Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network

Min Chen, *Senior Member, IEEE*, and Yixue Hao

***Abstract*— With the development of recent innovative applications (e.g., augment reality, self-driving, and various cognitive applications), more and more computation-intensive and data-intensive tasks are delay-sensitive. Mobile edge computing in ultra-dense network is expected as an effective solution for meeting the low latency demand. However, the distributed computing resource in edge cloud and energy dynamics in the battery of mobile device makes it challenging to offload tasks for users. In this paper, leveraging the idea of software defined network, we investigate the task offloading problem in ultra-dense network aiming to minimize the delay while saving the battery life of user's equipment. Specifically, we formulate the task offloading problem as a mixed integer non-linear program which is NP-hard. In order to solve it, we transform this optimization problem into two sub-problems, i.e., task placement sub-problem and resource allocation sub-problem. Based on the solution of the two sub-problems, we propose an efficient offloading scheme. Simulation results prove that the proposed scheme can reduce 20% of the task duration with 30% energy saving, compared with random and uniform task offloading schemes.**

***Index Terms*— Software defined networking, mobile edge computing, task offloading, resource allocation.**

I. Introduction

With the development in cloud computing and wireless communication technology, there is an explosive growth in the number of mobile devices accessing wireless networks. It is predicted that by 2020 the total quantity of world-wide devices would be 75 billion, while the volume of mobile traffic would exceed 24.3 exabytes/month [1]. Furthermore, user equipment (UE) will be more and more intelligent while the applications in UEs will require extensive computation power and persistent data processing. These applications include wearable virtual reality (VR) streaming [2], mobile social media [3], vehicular system [4], [5] and self-driving, etc. However, the development of these emerging applications and services is restricted by the computational capacity and battery life of UEs. These emerging applications with intense requirements for computational capacity could only rely on advanced computational offloading and the improved communication infrastructure. Therefore, future communication network not only needs to support wireless content access, but also offers the provisioning of computational offloading for UEs [6], [7], especially for mobile healthcare [8], smart grid [9], Internet of vehicles [10], intelligent services [11], etc.

Ultra-dense network is proposed as a key technology for 5G to address the above challenging demand on wireless access [12], which includes small cell base stations (BSs) and macro cell BSs. Due to the densified deployment of small cell BSs, a huge access capacities can be provided by 5G network to users [13], [14]. To address the challenge of timely offloading computation-intensive task, the mobile edge computing was proposed [15], [16]. It has provided computing services with short delay and high performance to users through edge clouds or fog nodes deployed on the network edge in order to meet the computing requirements of delay-sensitive tasks. There are two major advantages of using the edge cloud [17], [18]: (i) In contrast to the local computing [19], the edge cloud computing can overcome the restrictions of limited computation capacity on mobile terminals; (ii) Compared with the computation offloading towards remote cloud [20], the edge cloud computing can avoid a large latency caused by offloading the task contents on the remote cloud. Thus, mobile edge computing exhibits a potential to achieve a better tradeoff for delay-sensitive and computation-intensive tasks [21].

In the future communication network, the edge cloud will be deployed in the ultra-dense network and is able to provide computation offloading services to users [22]. The users can offload computation-intensive task to the edge cloud. However, at present, most of the existing work on ultra dense network and mobile edge computing are separate. The effective use of mobile edge computing in ultra-dense network faces the following challenges: (i) In ultra-dense network, the intensive deployment of small cell BSs increases the complexity of network environments. In addition, the computational resources of edge cloud are limited. Thus, how to control these distributed computing resources is a challenging problem; (ii) When users offload computing tasks, they have little information about the wireless networks to be accessed, including traffic load of accessed network and computation load of edge clouds. So, how to conduct a task offloading according to the residual battery capacity of mobile device and network status is challenging. Thus, to ensure the reasonable utilization

Manuscript received September 30, 2017; revised February 5, 2018; accepted February 27, 2018. Date of publication March 12, 2018; date of current version May 21, 2018. This work was supported by the National Natural Science Foundation of China (under Grant No. U1705261), and Director Fund of WNLO. *(Corresponding author: Yixue Hao.)*

M. Chen is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the Wuhan National Laboratory for Optoelectronics, Wuhan 430074, China (e-mail: minchen@ieee.org).

Y. Hao is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: yixuehao@hust.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2018.2815360

of computing resources and make task placement efficiently, a reasonable controller is needed.

As a new networking paradigm, software defined networking (SDN) [23] nowadays can achieve logically centralized control on the distributed network nodes and mobile devices. The SDN has been widely studied from the placement of SDN controller [24], [25], the network function placement [26] and the application of SDN (e.g., live stream transcoding [27], [28] and multi-paths routing [28]). However, to the best of our knowledge, there is no exiting work on the task offloading for mobile edge computing in ultra-dense network. By introducing SDN technology, this paper proposes a task offloading scheme in a software defined ultra dense network (SD-UDN), as shown in Fig. 1. Based on the separation of control plane and data plane, the major computational and control functionalities are decoupled from the distributed small cell BSs and consolidated in the centralized SD-UDN controller, which resides at macro cell BS. The SD-UDN controller can collect information of mobile device and edge cloud, sense the network status from a global perspective. Based on the decision of SD-UDN controller, the mobile device is advised to perform tasks locally or offload part of them to edge cloud for processing.

Furthermore, we formulate the task offloading scheme into a mixed integer non-linear program problem, and prove its NP-hardness. The proposed strategy of task offloading aims to minimize task duration under the battery energy consumption constraint of UE. To be specific, our task offloading scheme includes two parts: (i) Where should the task be processed? That is, decide to process a task locally or in the edge cloud according to the residual battery capacity of mobile device. Furthermore, decide which edge cloud for the task to be placed on according to the availability and load of various edge clouds (i.e., task placement problem); (ii) Decide how much computing resource of edge cloud should be allocated to each task (i.e., resource allocation problem). We then transform the problem into two sub-problems: a convex sub-problem (i.e., computing resource allocation problem) and 0-1 programming sub-problem (i.e., task placement problem). Using alternative optimization techniques, we obtain an efficient solution for computing resource allocation and task placement. The performance of the proposed task offloading scheme in SD-UDN is evaluated and compared with random and even offloading schemes. Experimental results demonstrate that our scheme can reduce task duration by 20% and energy cost by 30%. In summary, the main contributions of this paper include:

- We propose an innovative framework of task offloading for mobile edge computing in SD-UDN. By deploying controller at macro cell BS, the global information about mobile devices, base stations, edge cloud and tasks can be obtained, and thus enabling the optimal task offloading of mobile devices.
- We present an efficient software defined task offloading (SDTO) scheme, which not only reduces the task duration but also considers the battery capacity of UE. Specifically, when fixing offloading decision, we can demonstrate that computing resource allocation to each task is convex sub-problem. Based on this, we adopt a task placement algorithm to give the effective task offloading scheme.
- We conduct extensive experiments to evaluate the performance of SDTO scheme. The experimental results validate that our proposed algorithm can reduce 20% of the task duration and save 30% of the energy cost as compared to random and uniform task computation offloading.

The remainder of this paper is organized as follows. Section II reviews related works. The system model and problem are described in Section III. We propose the efficient task offloading scheme in Section IV. Our simulation results and discussions are given in Section V. Finally, Section VI concludes this paper.

