathcal{U}} p_k^o \lambda_k \leq \frac{c u_m^e}{\delta}, \\
& C_3: T_k^t \leq \tau_k^o, \forall U_k \in \mathcal{U}, \\
& C_4: T_k^l \leq \tau_k, \forall U_k \in \mathcal{U}, \\
& C_5: T_k^o \leq \tau_k, \forall U_k \in \mathcal{U}, \\
& C_6: p_k^o \in [0,1], \forall U_k \in \mathcal{U}. \quad (23)
\end{aligned}$$

Here, C_1 and C_2 represent the computing capability constraints, i.e., the workload arrival rates $\lambda_k(1-p_k^o)$ and $\sum_{U_k \in \mathcal{U}} p_k^o$ should not exceed the processing rate at UE U_k and VEC node I_m, respectively. C_3 denotes latency constraint of data transmission. C_4 and C_5 denote the latency constraints of local and remote workload executions, respectively. C_6 is the boundary constraint of p_k^o.

It is infeasible to find a polynomial-time solution for $\mathbf{P1}$ due to the following two reasons. First, the objective function

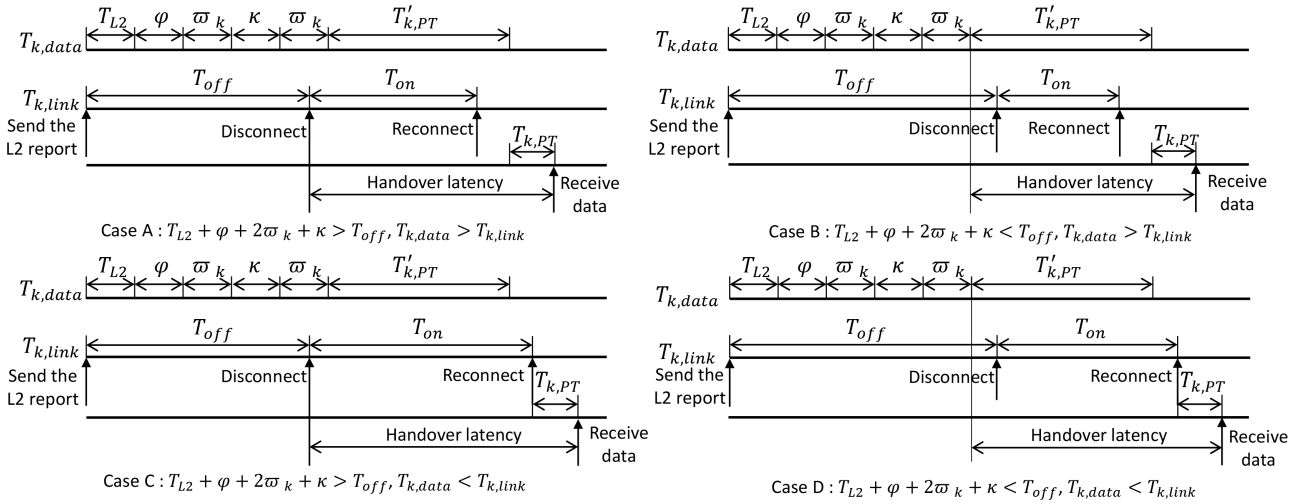


Fig. 2. Illustration of four handover cases.

is not convex. Second, the optimization variables of K UEs are coupled through the term C_2. Furthermore, it is noted that the problem size of $\mathbf{P1}$ grows enormously fast with the number of UEs. Therefore, it is difficult to solve $\mathbf{P1}$ via centralized solutions because the VEC node or the RSU has to collect every detailed piece of information from all of UEs. This might be infeasible for practical implementation considering the communication overhead constraint and the threat of privacy leakage. Hence, we aim at addressing $\mathbf{P1}$ in a distributed manner.

V. Consensus ADMM-based Energy-efficient Workload Offloading

In this section, we introduce an energy-efficient distributed solution based on consensus ADMM. First, we provide a brief introduction to consensus ADMM for the readers' better understanding. Then, we introduce the problem transformation which is a prerequisite for applying consensus ADMM. Next, the implementation procedures of the proposed distributed solution are elaborated. Finally, we analyze the convergence and complexity properties.

A. Introduction to Consensus ADMM

Generally, ADMM is suitable to solve the problems with the following forms [23]:

$$\begin{aligned} \mathbf{P1}: \min_{\mathbf{x},\mathbf{y}} \ & f(\mathbf{x})+g(\mathbf{y}) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}=\mathbf{c}, \end{aligned} \tag{24}$$

where $\mathbf{x}\in\mathbf{R}^{q_1\times 1}$, $\mathbf{y}\in\mathbf{R}^{q_2\times 1}$, $\mathbf{A}\in\mathbf{R}^{q_3\times q_1}$, $\mathbf{B}\in\mathbf{R}^{q_3\times q_2}$, and $\mathbf{c}\in\mathbf{R}^{q_3\times 1}$. The augmented Lagrangian of (24) is given by

$$\begin{aligned} &L_\rho(\mathbf{x},\mathbf{y},\boldsymbol{\mu}) \\ &= f(\mathbf{x})+g(\mathbf{y})+\boldsymbol{\mu}^T(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c})+\frac{\rho}{2}\parallel \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c}\parallel_2^2, \end{aligned} \tag{25}$$

where $\rho\in R_{++}$ denotes the penalty parameter in the augmented Lagrangian, which is used to increase the speed of convergence in ADMM [27]. ρ can be adjusted by using the self-adaptive approach [16]. $\boldsymbol{\mu}$ denotes the vector of Lagrange multipliers.

The problem (24) can be solved via the following iterations:

$$\mathbf{x}[t+1]=\arg\min_{\mathbf{x}} L_\rho(\mathbf{x},\mathbf{y}[t],\boldsymbol{\mu}[t]), \tag{26}$$

$$\mathbf{y}[t+1]=\arg\min_{\mathbf{y}} L_\rho(\mathbf{x}[t+1],\mathbf{y},\boldsymbol{\mu}[t]), \tag{27}$$

$$\boldsymbol{\mu}[t+1]=\boldsymbol{\mu}[t]+\rho(\mathbf{A}\mathbf{x}[t+1]+\mathbf{B}\mathbf{y}[t+1]-c), \tag{28}$$

where t is the index of iteration.

Next, we consider a global consensus problem with a global variable vector $\mathbf{z}$, i.e., $\mathbf{z}\in\mathbf{R}^{q_1\times 1}$, and several local variable vectors $\mathbf{x}_i$, i.e., $\mathbf{x}_i\in\mathbf{R}^{q_1\times 1}$, $i=1,\cdots,N$, which is formulated as [22]:

$$\begin{aligned} &\min_{\mathbf{x}_i} \sum_{i=1}^{N} f_i(\mathbf{x}_i) \\ &\text{s.t.} \quad \mathbf{x}_i-\mathbf{z}=0, i=1,\ldots,N. \end{aligned} \tag{29}$$

The consensus constraint guarantees that all of the local variables should be equal to the global variable. The augmented Lagrangian corresponding to (29) is given by

$$\begin{aligned} &\mathcal{L}_\rho(\mathbf{x}_1,\ldots,\mathbf{x}_N,\boldsymbol{\mu},\mathbf{z}) \\ &\quad=\sum_{i}^{N} f_i(\mathbf{x}_i)+(\boldsymbol{\mu}_i)^T(\mathbf{x}_i-\mathbf{z})+\frac{\rho}{2}\parallel \mathbf{x}_i-\mathbf{z}\parallel_2^2 \end{aligned} \tag{30}$$

The resulting iterations are given given by

$$\begin{aligned} \mathbf{x}_i[t+1]=\arg\min_{\mathbf{x}_i}\Big\{ & f_i(\mathbf{x}_i)+(\boldsymbol{\mu}_i[t])^T(\mathbf{x}_i-\mathbf{z}[t]) \\ & +\frac{\rho}{2}\parallel \mathbf{x}_i-\mathbf{z}[t]\parallel_2^2 \Big\} \end{aligned} \tag{31}$$

$$\mathbf{z}[t+1] = \arg\min_{\mathbf{z}} \sum_{i}^{I} \Big\{ (\boldsymbol{\mu}_i[t])^T(-\mathbf{z}) + \frac{\rho}{2} \parallel \mathbf{x}_i[t+1] - \mathbf{z} \parallel_2^2 \Big\} \quad (32)$$

$$\boldsymbol{\mu}_i[t+1] = \boldsymbol{\mu}_i[t] + \rho(\mathbf{x}_i[t+1] - \mathbf{z}[t+1]). \quad (33)$$

Remark 5. It is noted that the $\mathbf{x}$-minimization and $\mathbf{y}$-minimization steps in (26) and (27) are carried out in a sequential fashion, while the $\mathbf{x}_i$-minimization in (31) is carried out in parallel for each $i = 1, \cdots, N$.

B. Problem Transformation

To apply ADMM, we have to transform problem $\mathbf{P1}$ into a tractable form. First, the original problem with a fractional-form is transformed to a new problem with a subtractive-form objective function. Second, the problem with coupled variables is further transformed to a decomposable problem with separable objectives and decoupled variables. The details are illustrated as follows.

1) Nonlinear Fractional Programming: It can be observed from (10) that $E_k^{total}(p_k^o)$ is a fractional-form function. Hence, we can employ nonlinear fractional programming to transform the original problem in the fractional form into an equivalent problem in the subtractive form.

Let us define the numerator and denominator of (10) as $F_{k,1}(p_k^o)$ and $F_{k,2}(p_k^o)$, respectively. Denote ψ_k^* as the maximum value of $E_k^{total}(p_k^o)$, which is expressed as

$$\begin{aligned} \psi_k^* &= \min_{\{p_k^o\}} E_k^{total}(p_k^o) \quad (34) \\ &= \min_{\{p_k^o\}} \frac{F_{k,1}(p_k^o)}{F_{k,2}(p_k^o)} = \frac{F_{k,1}(p_k^{o*})}{F_{k,2}(p_k^{o*})}, \end{aligned}$$

where p_k^{o*} denotes the global optimal solution for UE U_k. Based on nonlinear fractional programming [34], we have the following property:

Theorem 1: ψ_k^* is achieved if and only if

$$\begin{aligned} &\min_{\{p_k^o\}} \Big(F_{k,1}(p_k^o) - \psi_k^* F_{k,2}(p_k^o) \Big) \\ =&F_{k,1}(p_k^{o*}) - \psi_k^* F_{k,2}(p_k^{o*}) = 0. \quad (35) \end{aligned}$$

Proof: The detailed proof is omitted due to space limitation. A similar proof can be found in our previous work [35]. ■

Theorem 1 indicates the necessary and sufficient conditions to obtain ψ_k^*. Accordingly, p_k^{o*} can be obtained by solving the following transformed problem:

$$\begin{aligned} \mathbf{P2}: &\min_{\{p_k^o\}} \sum_{U_k \in \mathcal{U}} \Big(F_{k,1}(p_k^o) - \psi_k^* F_{k,2}(p_k^o) \Big) \\ \text{s.t.} \quad & C_1 \sim C_6. \quad (36) \end{aligned}$$

Remark 6. It can be easily proved that the objective of $\mathbf{P2}$ is convex with regards to p_k^o by calculating the corresponding second derivative.