II. Related Works

Nowadays, with substantial increase in both quantity and intelligence of mobile devices, more and more mobile applications require a good deal of computational task treatment. However, due to the limited computation and battery capacity of UE, it is quite difficult to handle computing intensive tasks locally. To this end, with the development of mobile cloud computation(e.g., Clonecloud [30], Follow me cloud [31], [32]), mobile devices can offload computation-intensive tasks to the cloud for processing.

Mobile cloud computing mainly exhibits two advantages [19], [33]: (i) Reduction of the task duration which includes transmission time for mobile device to offload task to the cloud, execution time in cloud and transmission time for the cloud to send the task result to mobile device; (ii) Decrease in energy consumption at mobile devices. For the task duration, Barbera *et al.* [20] proposed that delay-sensitive tasks should be executed on the cloudlet, and tasks with loose delay requirements could be executed on the cloud. For the energy consumption problem at UEs, Liu *et al.* [34] showed that tasks needing transmission of large volumes of data with less requirement on computing resources are suitable for the execution at UE, and computation-intensive tasks are more suitable to be executed in cloud. For system energy consumption, Chen *et al.* [19] proposed a scheme where task is offloaded in cloud and mobile cloudlet based on user mobility, thus to reduce system energy consumption. However, task offloading to the cloud via cellular network would produce a large delay.

In order to overcome this challenge, mobile edge computing is proposed [35]. In mobile edge computing, the task offloading strategy and resource allocation are the main research points. For the task offloading, Tong *et al.* [17] designed a hierarchical edge computing architecture according to the distance between the edge server and users, and proposed an optimal offloading scheme for minimizing the task duration by using heuristic algorithm. Sun *et al.* [36] developed a novel user-centric mobility management scheme for mobile edge computing in ultra dense network. Chen *et al.* [18] designed multi-user task offloading in mobile edge computing. As for the resource allocation of edge computing, Xiao and Krunz [37] proposed

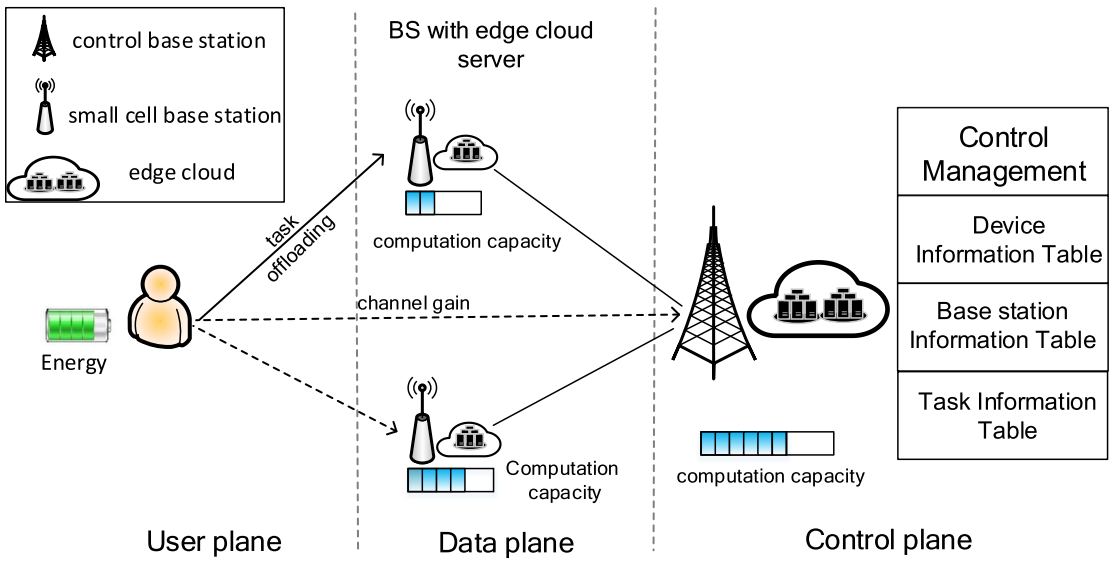


Fig. 1. Illustration of task offloading for mobile edge computing in software defined ultra-dense network.

that the cooperation of edge clouds can not only reduce the processing delay of user tasks, but also reduce the energy consumption. In a word, mobile edge computing can improve the quality of service and energy efficiency by optimizing task offloading and resource allocation policies. Furthermore, some existing works considered both task offloading and resource allocation in mobile edge computing, Chen *et al.* [12] considered the task offloading and resource allocation in mobile edge computing. However, most of the existing works consider a single mobile edge computing server. Multi-server mobile edge computing is only considered in Tran and Pompili [38], but this work does not consider the ultra dense network, it is hard to obtain insights for the design of key parameters.

Taking into account the limitations of existing work, in this paper, we design software defined task offloading for mobile edge computing in ultra-dense network and propose a efficient task offloading scheme. Furthermore, in Table. I, a taxonomy of various schemes is provided for a straightforward comparison in terms of computing resource pool, applicability of 5G network, computation capacity, etc.

III. System Model and Problem Formulation

In this section, we first introduce the framework of SD-UDN. Then, we give the task offloading problem and prove that it is a NP-hard problem.

A. The Architecture of SD-UDN

Fig. 1 shows the framework of task offloading for mobile edge computing in SD-UDN, which includes three planes, i.e., user plane, data plane and control plane. User plane consists of the users who need offloading computing tasks. Data plane mainly corresponds to small cell BSs and edge clouds deployed near small cell BSs. Control plane is realized by the SD-UDN controller deployed at macro cell BS. The users are connected to the small cell BSs or macro cell BS through wireless link while the distributed small cell BSs are connected to the central macro cell BS via high speed fronthaul network.

Although the concept of SDN evolves, its main idea is to decouple the control plane from the data plane by virtualization [42]. Then with air interface separation [43], the control coverage provided by macro cell BS can be further decoupled from the data coverage provided by small cell BSs in SD-UDN. To be specific, the macro cell BS supports control coverage to the entire macro cell. The major control functionalities like resource allocation and scheduling are centralized at the SD-UDN controller. From a global view of the network states, the SD-UDN controller can collect information of mobile users and the small cell BSs within the macro cell BS. Then, it optimizes the network configurations on demand.

The SD-UDN controller maintains mobile device information table, BS information table and task information table. The mobile device information table includes data such as the remaining battery capacity and CPU cycle of mobile device. BS information table includes radio access load, computation load of edge cloud, etc. Task information table includes the task type, task data amount and task computation amount. Typically, a user periodically sends the measurement information to the nearest serving BS. Then the BS integrates multiple users' information and edge cloud information together and periodically transmits to the SD-UDN controller.