However, the specific value of ψ_k^* required to solve $\mathbf{P2}$ is still unavailable. To obtain ψ_k^*, the iterative Dinkelbach method can be used [34]. Denote the iteration index as n and the initial value of ψ_k as a small positive number. At the n-th iteration, $p_k^o[n]$ is derived by using $\psi_k[n]$ obtained from the $(n-1)$-th iteration, which is given by

$$\begin{aligned} \mathbf{P3}: &\min_{\{p_k^o[n]\}} \sum_{U_k \in \mathcal{U}} \Big(F_{k,1}(p_k^o[n]) - \psi_k[n] F_{k,2}(p_k^o[n]) \Big) \\ \text{s.t.} \quad & C_1 \sim C_6. \quad (37) \end{aligned}$$

How to solve $\mathbf{P3}$ is provided in Subsection V-C. Then, upon obtaining $p_k^o[n]$, $\psi_k[n+1]$ is updated as

$$\psi_k[n+1] = \frac{F_{k,1}(p_k^o[n])}{F_{k,2}(p_k^o[n])}. \quad (38)$$

The iteration process will stop if

$$F_{k,1}(p_k^o[n]) - \psi_k[n] F_{k,2}(p_k^o[n]) < \varepsilon, \quad (39)$$

where ε represents the stopping criteria. The above implementation procedures are summarized as the outer loop of Algorithm 1.

2) Consensus Problem Formulation: At each iteration n, problem $\mathbf{P3}$ has to be solved with a given $\psi_k[n]$. However, the objectives in $\mathbf{P3}$ are not separable because the workload offloading variables of K UEs are coupled through the constraint term C_2. To provide a distributed solution, local copies of the global optimization variables are introduced to transform $\mathbf{P3}$ into a general consensus problem. Specifically, defining the vector of global optimization variables as $\mathbf{p}^o = \{p_1^o, \cdots, p_k^o, \cdots, p_K^o\}$, the local copy of the global vector $\mathbf{p}^o$ at UE U_k is denoted as $\tilde{\mathbf{p}}_k^o = \{\tilde{p}_1^{o,k}, \cdots, \tilde{p}_k^{o,k}, \cdots, \tilde{p}_K^{o,k}\}$. For instance, the local copy of the global variable p_{k-1}^o (i.e., the offloading strategy of UE U_{k-1}) at UE U_k is $p_{k-1}^{o,k}$.

Furthermore, we define the feasibility set of the local optimization variables for UE U_k as ω_k, which is given by

$$\omega_k = \left\{ \tilde{\mathbf{p}}_k^o | C_1 \sim C_6 \right\}. \quad (40)$$

We define the local objective function associated with the feasibility set ω_k as χ_k. If $\tilde{p}_k^{o,k} \in \omega_k$, i.e., the solution is feasible, then χ_k is equivalent to its global counterpart, i.e.,

$$\chi_k(\tilde{p}_k^{o,k}) = F_{k,1}(\tilde{p}_k^o) - \psi_k F_{k,2}(\tilde{p}_k^o). \quad (41)$$

Otherwise, if the constraints cannot be satisfied, $\chi_k(\tilde{p}_k^{o,k}) = \infty$.

Therefore, the general consensus problem corresponding to $\mathbf{P3}$ is given by

$$\begin{aligned} \mathbf{P4}: &\min_{\{\tilde{\mathbf{p}}_k^o\}} \sum_{U_k \in \mathcal{U}} \chi_k(\tilde{p}_k^{o,k}) \\ \text{s.t.} \quad & C_7: \tilde{\mathbf{p}}_k^o = \mathbf{p}^o, \forall U_k \in \mathcal{U}, \quad (42) \end{aligned}$$

where C_7 denotes the consensus constraint, i.e., the local variables duplicated at different UEs should be equal to the global variables.

Remark 7. C_7 guarantees that $\mathbf{P3}$ and $\mathbf{P4}$ are equivalent.

C. Consensus ADMM-based Distributed Solution

In this subsection, the proposed consensus ADMM-based solution is elaborated in details. Let Λ be the $K \times K$ matrix of the Lagrange multipliers corresponding to the consensus constraint C_7 in $\mathbf{P4}$. Λ is given by $\Lambda = [\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_k, \cdots, \boldsymbol{\mu}_K]$, where $\boldsymbol{\mu}_k$ is a $K \times 1$ vector.

The augmented Lagrangian for $\mathbf{P4}$ is expressed as

$$\begin{aligned} &\mathcal{L}(\{\tilde{\mathbf{p}}_k^o\}, \Lambda) \\ &= \sum_{U_k \in \mathcal{U}} \chi_k(\tilde{p}_k^{o,k}) + \sum_{U_k \in \mathcal{U}} \boldsymbol{\mu}_k^T (\tilde{\mathbf{p}}_k^o - \mathbf{p}^o) \\ &+ \frac{\rho}{2} \sum_{U_k \in \mathcal{U}} \parallel \tilde{\mathbf{p}}_k^o - \mathbf{p}^o \parallel_2^2 . \end{aligned} \tag{43}$$

The resulting iterations of updating local variables, global variables, and Lagrange multipliers are given in (44), (45), and (46), respectively.

Remark 8. From (44), it is clear that the optimization of $\tilde{\mathbf{p}}_k^o$ is carried out independently for each UE. As a result, $\mathbf{P4}$ can be decomposed into a set of subproblems, which are distributed across UEs and solved in parallel. The corresponding optimization objective for U_k is exactly χ_k.