Under the framework of SD-UDN, the process of task handing is illustrated as follows. The mobile device selects the nearest BS for task requesting, then the BS transmits the corresponding task request to the SD-UDN controller. When the SD-UDN controller receives the task request information, the SD-UDN controller updates all the information tables, gives the task offloading policy of mobile device and the resource allocation strategy of edge cloud according to the delay and energy consumption of the task. Since the requesting

TABLE I
COMPARISON OF SEVERAL COMPUTATION OFFLOADING SCHEME

Scheme in References	Computing Resource Pool	Oriented 5G network	Computation Capacity	Task Duration	Energy Cost
[30] [20]	Remote Cloud	No	High	High	High
[39] [40]	Ad-hoc	Yes (D2D)	Low	High	Low
[19] [41]	Cloudlets	Yes (D2D)	Low	High	Low
[17]	Edge Cloud	No	Medium	Low	Medium
[18]	Edge Cloud	Yes	Medium	Low	Medium
SDTO	Edge Cloud	Yes	Medium	Low	Low

data packet is very small, this paper ignores the delay and energy consumption of sending request from mobile device to the SD-UDN controller.

B. Network Model

We assume that SD-UDN includes n densely deployed BSs, and edge clouds are equipped at each BS. The mobile edge clouds are indexed as $\mathcal{B} = \{b_1, b_2, \cdots, b_n\}$. We denote the set of users in SD-UDN as $\mathcal{U} = \{u_1, u_2, \cdots, u_m\}$ and consider each mobile device user u_i, has a computation task to be executed locally or offload to the edge cloud. We adopt a widely used task model (see [16], [18], [44]) to describe task Q_i, i.e., $Q_i = (\omega_i, s_i)$, where ω_i stands for computation amount of the task Q_i, i.e., the CPU cycles needed in total to complete the task, and s_i stands for the size of computation task Q_i, i.e., the amount of data contents (e.g., the data input and associated processing code) to be delivered toward the edge cloud. Due to the densification deployment of the BS, each user can be served by multiple BS. Denote $\mathcal{A}(u_i)$ as the set of BS that provide services to the user u_i. In this paper, we focus on local computing and offloading the computation task to the edge cloud without further offloading to the remote cloud.

C. Communication Model

We first introduce the communication model. We give the uplink data rate when user mobile device offloads task onto edge cloud. Let $h_{i,j}$ denote the channel gain between the user u_i and BS b_j, where $b_j \in \mathcal{A}(u_i)$. In this paper, we assume that the user mobility is not high during the task offloading, so $h_{i,j}$ is a constant. Denote p_i as the transmission power of users, thus user uplink data rate of $r_{i,j}$ can be obtained as follows:

$$r_{i,j} = B \log_2 \left(1 + \frac{p_i h_{i,j}}{\sigma^2 + I_{i,j}}\right) \tag{1}$$

where σ^2 is noise power of mobile device, B is channel bandwidth and $I_{i,j}$ is the inter-cell interference power. Then we can obtain that transmission delay for user offloading the task Q_i to edge cloud is as follows:

$$t_{i,j}^T = \frac{s_i}{r_{i,j}} \tag{2}$$

Further we can obtain that the transmission energy consumption for user offloading the task Q_i to edge cloud is as follows:

$$\varepsilon_{i,j}^E = \frac{p_i s_i}{r_{i,j}} \tag{3}$$

D. Task Offloading Model

A computation task can be handled locally or be offloaded to edge cloud for processing. Next we will discuss the local computing and mobile edge computing.

1) Local Computing: For local task computing, we define f_i^l as a CPU computing capability of mobile device user u_i. Thus, the local execution time of task Q_i can be expressed as follows:

$$t_i^L = \frac{\omega_i}{f_i^l} \tag{4}$$

Also, we can obtain the energy consumption for local task processing as:

$$\varepsilon_i^L = \rho_i \omega_i \tag{5}$$

where ρ_i is the power coefficient of energy consumed per CPU cycle.

2) Mobile Edge Computing: Considering the difference of computing resource of edge clouds, we denote the computing resource (CPU cycles per second) of edge cloud as $f^c = \{f_1^c, f_2^c, \cdots, f_n^c\}$. According to the communication model, the total task duration in edge cloud consists of time consumed by two procedures, i.e., (i) time consumed when the user offloads the task, (ii) time consumed when computation tasks are processed on the edge cloud. Therefore, the task duration of task Q_i can be obtained as follows:

$$t_i^E = \frac{\omega_i}{\kappa_i^{\gamma_i} f_{\gamma_i}^c} + \frac{s_i}{r_{i,\gamma_i}} \tag{6}$$

where γ_i stands for the edge cloud where task Q_i is placed on, i.e., $\gamma_i \in \mathcal{A}(u_i)$, and $\kappa_i^{\gamma_i}$ stands for the proportion of computing resource allocated to task Q_i by edge cloud.

Similar to many studies such as [18] and [36], we ignore the transmission delay for edge clouds to sends the task results back to the user. This is because the data size after task processing is generally smaller than it before processing, and downlink rate from BS to mobile device is higher than uplink rate from mobile device to BS.

E. Problem Formulation

In this paper, in term of the limited computation power of edge cloud and the restricted battery capacity of mobile devices, we consider the following two problems:

- Task placement problem: decide mobile user processing locally or in the edge cloud, and which edge cloud task should be place on.

TABLE II
NOTATION TABLE

Q_i	Computation task Q_i of user
ω_i	CPU cycle needed by computation task Q_i
s_i	Computation amount required by computation task Q_i
$E^i_{\max}$	Total battery energy of user u_i.
p_i	The transmitting power of user u_i
$h_{i,j}$	The channel gain between the user u_i and BS b_j
σ^2	The noise power of user mobile device.
$I_{i,j}$	The inter-cell interference power.
f^l_i	The CPU frequency of user mobile device u_i.
f^c_i	The CPU frequency of edge cloud b_i
ρ_i	The power coefficient of energy consumed per CPU cycle of user u_i
t^L_i	The total task duration of Q_i processing locally
t^E_i	The total task duration of Q_i in edge cloud processing
ε^L_i	The total energy consumption of u_i when processing locally
ε^E_i	The total energy consumption of u_i when offloading to edge cloud

- Resource allocation problem: how much computing resource of edge cloud allocate to each task.

Our aim is to minimize the average task duration with the limited battery capacity. To be specific, we defined the integer decision variable, $x_i \in \{0,1\}$ that indicates task Q_i is processed locally ($x_i = 1$) or in edge cloud ($x_i = 0$). Thus, The variables of task placement are as follows: $\mathrm{X} = \{x_1, x_2, \cdots, x_m\}$ and $\gamma = \{\gamma_1, \gamma_2, \cdots, \gamma_m\}$. The corresponding variable of resource allocation is as follows: $\kappa = \{\kappa_1, \kappa_2, \cdots, \kappa_m\}$. Table. II shows the notation in this paper. Formally, the task offloading problem can be formulated as follows:

$$\underset{\mathbf{x},\gamma,\kappa}{\text{minimize}} \ \sum_{i=1}^{m} \left[x_i t^L_i + (1-x_i) t^E_i\right] \tag{7}$$

$$\text{subject to: } x_i \varepsilon^L_i \le \alpha_i E^i_{max} \quad \forall i = 1, \cdots, m. \tag{8}$$

$$(1-x_i)\varepsilon^E_i \le \alpha_i E^i_{max} \quad \forall i = 1, \cdots, m. \tag{9}$$

$$\gamma_i \in \mathcal{A}(u_i) \quad \forall i = 1, \cdots, m. \tag{10}$$

$$\sum_{i \in o_{\gamma_i}} \kappa^{\gamma_i}_i \le 1. \tag{11}$$

The objective function (7) is to minimize the total task duration. The first constraint (8) indicates the local energy consumption is less than the remainder battery capacity of mobile device, where α_i is the remainder energy consumption relative to the total battery capacity E^i_{max}. The second constraint (9) states the energy consumption of transmission the task to edge cloud is limited by the battery of mobile device. The third constraint (10) indicates that the associated BS are those provide service to the user u_i. The last condition (11) indicates the amount of computation assigned to the γ_i should not exceed its total amount of computation, where o_{γ_i} denotes the set of computing task placed at edge cloud γ_i. Without consider the resource allocation, the task duration minimization problem (7) can be proved NP-hard according to [45].