Based on the above analysis, the energy-efficient workload offloading algorithm based on consensus ADMM is summarized as Algorithm 1. It consists of two loops. The outer loop represents the iterations to solve the nonlinear fractional programming problem, and the corresponding iteration index is n. The inner loop represents the iterations for updating the primal and dual variables, and the corresponding iteration index is defined as t. In iteration n of the outer loop, given $\psi_k[n]$, the primal and dual variables are updated sequentially to find the optimal workload offloading strategy, which are elaborated as follows:

1) $\{\tilde{\mathbf{p}}_k^o\}$ ***update*** : In iteration t of the inner loop, the optimization of $\tilde{\mathbf{p}}_k^o[t+1]$ is carried out by using (44). It is noted that (44) is actually a quadratic programming (QP) problem [16], which can be easily solved by existing QP solvers.
2) $\{\mathbf{p}^o, \boldsymbol{\mu}_k\}$ ***update*** : Compared with $\{\tilde{\mathbf{p}}_k^o\}$ update, the optimization of $\mathbf{p}^o[t+1]$ and $\boldsymbol{\mu}_k[t+1]$ can be carried out more easily due to the nature of un-constrained quadratic optimization. The specific updating processes of $p_k^o[t+1]$ and $\boldsymbol{\mu}_k[t+1]$ are shown in (45) and (46), respectively.
3) ***Termination criteria of the inner iteration***: The inner iteration stops

$$\parallel \mathbf{r}_k[t+1] \parallel_2^2 = \parallel \tilde{\mathbf{p}}_k^o[t+1] - \mathbf{P}^o[t+1] \parallel_2^2 \leq \epsilon^{pri}, \quad \forall U_k \in \mathcal{U}, \tag{47}$$

$$\parallel \mathbf{s}[t+1] \parallel_2^2 = \rho \parallel \mathbf{P}^o[t+1] - \mathbf{P}^o[t] \parallel_2^2 \leq \epsilon^{dual}. \tag{48}$$

where $\mathbf{r}_k$ and $\mathbf{s}$ denote the primal residual and the dual residual, respectively. ϵ^{pri} and ϵ^{dual} denote the thresholds for $\mathbf{r}_k$ and $\mathbf{s}$, respectively. Moreover, as proved in Subsection V-D, the primal and dual update iterations in consensus ADMM satisfy objective convergence, residual convergence and dual variable convergence as $t \to \infty$.

Algorithm 1 Consensus ADMM-based Workload Offloading Optimization Algorithm

for $k = 1, 2, \cdots, K$ **do**
  Initialize: n, t, p_k^o, ψ_k, ϵ^{pri}, ϵ^{dual}, and ε.
  Convergence = False;
  while Convergence = False **do**
    while $\parallel \mathbf{r}_k[t] \parallel_2^2 > \epsilon^{pri}$ and $\parallel \mathbf{s}[t] \parallel_2^2 > \epsilon^{dual}$ **do**
      Update $\tilde{\mathbf{p}}_k^o[t+1]$, $k = 1, \ldots, K$, concurrently via (44);
      Update $\mathbf{p}^o[t+1]$ via (45);
      Update $\boldsymbol{\mu}_k[t+1]$ via (46);
      Calculate
      $\parallel \mathbf{r}_k[t+1] \parallel_2^2 = \parallel \tilde{\mathbf{p}}_k^o[t+1] - \mathbf{P}^o[t+1] \parallel_2^2$;
      $\parallel \mathbf{s}[t+1] \parallel_2^2 = \rho \parallel \mathbf{P}^o[t+1] - \mathbf{P}^o[t] \parallel_2^2$;
      Update $t \to t+1$;
    end while
    Update $\tilde{p}_k^o[t] \to p_k^o[n]$;
    if $F_{k,1}\big(p_k^o[n]\big) - \psi_k[n] F_{k,2}\big(p_k^o[n]\big) > \varepsilon$ **then**
      $\psi_k[n+1] = F_{k,1}\big(p_k^o[n]\big) / F_{k,2}\big(p_k^o[n]\big)$
      Convergence = False
    else
      Convergence = True
    end if
    Update $n \to n+1$;
  end while
  Set $\{p_k^{o*}\} = \{p_k^o[n]\}$;
  Calculate ψ_k^* by (34);
  output: p_k^{o*}, and ψ_k^*.
end for

4) ***Termination criteria of the outer iteration***: When iteration n terminates, $p_k^o[n]$ is used to update $\psi_k[n+1]$ for the $[n+1]$-th iteration as (38). The stopping criteria of the outer loop is given in (39). In the final iteration of outer loop, the obtained workload offloading strategy converges to the optimal strategies, i.e., p_k^{o*}. ψ_k^* is calculated by using p_k^{o*} as (34).

D. Property Analysis

In this subsection, we analyze the convergence and complexity of the proposed algorithm.

1) Convergence of the Inner Iteration: The objective function of $\mathbf{P4}$ is closed, proper, and convex, and the corresponding epigraph is a closed nonempty convex set. Furthermore, the Lagrangian $\mathcal{L}(\{\tilde{\mathbf{p}}_k^o\}, \Lambda)$ has a saddle point. Thus, based on [16], the inner iteration satisfies residual convergence, objective convergence and dual variable convergence, which is shown as below.

- *Residual convergence:*

$$\sum_{U_k \in \mathcal{U}} (\tilde{\mathbf{p}}_k^o[t] - \mathbf{p}^o[t]) \to 0, t \to \infty, \tag{49}$$

  which indicates that the iterations approach feasibility.
- *Objective convergence:*

$$\sum_{U_k \in \mathcal{U}} \chi_k(\tilde{p}_k^{o,k}[t,n]) \to \sum_{U_k \in \mathcal{U}} \psi_k^*[n], t \to \infty, \tag{50}$$

$$\{\tilde{\mathbf{p}}_k^o[t+1]\} = \arg\min_{\tilde{\mathbf{p}}_k^o} \left\{ \chi_k(\tilde{p}_k^{o,k}) + \boldsymbol{\mu}_k^T(\tilde{\mathbf{p}}_k^o - \mathbf{p}^o[t]) + \frac{\rho}{2} \parallel \tilde{\mathbf{p}}_k^o - \mathbf{p}^o[t] \parallel^2 \right\}, \tag{44}$$

$$\{\mathbf{p}^o[t+1]\} = \arg\min_{\mathbf{p}^o} \left\{ \sum_{U_k \in \mathcal{U}} \boldsymbol{\mu}_k^T(-\mathbf{p}^o) + \frac{\rho}{2} \sum_{U_k \in \mathcal{U}} \parallel \tilde{\mathbf{p}}_k^o[t+1] - \mathbf{p}^o \parallel^2 \right\}, \tag{45}$$

$$\{\boldsymbol{\mu}_k[t+1]\} = \{\boldsymbol{\mu}_k\}[t] + \rho(\tilde{\mathbf{p}}_k^o[t+1] - \mathbf{p}^o[t+1]). \tag{46}$$

TABLE II
PARAMETERS.