IV. EFFICIENT TASK OFFLOADING SCHEME

In this section, we demonstrate an efficient task offloading scheme according to the optimization problem mentioned in (7). We first defined the objective function as follows:

$$f(\mathbf{x}, \kappa, \lambda) = \sum_{i=1}^{m} \left[x_i \frac{\omega_i}{f^l_i} + (1-x_i)\left(\frac{\omega_i}{\kappa^{\gamma_i}_i f^c_{\gamma_i}} + \frac{s_i}{r_{i,\gamma_i}}\right)\right] \tag{12}$$

For the sake of simplicity, we define $\lambda = (\mathbf{x}, \gamma)$. Let $\mathbb{K}$ and $\mathbb{H}$ denote the feasible sets for κ and λ. Problem in (7) is a mixed integer non-linear optimization problem. In this paper, we adopt alternative optimization techniques and consider two sub-problems as follows:

- *Computing resource allocation problem:* Given $\lambda = \lambda^0 \in \mathbb{H}$, i.e., when x_i and γ_i is fixed, the original optimization problem in (7) is a convex problem with respect to κ. Then, we adopt the Karush¨CKuhn¨CTucker (KKT) condition to obtain an optimal solution which is denoted by $f(\kappa^*, \lambda_0)$.
- *Task placement problem:* Based on the solution κ^*, the sub-problem $f(\kappa^*, \lambda)$ is a 0-1 integer programming problem with respect to λ. We adopt the task placement algorithm to obtain the solution.

Then, we provide a convergence proof of the problem. In this paper, we call this task offloading strategy as software defined task offloading (SDTO).

A. Computing Resource Allocation Problem

Lemma 1: Given $\lambda = \lambda^0 \in \mathbb{H}$, the original optimization problem in (7) with respect to κ_i is a convex optimization problem.

Proof: From the (7), we can get the resource allocation only when the task is offload to the edge cloud, i.e., $x_i = 0$. Given $\gamma = \gamma^0 = (\gamma^0_1, \gamma^0_2, \cdots, \gamma^0_m) \in \mathbb{H}$, we denote $j = \gamma^0_i$, the number of $x^0_i = 1$ is l, then the objective function (7) can be converted into the following equation:

$$f(\kappa, \gamma^0) = \sum_{i=1}^{l} \left(\frac{\omega_i}{\kappa^j_i f^c_j} + \frac{s_i}{r_{i,j}}\right) \tag{13}$$

where $f(\kappa, \gamma^0)$ is a function with respect to $\{\kappa_1, \kappa_2, \cdots, \kappa_m\}$. Then we can obtain its Hessian matrix as follows:

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial^2 \kappa_1} & \frac{\partial^2 f}{\partial \kappa_1 \partial \kappa_2} & \cdots & \frac{\partial^2 f}{\partial \kappa_1 \partial \kappa_m} \\ \frac{\partial^2 f}{\partial \kappa_2 \partial \kappa_1} & \frac{\partial^2 f}{\partial^2 \kappa_2} & \cdots & \frac{\partial^2 f}{\partial \kappa_2 \partial \kappa_m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 f}{\partial \kappa_m \partial \kappa_1} & \frac{\partial^2 f}{\partial \kappa_m \partial \kappa_2} & \cdots & \frac{\partial^2 f}{\partial^2 \kappa_m} \end{bmatrix}.$$

where each specific element is:

$$\frac{\partial^2 f}{\partial \kappa_i \partial \kappa_j} = \begin{cases} \frac{2\omega_i}{(\kappa_i^j)^3 f_j^c} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The parameters in (14) are all positive numbers, which indicates $\frac{\partial^2 f}{\partial \kappa_i \partial \kappa_j} \geq 0$. Thus, it is concluded that all eigenvalues of Hessian matrix $\mathbf{H}$ are positive numbers, and that $\mathbf{H}$ is a symmetrical positive definite matrix. In accordance with Theorems in [46], it is concluded that (7) is convex. Since constraint is linear, it can be concluded that this problem is a convex optimization problem. □

In accordance with lemma 1, Theorem 1 can be obtained with the KKT conditions.

Theorem 1: Given $\gamma = \gamma_0 \in \mathbb{H}$, the optimal value with respect to κ can be obtained, as shown in (17).

Proof: According to lemma 1, the Lagrange function of the original optimization problem about (7) and (8) can be obtained as follows:

$$L(\kappa, \nu) = \sum_{i=1}^{l} \left(\frac{\omega_i}{\kappa_i^j f_j^c} + \frac{s_i}{r_{ij}}\right) + \sum_j \nu_j \left(\sum_{i \in o_j} \kappa_i^j - 1\right) \tag{15}$$

By the use of the KKT conditions, if $\tilde{\kappa}, \tilde{\nu}$ is any point that satisfies the KKT conditions, we can conclude that:

$$\nabla f(\tilde{\kappa}_1^j, \tilde{\kappa}_2^j, \cdots, \tilde{\kappa}_n^j) + \sum_{j=1}^{n} \tilde{\nu}_j \nabla \left(\sum_{i \in o_j} \tilde{\kappa}_i^j - 1\right) = 0$$

$$\sum_{i \in o_j} \tilde{\kappa}_i^j - 1 = 0$$

By solving the formula above, we can obtain the optimal $\kappa_i^* = \tilde{\kappa}_i^j$ by:

$$\kappa_i^* = \frac{\sqrt{\omega_i}}{\sum_{i \in o_j} \sqrt{\omega_i}} \tag{16}$$

Since $\sum_{i=1}^{l} f(\kappa, \gamma^0) = \sum_{j=1}^{n} \sum_{i \in o_j} f(\kappa, \gamma^0)$, substitute (16) into (13), we can obtain the optimal value of $f(\kappa^*, \gamma^0)$ as follows:

$$\begin{aligned} f(\kappa^*, \gamma^0) &= \sum_{j=1}^{n} \sum_{i \in o_j} \left[\frac{\sqrt{\omega_i} \sum_{i \in o_j} (\sqrt{\omega_i})}{c_j} + \frac{s_i}{r_{i,j}}\right] \\ &= \sum_{j=1}^{n} \left[\frac{\left(\sum_{i \in o_j} \sqrt{\omega_i}\right)^2}{c_j} + \sum_{i \in o_j} \frac{s_i}{r_{i,j}}\right] \end{aligned} \tag{17}$$

□

B. Task Placement Problem

Through the above discussion, given $\lambda = \lambda^0 \in \mathbb{H}$, we can obtain the optimal solution in (17). Based on this, the original optimization problem is converted into integer programming problem with respect to $\mathbf{x}$ and γ as follows,

$$\underset{\mathbf{x}, \gamma}{\text{minimize}} \ f(\mathbf{x}, \kappa^*, \gamma) \tag{18}$$

$$\text{subject to: } x_i \varepsilon_i^L \leq \alpha_i E_{max}^i \quad \forall i = 1, 2, \cdots, m. \tag{19}$$

$$(1 - x_i) \varepsilon_i^E \leq \alpha_i E_{max}^i \quad \forall i = 1, 2 \cdots, m. \tag{20}$$

$$\gamma_i \in \mathcal{A}(u_i) \quad \forall i = 1, 2, \cdots, m. \tag{21}$$

In this section, we propose a task placement algorithm to solve this optimization problem.