Parameter	Value
Number of UEs K	$10 \sim 20$
Number of RSUs M	4
Number of servers in the RSU c	4
Diameter of RSU coverage d_m	400 m $\sim$ 650 m
Workload data size θ_k	40 $\sim$ 150 Mb
Average vehicle velocity $\bar{v}_k$	40 $\sim$ 80 km/h
Delay constraint τ_k	2 $\sim$ 50 s
Local computing power β_k	0.5 W
Transmission power of vehicle P_k^V	23 dBm
Bandwidth B_k	2 MHz
Average workload arrival rate λ_k	2 $\sim$ 5 workload/s
Local computing capability u_k^l	1.4 $\sim$ 2.2 GHz
Workload computation complexity δ	0.5 GHz/workload
Edge computing capability u_m^e	12 GHz
Noise power N_0	-97 dBm
Time required to deliver the TPBU message κ	20 ms
Time required to deliver the L2 report T_{L2}	15 ms
Time required for the sMAG (nMAG) to send the data packets to the nMAG (RSU) $T'_{k,PT}$ ($T_{k,PT}$)	10 $\sim$ 30 ms
Time required to deliver the HI message φ	10 ms
Time for confirming the received profile and creating a new cache entry ϖ_k	10 ms

which indicates that the objective function eventually converges to the optimal value.

- *Dual variable convergence:* $\boldsymbol{\mu}_k[t] \rightarrow \boldsymbol{\mu}_k^*$ as $t \rightarrow \infty$, where $\boldsymbol{\mu}_k^*$ is a dual optimal vector.

2) Convergence of the Outer Iteration: It can be proved that $p_k^o[n]$ converges to p_k^{o*} in a super-linear speed as n increases. A similar proof can be found in [35].

3) Complexity: In each iteration of the outer loop, **P4** is solved to produce a decreasing sequence of ψ_k. Here, we define n^{loop} as the required number of iterations by the outer loop to reach convergence. Similarly, in each iteration of the inner loop, (44), (45) and (46) are updated sequentially to obtain $p_k^{o*}[n]$ for a given $\psi_k[n]$. We define t^{loop} as the number of iterations required by the inner loop to reach convergence. Hence, the computation complexity for solving each decomposed subproblem is $\mathcal{O}(n^{loop}t^{loop})$.

VI. Simulation Results and Discussions

In this section, we validate the proposed algorithm based on the real-world topology of the Xidan area in Beijing, China. This area is featured with the Chang'an avenue, which is the road to several scenic spots such as Tian'an men Square and the Forbidden City as well as the headquarters of many companies and government agencies are located in this area. An aerial snapshot obtained from the Baidu map is shown in Fig. 3. First, The data of the digital map downloaded from OpenStreetMap is imported to SUMO. Then, vehicle traffics are generated based on the realistic road topologies, which are marked as small yellow triangles in Fig. 3. The critical attributes of each vehicle such as location and velocity are obtained during simulation, based on which the average velocity of each vehicle is estimated by using a simple rolling window regression approach [36]. The RSUs are also deployed

Fig. 3. Evaluation scenario based on the real-world topology of Xidan area, Beijing,China.

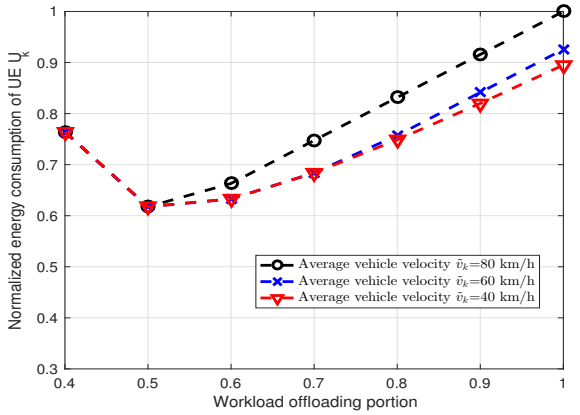


Fig. 4. The relationship between p_k^o and the energy consumption of UE U_k. ($K = 1, d_m = 400\text{m}, \theta_k = 120\text{Mb}, \lambda_k = 4$)

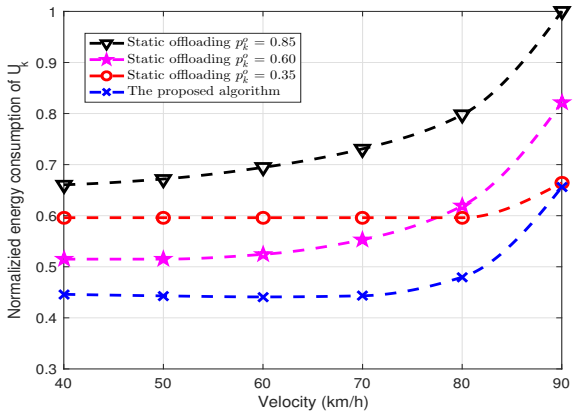


Fig. 5. The energy consumption of UE U_k versus average vehicle velocity $\bar{v}_k$. ($K = 1, d_m = 400\text{m}, \theta_\text{k} = 120\text{Mb}, \lambda_\text{k} = 4$)

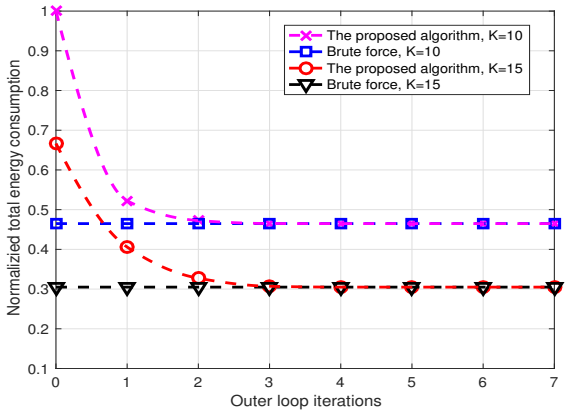


Fig. 7. The convergence performance of the proposed algorithm. ($K = 10, 15$, and $d_m = 400\text{m}$)

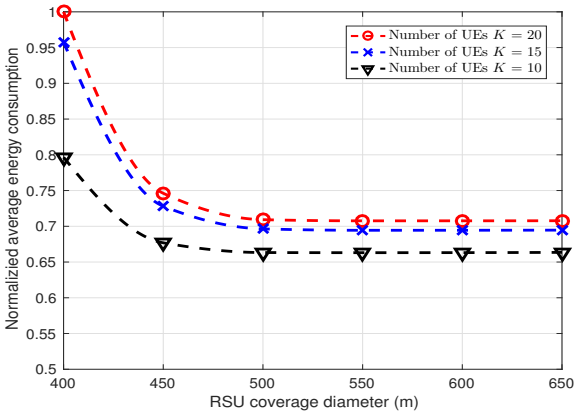


Fig. 6. Average energy consumption per UE versus RSU coverage diameter with different numbers of UEs. ($\bar{v}_k = 80\text{km/h}$)

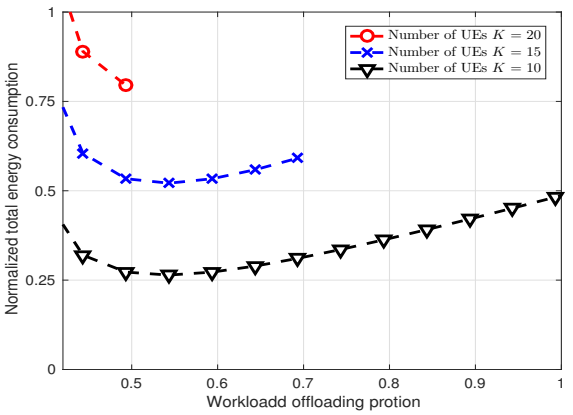


Fig. 8. The total energy consumption versus workload offloading portion with different numbers of UEs. ($d_m = 400\text{m}, \bar{\text{v}}_\text{k} = 40\text{km/h}$)

along the Chang'an avenue.

The simulation parameters are summarized in Table II [37]–[39]. The proposed algorithm is compared with two heuristic algorithms including the brute-force searching algorithm, and the static offloading algorithm algorithm [11]. In the static offloading algorithm, the portion of offloaded workload is fixed and the same for any UE.

Fig. 4 shows the relationship between p_k^o and the energy consumption of UE U_k under different average vehicle velocities. It is clear that the energy consumption decreases firstly and then increases with p_k^o. When p_k^o is small, the energy consumption of transmission is less than that of the local computing. Hence, more energy can be saved by increasing p_k^o. However, when $p_k^o > 0.5$, the energy consumed for data transmission starts to dominate the total energy consumption. In other words, the energy saving brought by workload offloading cannot compensate the energy consumed for data transmission. As a result, the energy consumption increases monotonically with p_k^o. Furthermore, we found that the energy consumption also increases with the average vehicle velocity when $p_k^o > 0.5$. The reason is that higher velocity will cause more offloading failures when p_k^o is large. This not only increases transmission energy consumption, but also results in higher energy consumption of local computing because more workloads have to be processed locally.

Fig. 5 shows the energy consumption versus vehicle velocity. The proposed algorithm is compared with the static offloading algorithm under different workload offloading portions. When p_k^o is large, i.e., $p_k^o = 0.85$ and $p_k^o = 0.6$, the energy consumption of the static offloading algorithm increases dramatically with the vehicle velocity. The reason behind is that higher velocity leads to frequent offloading failures. In comparison, the energy consumption of the proposed algorithm remains constant when the vehicle velocity is increased from 40 to 70 km/h. Simulation results demonstrate that the proposed algorithm is more robust to the negative impact caused by high vehicle mobility. Even when the velocity exceeds 70 km/h, the proposed algorithm still outperforms the static offloading algorithm. The reason is that the proposed algorithm is able to reduce the energy consumption by dynamically adjusting the offloading portion. For example, the optimal offloading portions for the velocities

of 70, 80, and 90km/h are 0.4576, 0.3918, and 0.3407, respectively. That is, as velocity increases, the portion of workload to be offloaded is also reduced accordingly to avoid offloading failure.

Fig. 6 shows the average energy consumption per UE versus the RSU coverage diameter with different numbers of UEs. Enhancing the RSU coverage has positive impacts on the energy consumption. It not only reduces the handover latency but also relaxes the latency requirement of data transmission. Hence, the average energy consumption per UE decreases monotonically with the RSU coverage diameter. However, when the coverage diameter reaches a certain value, the performance improvement becomes saturated because the optimal energy consumption have already been achieved. Furthermore, when the number of UEs is doubled, the average energy consumption per UE only increases slightly. The reason is that the offloading portion is dynamically adjusted in accordance with the number of UEs.