First, we rewrite this optimization problem and transfer it into a 0-1 integer programming. Let $\mathbf{Z} = (z_{ij})_{m \times (n+m)}$, $z_{ij} \in \{0, 1\}$ denote the matrix to be solved out. Specifically, when $j = 1, 2, \cdots, n$, it indicate task is processed on edge cloud; when $j = n+1, n+2, \cdots, n+m$, it indicates that the task Q_i is locally processed. z_{ij} represents whether task Q_i is processed locally, or processed at edge cloud b_j. If $z_{ij} = 0$, the task Q_i will not be offloaded on edge cloud and processed locally. Likewise, if $z_{ij} = 1$, the task i will be offloaded on edge cloud or processed locally. Here, $\mathbf{Z}$ is analogous to γ and $\mathbf{x}$, i.e., aiming at the same task placement problem. Furthermore, we denoted variables as follows:

$$W = (\sqrt{\omega_1}, \sqrt{\omega_2}, \cdots, \sqrt{\omega_m})$$

$$S = (s_1, s_2, \cdots, s_m)$$

$$P = (p_1, p_2, \cdots, p_m)$$

$$\mathbf{C} = \begin{bmatrix} \frac{1}{\sqrt{f_1^c}} & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{\sqrt{f_2^c}} & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sqrt{f_{m-1}^l}} & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{\sqrt{f_m^l}} \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} r_{11} & \cdots & r_{1n} & 0 & \cdots & 0 \\ r_{21} & \cdots & r_{2n} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \\ r_{m1} & \cdots & r_{mn} & 0 & \cdots & 0 \end{bmatrix}_{m \times (n+m)}$$

Therefore, we can rewrite the optimization problem in (18) with respect to γ and x into optimization problem as follows:

$$\underset{\mathbf{Z}}{\text{minimize}} \ \|(W\mathbf{Z}\mathbf{C})^T\|_2^2 + S\mathbf{Z}\mathbf{R} \tag{22}$$

$$\text{subject to } x_{ij} \in \{0, 1\} \tag{23}$$

$$\sum_{j=1}^{n+1} x_{ij} = 1, \quad i = 1, 2, \cdots, m. \tag{24}$$

The objective function with respect to variable $\mathbf{Z}$ aims to reduce the task duration The constraint is that each task is placed on edge cloud or process locally. As for this problem,

assume there are n tasks to be processed, then there are 2^{n+1} choices to place task on, which is NP-hard.

In order to solve this problem, we define a continuous normal variable y_{ij}, which satisfies the following conditions:

$$y_{ij} \geq 0, \quad \sum_{j=1}^{n+1} y_{ij} = 1, \ i = 1, 2, \cdots, m.$$

Based on the variable y_{ij}, we denote a linear function as $\phi(y_{ij}) = \frac{y_{ij}}{y_{ij}^{t-1}+\epsilon}$, where ϵ is a quite small regularization constant and t is the number of iterations. Similar with the work in [47], we utilize linear function $\phi(y_{ij})$ to substitute X in the (22). Then, similar with Theorem 1, it is concluded that the modified objective function of (22) is a convex function. The specific solving algorithm is shown in Algorithm 1.

Algorithm 1 Task Placement Algorithm

Input:

Give initial task placement, y_{ij}^0;

Give a quite small error bound, δ;

Output:

Task placement, ϕ;

1: $t := 0$. $y_{ij}^0 := 1 - \epsilon$, where ϵ is a quite small regularization constant;

2: $t := t + 1$;

3: Assume the iterative result above is given as $\{y_{ij}^{t-1}\}$, define $\phi_{ij}^t(y_{ij}) = \frac{y_{ij}}{y_{ij}^{t-1}+\epsilon}$, $i = 1, 2, \cdots, m$, $j = 1, 2, \cdots, n+1$;

4: Substitute the variable $\phi_{ij}^t(y_{ij})$ obtained above for X in optimization problem, and $\{y_{ij}^t\}$ can be obtained through solving the modified optimization problem of Eq.(16).

5: If $|y_{ij}^t - y_{ij}^{t-1}| < \delta$, we can obtain $y_{ij}^* = y_{ij}^t$; else, go to step 2;

The key step in algorithm above is step 4. As for each iteration in step 4, we substitute function $\phi_{ij}^t(y_{ij}) = \frac{y_{ij}}{y_{ij}^{t-1}+\epsilon}$ for variable x_{ij} in the objective function of (22) and replace function y_{ij} for x_{ij} in the condition of (23) and (24). As discussed above, we can conclude that the modified objective function is a convex, and that modified condition are linear. Thus, the modified problem is a convex optimization problem which can be solved.

C. Convergence Analysis

In this section, the convergence of Algorithm 1 will be analyzed. Based on the step 4 at Algorithm 1, we can obtain

$$y_{ij}^t = \underset{y_{ij}}{\operatorname{argmin}} f(\frac{y_{ij}}{y_{ij}^{t-1} + \delta})$$

According to the global convergence theorem in [48], Algorithm 1 is converged. However, there is a question: why the solution of the modified optimization problem in Algorithm 1 is approximately equal to the solution of the problem of (16). Now, we give proof that function $\phi_{ij}^t(y_{ij})$ is equal to variable x_{ij} if $|y_{ij}^t - y_{ij}^{t-1}| < \delta$. Since δ is a minimal positive number, we can obtain that $y_{ij}^{t-1} \approx y_{ij}^t = y_{ij}^*$. Thus, we can obtain $\phi_{ij}^t(y_{ij}^*)$ as follows:

$$\phi_{ij}^t(y_{ij}^*) = \frac{y_{ij}^*}{y_{ij}^{t-1} + \epsilon} \approx \begin{cases} 1 & \text{if } y_{i,j} > 0 \\ 0 & \text{if } y_{ij} = 0 \end{cases}$$

Therefore, we can obtain that $\phi_{ij}^t(y_{ij}^*)$ is approximately equal to $\mathbf{Z}$, i.e., the solution of the modified optimization problem is approximately equal to the solution of the original problem.