The convergence performance of the proposed algorithm is shown in Fig. 7. The brute-force searching algorithm which examines all possible of combinations to find the optimal solution is utilized as a performance benchmark. It is observed that the proposed algorithm can converge rapidly to the optimal result only within $2 \sim 3$ iterations.

Fig. 8 shows the total energy consumption versus different workload offloading portions. The offloading portion of any UE is kept as the same, i.e., $p_k^o = p_{k'}^o, \forall k' \neq k$. The numerical results are consistent with Fig. 4, i.e., the energy consumption decreases firstly and then increases with p_k^o. Moreover, it is observed that the maximally allowed offloading portion decreases monotonically as the number of UEs increases. This is due to the constraint C_2 of problem $\mathbf{P1}$ that the sum arrival rate of all UEs' workload cannot exceed the processing node of the VEC node.

VII. Conclusion

In this paper, we have investigated the energy-efficient workload offloading for in-vehicle UEs with limited battery capacity, and proposed the consensus ADMM-based energy-efficient resource allocation algorithm. First, by taking the high mobility of vehicles into account, we have proposed a queuing model to derive the closed-form expressions of the computation latency and the waiting latency. Then, we have formulated a workload offloading optimization problem with the explicit considerations of the overall energy consumption and latency. Next, we have proposed a consensus ADMM-based distributed solution. The formulated joint problem was decomposed into a set of subproblems and solved in parallel. Finally, a real-world topology based simulation has been conducted. For the future work, we will investigate the delay minimization problem in VEC by employing machine learning based workload prediction and computation resource prediction.

References

[1] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, "Energy-efficient workload offloading and power control in vehicular edge computing," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops*, Barcelona, Spain, Apr. 2018, pp. 191–196.

[2] Z. Zhou, H. Yu, C. Xu, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Dependable content distribution in D2D-based cooperative vehicular networks: A big data-integrated coalition game approach," *IEEE Trans. Intell. Transport. Syst.*, vol. 19, no. 3, pp. 953–964, Mar. 2018.

[3] C. Huang, R. Lu, X. Lin, and X. Shen, "Secure automated valet parking: A privacy-preserving reservation scheme for autonomous vehicles," *IEEE Trans. Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11 169–11 180, Nov. 2018.

[4] H. Peng, D. Li, Q. Ye, K. Abboud, H. Zhao, W. Zhuang, and X. Shen, "Resource allocation for cellular-based inter-vehicle communications in autonomous multiplatoons," *IEEE Trans. Trans. Veh. Technol.*, vol. 66, no. 12, pp. 11 249–11 263, Dec. 2017.

[5] F. Tang, B. Mao, Z. M. Fadlullah, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 154–160, Feb. 2018.

[6] H. Wu, N. Zhang, X. Tao, Z. Wei, and X. Shen, "Capacity- and trust-aware BS cooperation in non-uniform HetNets: Spectral efficiency and optimal BS density," *IEEE Trans. Trans. Veh. Technol.*, vol. 66, no. 12, pp. 11 317–11 329, Dec. 2017.

[7] J. Feng, Z. Liu, C. Wu, and Y. Ji, "AVE: Autonomous vehicular edge computing framework with ACO-based scheduling," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 10 660–10 675, Dec. 2017.

[8] Z. Zhou, J. Feng, C. Zhang, Z. Chang, Y. Zhang, and K. M. S. Huq, "SAGECELL: Software-defined space-air-ground integrated moving cells," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 92–99, Aug. 2018.

[9] Z. Zhou, J. Feng, B. Gu, B. Ai, S. Mumtaz, J. Rodriguez, and M. Guizani, "When mobile crowd sensing meets UAV: Energy-efficient task assignment and route planning," *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5526–5538, Nov. 2018.

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

[11] Z. Chang, Z. Zhou, T. Ristaniemi, and Z. Niu, "Energy efficient optimization for computation offloading in fog computing system," in *Proc. IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–6.

[12] R. Deng, R. Lu, C. Lai, T. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things J.*, vol. 3, no. 6, pp. 1171–1181, May 2016.

[13] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Select. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Sept. 2016.

[14] 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, Dec. 2017.

[15] M. Li, F. R. Yu, P. Si, H. Yao, E. Sun, and Y. Zhang, "Energy-efficient M2M communications with mobile edge computing in virtualized cellular networks," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, July 2017, pp. 1–6.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[17] Z. Zhou, H. Liao, B. Gu, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Robust mobile crowd sensing: When deep learning meets edge computing," *IEEE Network*, vol. 32, no. 4, pp. 54–60, July 2018.

[18] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Commun. Mag*, vol. 55, no. 3, pp. 38–43, Mar. 2017.

[19] B. Yin, W. Shen, Y. Cheng, L. X. Cai, and Q. Li, "Distributed resource sharing in fog-assisted big data streaming," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, July 2017, pp. 1–6.

[20] Q. Vu, L. Tran, R. Farrell, and E. Hong, "Energy-efficient zero-forcing precoding design for small-cell networks," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 790–804, Nov. 2016.

[21] W. Yu, G. Chen, and M. Cao, "Consensus in directed networks of agents with nonlinear dynamics," *IEEE Trans. Automat. Contr.*, vol. 56, no. 6, pp. 38–43, June 2011.

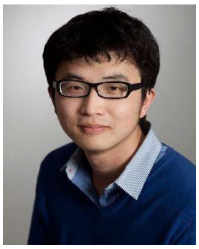
[22] C. Liang, F. R. Yu, H. Yao, and Z. Han, "Virtual resource allocation in information-centric wireless networks with virtualization," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9902–9914, Feb. 2016.

[23] Y. Wang, L. Wu, and S. Wang, "A fully-decentralized consensus-based ADMM approach for DC-OPF with demand response," *IEEE Trans. Smart Grid*, vol. 8, no. 6, pp. 2637–2647, Nov. 2017.