Secondly, the solution to the two sub-problems is also applicable for the original optimization problem. $f(\kappa, \lambda)$ is the original optimization function. Given $\lambda = \lambda^0 \in \mathbb{H}$, based on the discussion in Section V(A), $f(\kappa, \gamma_0)$ is convex in κ. Thus, $\exists\ \kappa^* \in \mathbb{K}$, and the following inequality holds

$$f(\kappa^*, \lambda^0) \leq f(\kappa, \lambda^0)$$

Given $\kappa = \kappa^* \in \mathbb{K}$, according to the discussion in Section V(B), $f(\kappa^*, \lambda)$ is 0-1 integer programming with respect to λ. Based on Algorithm 1, for $\exists\ \lambda^* \in \mathbb{H}$, the following inequality can be obtained

$$f(\kappa^*, \lambda^*) \leq f(\kappa^*, \lambda)$$

Based on above two inequalities, for $\forall \kappa \in \mathbb{K}$ and $\lambda \in \mathbb{H}$, the following inequality holds

$$f(\kappa^*, \lambda^*) \leq f(\kappa, \lambda^*) \leq f(\kappa, \lambda)$$

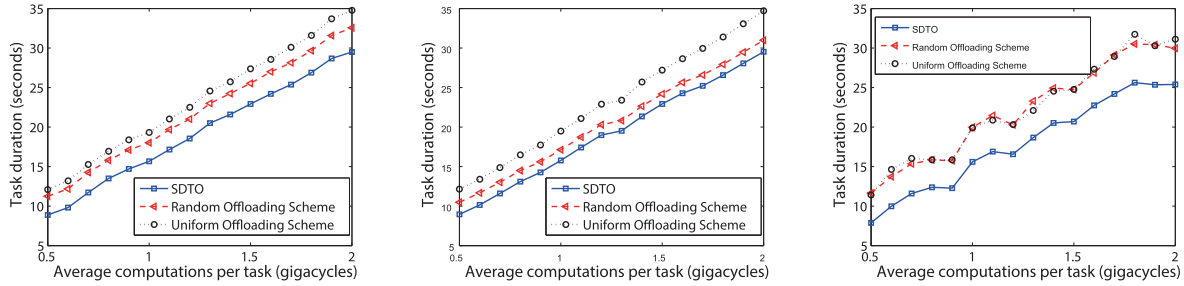
Thus, $f(\kappa^*, \lambda^*)$ can be obtained through solving the above two sub-problems, and also serves as the efficient solution for the original optimization problem.

V. Performance Analysis

In this section, a simulation experiment is provided concerning task offloading for mobile computing in SD-UDN. The experimental results are divided into three parts: (i) we compare the proposed SDTO with several task offloading schemes in terms of task duration and energy cost; (ii) we study the impact of task computation amount on the evaluated metrics; (iii) we investigate the impact of task data size on the performance of task offloading.

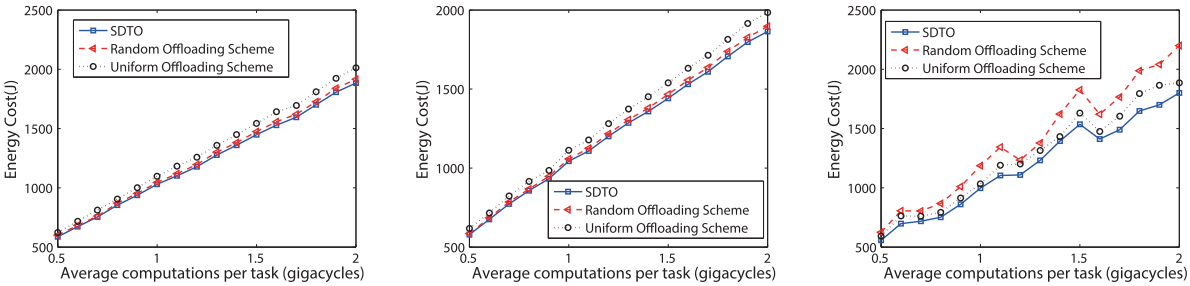
A. Experiment Setup

For task Q_i, we assume computation amount ω_i and data size s_i are generated by a probability distribution. The total computing resource of edge cloud is 25 GHz, while the total computing resource of mobile device is 10 GHz. Then, the processing time is $1/20$ second when the computation task is processed by edge cloud and $1/10$ second when the computation task is processed by mobile device. To mimic a ultra-dense network environment, we simulate a $500\ m \times 500\ m$ square area with 15 BSs. The users can connect with BSs within 100m. Similar as the work in [36], we model the channel gain as follows: $127 + 30 \times \log d$, the other settings of the simulation parameters are given in Table III.



(a) Computation amount: Normal distribution (b) Computation amount: Uniform distribution. (c) Computation amount: Pareto distribution.

Fig. 2. Impact of computation amount on task duration.



(a) Computation amount: Normal distribution (b) Computation amount: Uniform distribution. (c) Computation amount: Pareto distribution.

Fig. 3. Impact of computation amount on energy cost.

TABLE III
SIMULATION PARAMETERS

Parameters	Value
Number of Task	500
Transmission bandwidth of mobile device, B	20 MHz
Transmit power of user mobile device, p_i	0.5 W
Gaussian channel noise, σ^2	2×10^{-13} W
Channel power gain, $h_{i,j}$	$127 + 30 \times \log d$
Computing energy consume in edge cloud, ρ	90 W/Gigacycles
The total battery capacity, E_{max}	1000J

B. Comparison With Other Methods

The SDTO, proposed in this paper, is compared with two different offloading strategies: *random offloading scheme* and *uniform offloading scheme*.

- Random offloading scheme: (i) Task placement, the computation tasks are offloaded to edge cloud for processing or processed locally randomly. We set up a random generator that can generate the number 0 and 1 with equal probability, and then we offload computation tasks to be processed onto edge cloud or process locally randomly according to the random generator. (ii) Resource allocation, based on the random task placement strategy, the resource allocation is given by using the (17).
- Uniform offloading scheme: (i) Task placement, we divide all the tasks into two parts according the user's battery capacity, one part in the edge cloud processing and the other part is local processing. (ii) Resource allocation, similar to random offloading scheme, based on the uniform task placement strategy, the resource allocation is given by using the (17).

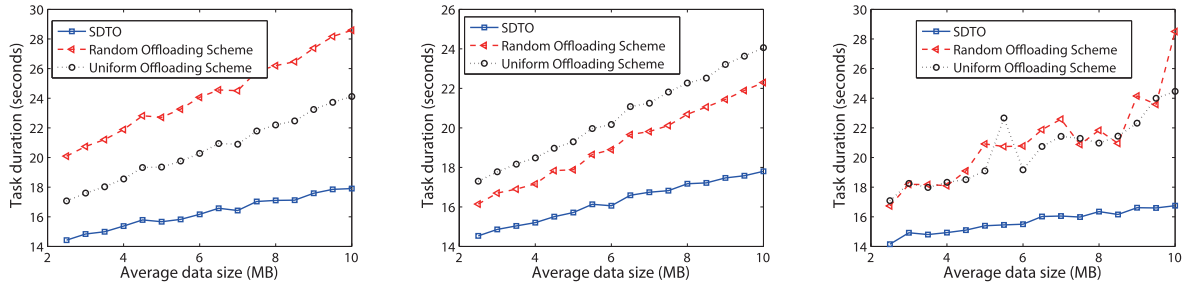
For the three methods above, we evaluate offloading performance in terms of task duration and energy cost. We first give the average utilization rate of the above three kinds of offloading schemes, the resource utilization rate of the random offloading scheme, uniform offloading scheme and the SDTO scheme are 75%, 80% and 92%, respectively.

C. Impact of Computation Amount on Offloading Performance

We first consider the impact of computation amount on offloading performance. The data size of task follows a normal distribution with mean value of 5 MB. As for the computation amount, three kinds of distribution are utilized, i.e., uniform distribution, normal distribution and Pareto distribution.

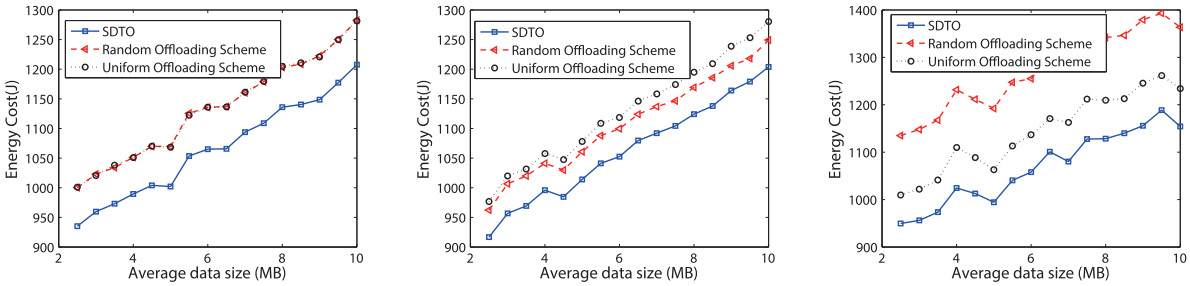
As shown in Fig. 2 and Fig. 3, we can conclude that the larger the computation amount of a task, the longer task duration is caused with more energy cost. Furthermore, our proposed SDTO exhibits shorter task duration and lower energy cost than random offloading scheme and uniform offloading scheme. This is because the efficient offloading scheme is obtained by comprehensively considering the computation amount and data size of task content, so as to minimize task duration and energy cost.

In Fig. 2(a) and Fig. 3(a), when the computation amount follows normal distribution, compared with random offloading



(a) Data size: Normal distribution (b) Data size: Uniform distribution. (c) Data size: Pareto distribution.

Fig. 4. Impact of data size on task duration.



(a) Data size: Normal distribution (b) Data size: Uniform distribution. (c) Data size: Pareto distribution.

Fig. 5. Impact of data size on energy cost.

scheme, it is shown that our proposed SDTO can reduce 30% task duration and 10% energy cost. And compared with uniform offloading scheme, our scheme can reduce 41% of the task duration and 11% of the energy cost. As shown in Fig. 2(b) and Fig. 3(b), when the computation amount follows uniform distribution, the results are similar with the case based on normal distribution. In Fig. 2(c) and Fig. 3(c), when the computation amount follows Pareto distribution, it is observed the curve trend is not as smooth as that in normal distribution and uniform distribution, which is because the Pareto distribution has the long tail effect, and we set the interval as 0.1 when we generate the Pareto distribution.

D. Impact of Data Size on Computation Offloading

In this section, we consider the impact of data size on computation offloading. The computation amount of task follows a normal distribution with mean value of 1 Gigacycle. For data size, three kinds of distributions are utilized, i.e., uniform distribution, normal distribution and Pareto distribution.

As shown in Fig. 4 and Fig. 5, we can conclude that the larger the data size of the task, the longer task duration and the more energy cost. We can further conclude that our proposed offloading scheme exhibits shorter task duration and lower energy cost than random offloading scheme and uniform offloading scheme.

In Fig. 4(a) and Fig. 5(a), when computation amount follows normal distribution, compared with random offloading scheme, it is shown that our proposed offloading scheme can reduce 30% of the task duration and 5% of the energy cost. And compared with uniform offloading scheme, our scheme can reduce 28% of the task duration and 5% of the energy cost. As shown in Fig. 4(b) and Fig. 5(b), when computation amount follows uniform distribution, the results are similar with the case of normal distribution. In Fig. 4(c) and Fig. 5(c), when computation amount follows Pareto distribution, we see that the curve trend is not as smooth as that in normal distribution and uniform distribution, which is because the Pareto distribution has the long tail effect, and we set the interval at 0.5 when we generate the Pareto distribution. Throughout the above analysis, we can also conclude that the effect of data size on task duration and energy cost is lower than that of computation amount.

VI. Conclusion

In this paper, we first propose the architecture of software defined ultra dense network (SD-UDN). Then, we propose a scheme to offload task on edge cloud or process locally. In order to minimize the task duration, computing resource is optimally allocated to each task. To the best of our knowledge, this is the first study of task offloading for mobile edge computing in SD-UDN. Simulation results have shown that our proposed scheme is more efficient compared to the random and uniform computation offloading schemes. In future work, we will consider task offloading in more complicate deployment with users mobility.

References

[1] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2016–2021 White Paper*, Feb. 2017. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf

[2] Z. Chen *et al.*, "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance," in *Proc. 2nd ACM/IEEE Symp. Edge Comput.*, Oct. 2017, pp. 1–14.

[3] T. Taleb, A. Ksentini, M. Chen, and R. Jantti, "Coping with emerging mobile social media applications through dynamic service function chaining," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2859–2871, Apr. 2016.

[4] D. Tian, J. Zhou, Z. Sheng, and V. C. Leung, "Robust energy-efficient MIMO transmission for cognitive vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3845–3859, Jun. 2016.

[5] D. Tian, J. Zhou, Z. Sheng, M. Chen, Q. Ni, and V. C. M. Leung, "Self-organized relay selection for cooperative transmission in vehicular ad-hoc networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9534–9549, Oct. 2017.

[6] Y. Li, D. Jin, P. Hui, and Z. Han, "Optimal base station scheduling for device-to-device communication underlaying cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 27–40, Jan. 2016.

[7] H. Wang, F. Xu, Y. Li, P. Zhang, and D. Jin, "Understanding mobile traffic patterns of large scale cellular towers in urban environment," in *Proc. ACM Conf. Internet Meas. Conf.*, 2015, pp. 225–238.

[8] Y. Zhang, M. Qiu, C.-W. Tsai, M. M. Hassan, and A. Alamri, "Health-CPS: Healthcare cyber-physical system assisted by cloud and big data," *IEEE Syst. J.*, vol. 11, no. 1, pp. 88–95, Mar. 2017.

[9] Y. Zhang, R. Yu, M. Nekovee, Y. Liu, S. Xie, and S. Gjessing, "Cognitive machine-to-machine communications: Visions and potentials for the smart grid," *IEEE Netw.*, vol. 26, no. 3, pp. 6–13, May/Jun. 2012.

[10] Y. Zhang, M. Chen, N. Guizani, D. Wu, and V. C. Leung, "SOVCAN: Safety-oriented vehicular controller area network," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 94–99, Aug. 2017.

[11] Y. Zhang, Z. Tu, and Q. Wang, "TempoRec: Temporal-topic based recommender for social network services," *Mobile Netw. Appl.*, vol. 22, no. 6, pp. 1182–1191, 2017.

[12] M. Chen, Y. Hao, M. Qiu, J. Song, D. Wu, and I. Humar, "Mobility-aware caching and computation offloading in 5G ultra-dense cellular networks," *Sensors*, vol. 16, no. 7, p. 974, 2016.