[24] N. Zhang, S. Zhang, P. Yang, O. Alhussein, W. Zhuang, and X. Shen, "Software defined space-air-ground integrated vehicular networks: Challenges and solutions," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 101–109, July 2017.
[25] M. Kim and S. Lee, "Enhanced network mobility management for vehicular networks," *IEEE Trans. Intell. Transport. Syst.*, vol. 17, no. 5, pp. 1329–1340, May 2016.
[26] W. Na, N. Dao, and S. Cho, "Mitigating WiFi interference to improve throughput for in-vehicle infotainment networks," *IEEE Wireless Commun.*, vol. 23, no. 1, pp. 22–28, Mar. 2016.
[27] Y. Zhang, D. Niyato, and P. Wang, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Trans. Mobile Comput.*, vol. 14, no. 12, pp. 2516–2529, Dec. 2015.
[28] M. Patra, R. Thakur, and C. Murthy, "Improving delay and energy efficiency of vehicular networks using mobile femto access points," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1496–1505, Feb. 2017.
[29] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Trans. Cloud Comput.*, vol. 5, no. 4, pp. 725–737, Oct./Dec. 2017.
[30] T. Kim and M. Dong, "An iterative hungarian method to joint relay selection and resource allocation for D2D communications," *IEEE Trans. Wireless Commun. Lett.*, vol. 3, no. 6, pp. 625–628, Dec. 2014.
[31] S. Zhao, Y. Chen, and J. Farrell, "High-precision vehicle navigation in urban environments using an MEM's IMU and single-frequency GPS receiver," *IEEE Wireless Commun.*, vol. 17, no. 10, pp. 2854–2867, Apr. 2016.
[32] J. Kingman, *Poisson Processes*. Oxford University Press, New York, 2005.
[33] L. Kleinrock, *Queueing Systems.Volume I:Theory*. John Wiley & Sons, New York, 1972.
[34] W. Dinkelbach, "On nonlinear fractional programming," *Manag. Sci.*, vol. 13, no. 7, pp. 492–498, Mar. 1967.
[35] Z. Zhou, K. Ota, M. Dong, and C. Xu, "Energy-efficient matching for resource allocation in D2D enabled cellular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5256–5268, June 2017.
[36] S. Hadjiantoni and E. J. Kontoghiorghes, "A numerical method for the estimation of time-varying parameter models in large dimensions," *stat.ME*, pp. 1–20, Mar. 2017.
[37] Y. Wang, X. Lin, and M. Pedram, "A nested two stage game-based optimization framework in mobile cloud computing system," in *Proc. IEEE Int. Symp. Service-Oriented Syst. Eng.*, Redwood City, USA, Mar. 2013, pp. 494–502.
[38] Y. Sun, S. Zhou, and J. Xu, "EMM energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Select. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Oct. 2017.
[39] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Networking*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

Zhenyu Zhou (M'11-SM'17) received his M.E. and Ph.D degree from Waseda University, Tokyo, Japan in 2008 and 2011 respectively. From April 2012 to March 2013, he was the chief researcher at Department of Technology, KDDI, Tokyo, Japan. From March 2013 to now, he is an Associate Professor at School of Electrical and Electronic Engineering, North China Electric Power University, China. He served as an Associate Editor for IEEE Access, EURASIP Journal on Wireless Communications and Networking and a Guest Editor for IEEE Communications Magazine and Transactions on Emerging Telecommunications Technologies. He also served as workshop co-chair for IEEE Globecom 2018, IEEE ISADS 2015, and TPC member for IEEE Globecom, IEEE CCNC, IEEE ICC, IEEE APCC, IEEE VTC, IEEE Africon, etc. He is a voting member of IEEE Standard Association P1932.1 Working Group. He was the recipient of the IEEE Vehicular Technology Society "Young Researcher Encouragement Award" in 2009, the Beijing Outstanding Young Talent Award in 2016, the IET Premium Award in 2017, the IEEE ComSoc Green Communications and Computing Technical Committee 2017 Best Paper Award, the IEEE Globecom 2018 Best Paper Award, and the IEEE ComSoc Green Communications and Computing Technical Committee 2018 Best Paper Award. His research interests include green communications, vehicular communications, and smart grid communications. He is a senior member of IEEE.

Junhao Feng is currently working toward the M.S. degree with North China Electric Power University, Beijing, China. His research interests include resource allocation, interference management, and energy management in D2D communications. He was the recipient of the IEEE Globecom 2018 Best Paper Award and the IEEE ComSoc Green Communications and Computing Technical Committee 2018 Best Paper Award.



Zheng Chang received the B.Eng. degree from Jilin University, Changchun, China, in 2007, the M.Sc. (Tech.) degree from the Helsinki University of Technology, Espoo, Finland, in 2009, and the Ph.D. degree from the University of Jyväskylä, Jyväskylä, Finland, in 2013. Since 2008, he has held various research positions at the Helsinki University of Technology, the University of Jyväskylä, and Magister Solutions Ltd., Finland. He was also a Visiting Researcher with Tsinghua University, China, in 2013, and the University of Houston, TX, USA, in 2015. He has been awarded by the Ulla Tuominen Foundation, the Nokia Foundation, and the Riitta and Jorma J. Takanen Foundation for his research work. He is currently with the University of Jyväskylä. His research interests include cloud computing, radio resource allocation, IoT, vehicular networks, security and privacy, and green communications.

Xuemin (Sherman) Shen (M'97-SM'02-F'09) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, social networks, smart grid, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and the Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award 5 times from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, the Tutorial Chair for the IEEE VTC'11 Spring, the Chair for the IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking. He is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL and the Vice President on Publications of the IEEE Communications Society.