[13] T. Taleb, A. Ksentini, and R. Jantti, "'Anything as a service' for 5G mobile systems," *IEEE Netw.*, vol. 30, no. 6, pp. 84–91, Nov. 2016.

[14] M. Chen, Y. Zhang, L. Hu, T. Taleb, and Z. Sheng, "Cloud-based wireless network: Virtualized, reconfigurable, smart wireless network to enable 5G technologies," *Mobile Netw. Appl.*, vol. 20, no. 6, pp. 704–712, Dec. 2015.

[15] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[16] S. Zhang, N. Zhang, S. Zhou, J. Gong, Z. Niu, and X. Shen, "Energy-aware traffic offloading for green heterogeneous networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1116–1129, May 2016.

[17] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. 35th IEEE Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[18] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[19] M. Chen, Y. Hao, Y. Li, C.-F. Lai, and D. Wu, "On the computation offloading at ad hoc cloudlet: Architecture and service modes," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 18–24, Jun. 2015.

[20] M. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? The bandwidth and energy costs of mobile cloud computing," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1285–1293.

[21] 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.

[22] C. Gao, Y. Li, Y. Zhao, and S. Chen, "A two-level game theory approach for joint relay selection and resource allocation in network coding assisted D2D communications," *IEEE Trans. Mobile Comput.*, vol. 16, no. 10, pp. 2697–2711, Oct. 2017.

[23] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[24] A. Ksentini, M. Bagaa, and T. Taleb, "On using SDN in 5G: The controller placement problem," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.

[25] A. Ksentini, M. Bagaa, T. Taleb, and I. Balasingham, "On using bargaining game for optimal placement of SDN controllers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[26] M. Bagaa, T. Taleb, and A. Ksentini, "Service-aware network function placement for efficient traffic handling in carrier cloud," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2014, pp. 2402–2407.

[27] B. E. Mada, M. Bagaa, and T. Taleb, "Efficient transcoding and streaming mechanism in multiple cloud domains," in *Proc. IEEE Global Commun. Conf. GLOBECOM*, Dec. 2017, pp. 1–6.

[28] I. Benkacem, T. Taleb, M. Bagaa, and H. Flinck, "Performance benchmark of transcoding as a virtual network function in CDN as a service slicing," in *Proc. IEEE WCNC*, Barcelona, Spain, Apr. 2018, pp. 1–6.

[29] D. L. C. Dutra, M. Bagaa, T. Taleb, and K. Samdanis, "Ensuring end-to-end QoS based on multi-paths routing using SDN technology," in *Proc. IEEE GLOBECOM*, Singapore, Dec. 2017, pp. 1–6.

[30] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proc. 6th Conf. Comput. Syst.*, 2011, pp. 301–314.

[31] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 1350–1354.

[32] T. Taleb and A. Ksentini, "An analytical model for follow me cloud," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 1291–1296.

[33] M. Chen, Y. Hao, L. Hu, K. Huang, V. Lau, "Green and mobility-aware caching in 5G networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 12, pp. 8347–8361, 2017.

[34] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Trans. Mobile Comput.*, vol. 15, no. 10, pp. 2398–2410, Oct. 2016.

[35] 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.

[36] 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.

[37] Y. Xiao and M. Krunz, "QoE and power efficiency tradeoff for fog computing networks with fog node cooperation," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.

[38] T. X. Tran and D. Pompili. (2017). "Joint task offloading and resource allocation for multi-server mobile-edge computing networks." [Online]. Available: https://arxiv.org/abs/1705.00704

[39] Y. Li and W. Wang, "Can mobile cloudlets support mobile applications?" in *Proc. IEEE INFOCOM*, Apr. 2014, pp. 1060–1068.

[40] C. Wang, Y. Li, and D. Jin, "Mobility-assisted opportunistic computation offloading," *IEEE Commun. Lett.*, vol. 18, no. 10, pp. 1779–1782, Oct. 2014.

[41] T. Truong-Huu, C.-K. Tham, and D. Niyato, "A stochastic workload distribution approach for an ad hoc mobile cloud," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2014, pp. 174–181.

[42] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: A survey," *IEEE Commun. Mag.*, vol. 51, no. 11, pp. 24–31, Nov. 2013.

[43] S. Retal, M. Bagaa, T. Taleb, and H. Flinck, "Content delivery network slicing: QoE and cost awareness," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[44] 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.

[45] Y. Pochet, L. A. Wolsey, *Production Planning by Mixed Integer Programming*. Berlin, Germany: Springer, 2006.

[46] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[47] Y. Liu, D. Niu, and B. Li, "Delay-optimized video traffic routing in software-defined interdatacenter networks," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 865–878, May 2016.

[48] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*, vol. 28. Reading, MA, USA: Addison-Wesley, 1973.

Min Chen (SM'09) was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University (SNU). He was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of British Columbia for three years and also with SNU for one and a half years. He has been a Full Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology since 2012, where he is currently the Director of the Embedded and Pervasive Computing Lab. He has authored over 300 papers, including over 200 SCI papers, over 80 IEEE Transactions/journal papers, 16 ISI highly cited papers, and eight hot papers. He has authored four books: *OPNET IoT Simulation* (HUST Press, 2015), *Big Data Inspiration* (HUST Press, 2015), *5G Software Defined Networks* (HUST Press, 2016), and *Introduction to Cognitive Computing* (HUST Press, 2017), a book on big data: *Big Data Related Technologies* (2014), and a book on 5G: *Cloud Based 5G Wireless Networks* (2016) with Springer Series in computer science. His latest book (co-authored with Prof. K. Hwang) on *Big Data Analytics for Cloud/IoT and Cognitive Computing* (U.K.: Wiley, 2017). His Google Scholars Citations reached over 11 800 with an h-index of 53. His top paper was cited over 1200 times. His research interests include cyber physical systems, IoT sensing, 5G networks, mobile cloud computing, SDN, healthcare big data, medica cloud privacy and security, body area networks, emotion communications, and robotics. He received the IEEE Communications Society Fred W. Ellersick Prize in 2017. He received the Best Paper Award from QShine 2008, the IEEE ICC 2012, ICST IndustrialIoT 2016, and the IEEE IWCMC 2016. He is the Chair of the IEEE Computer Society Special Technical Communities on Big Data. He is the Co-Chair of the IEEE ICC 2012-Communications Theory Symposium and the IEEE ICC 2013-Wireless Networks Symposium. He is the General Co-Chair for the IEEE CIT-2012, Tridentcom 2014, Mobimedia 2015, and Tridentcom 2017. He is a Keynote Speaker for CyberC 2012, Mobiquitous 2012, Cloudcomp 2015, IndustrialIoT 2016, and The 7th Brainstorming Workshop on 5G Wireless. He serves as an Editor or Associate Editor for the *Information Sciences*, *Information Fusion*, and the IEEE ACCESS. He is a Guest Editor for the IEEE NETWORK, the IEEE WIRELESS COMMUNICATIONS, and the IEEE TRANSACTIONS ON SERVICE COMPUTING.

Yixue Hao received the B.E. degree from Henan University, China, and the Ph.D. degree in computer science from the Huazhong University of Science and Technology, China, in 2017. He is currently a Post-Doctoral Scholar with the School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include 5G network, Internet of Things, edge caching, and mobile edge computing.