Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds

Xinchen Lyu, Hui Tian, Cigdem Sengul, and Ping Zhang, *Senior Member, IEEE*

***Abstract*—Proximate cloud computing enables computationally intensive applications on mobile devices, providing a rich user experience. However, remote resource bottlenecks limit the scalability of offloading, requiring optimization of the offloading decision and resource utilization. To this end, in this paper, we leverage the variability in capabilities of mobile devices and user preferences. Our system utility metric is a measure of quality of experience (QoE) based on task completion time and energy consumption of a mobile device. We propose a heuristic offloading decision algorithm (HODA), which is semidistributed and jointly optimizes the offloading decision, and communication and computation resources to maximize system utility. Our main contribution is to reduce the problem to a submodular maximization problem and prove its NP-hardness by decomposing it into two subproblems: 1) optimization of communication and computation resources solved by quasiconvex and convex optimization and 2) offloading decision solved by submodular set function optimization. HODA reduces the complexity of finding the local optimum to $O(K^3)$, where K is the number of mobile users. Simulation results show that HODA performs within 5% of the optimal on average. Compared with other solutions, HODA's performance is significantly superior as the number of users increases.**

***Index Terms*—Mobile cloud computing, multiuser offloading, proximate cloud, resource optimization.**

I. Introduction

RESOURCE-INTENSIVE mobile applications, such as e-Health, face recognition, natural language processing, interactive gaming, and augmented reality, are fast developing and increasingly outgrowing the limited capabilities of mobile devices [1]. By offloading the computation and storage to the resource providers, mobile cloud computing can bridge the gap between limited capabilities of mobile devices and increasing demand of resource-intensive applications [2].

However, offloading incurs extra overhead due to the communication required between the devices and the cloud. The additional communication affects both energy consumption and latency, and consequently, the offloading decision becomes an issue [3]. In [4]–[8], the problem was addressed by choosing between local or remote execution, where each user decides to offload or not independently of others. It is assumed that the cloud always has enough resources to accommodate the offloaded tasks without delay, regardless of how many tasks are concurrently executed. Recently, finite cloud resources have been taken into account, along with a large number of mobile users in [9]. In the case of multiple users, solutions vary such that some only optimize the offloading decision [9], [10]; some only optimize communication resources [11], [12]; and others only optimize computation resources [13], [14]. In the case of jointly optimizing communication and computation resources, in [15] and [16], it is assumed that all tasks are offloaded.

In this paper, we consider the task offloading problem in proximate clouds, which bring computation resources to the edge network to enrich user experience. Proximate clouds may either be macro/pico/femto base stations (BSs) connected to colocated computing centers, or intelligent BSs with computation capabilities, but may have scarce resources. Therefore, offloading bottlenecks are expected as the number of offloaded mobile users increases. Moreover, during the competition for limited resources, the offloading competition among multiple users makes the offloading decision a challenging problem when coupled with resource optimization. Specifically, resource optimization depends on the results of offloaded users, while mobile users can choose to offload or execute tasks locally according to the results of resource allocation. In this paper, we jointly optimize the offloading decision (i.e., which users should be offloaded and which users should use local execution) and resource utilization.

Our solution takes into account the inherent heterogeneity across mobile devices and computation tasks. We design a quality of experience (QoE)-based utility function, which measures the utility of task completion time and energy consumption of offloading compared with local execution. Then, we formulate the problem as a system utility maximization problem, which jointly optimizes the offloading decision, and communication and computation resources. We reduce the problem to a submodular maximization problem and prove its NP-hardness by decomposing it into two subproblems: 1) optimization of communication and computation resources and 2) offloading decision. Communication and computation resources are optimized through convex and quasiconvex optimization. Based on the results of resource optimization, we prove the system utility to be a submodular set function in terms of the offloading decision and compute the local optimum through submodular optimization. Our algorithm, i.e., heuristic offloading decision algorithm (HODA), is semidistributed and runs in two stages. In the first stage, each mobile user independently optimizes transmission power and determines whether to send an offloading

Manuscript received December 11, 2015; revised March 22, 2016, May 21, 2016, and July 4, 2016; accepted July 6, 2016. Date of publication July 20, 2016; date of current version April 14, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61471060 and Grant 61421061 and in part by the Huawei Innovation Research Program. The review of this paper was coordinated by Prof. C. Assi. (*Corresponding author: Hui Tian.*)

X. Lyu, H. Tian, and P. Zhang are with Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: tianhui@bupt.edu.cn).

C. Sengul is with Nominet, Oxford OX4 4DQ, U.K.

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/TVT.2016.2593486

request. These offloading requests include the information on mobile device features, user preferences, and the properties of computation task. In the second stage, the macrocell forms a locally optimal offloading set by prioritizing users with maximum utility. Finally, the selected mobile users offload their computation tasks.

HODA reduces the complexity of finding the local optimum to $O(K^3)$, compared with the $O(2^K)$ complexity of the optimal solution, where K is the number of mobile users. As the number of mobile users increases, which results in the shortage of resources, HODA is able to manage the bottleneck by selecting the users that will benefit most from offloading, and its performance is within 5% of the optimal on average. In contrast, the solutions that execute all tasks locally, offload all users while optimizing communication and computation resources [15], [16], or independently select tasks to offload perform significantly worse, compared with HODA.

The rest of this paper is organized as follows. We discuss the related works in Section II and introduce the system model in Section III. Section IV formulates the problem as a system utility maximization problem. We reduce the problem to a submodular maximization problem in Section V and then propose a semidistributed HODA in Section VI. Section VII evaluates the efficiency of HODA. Finally, Section VIII concludes this paper.

II. Related Works

Mobile cloud computing uses either distant or proximate clouds to enhance computing capabilities of mobile devices, to support mainly resource-intensive applications [17]. For distant clouds (e.g., Amazon Elastic Compute Cloud), the computation units are distributed in the core network with enormous computation resources, whereas proximate clouds provide more limited resources in proximity to users. However, distant cloud computing consumes backhaul resources in the core network and, therefore, faces the challenges of fluctuating latency and limited capacity [18], [19]. Mobile users are sensitive to delay and jitter, which are difficult to constrain in the core network [20]. Therefore, the Cloudlet concept, which involves a proximate cloud computing architecture, addresses these challenges by reducing offloading latency and minimizing security risks [20], [21]. Compared with the distant cloud, a proximate cloud has also the advantage of using less of the backhaul resources. Cloudlets can be located near the mobile users, e.g., intelligent macro BSs, small-cell BSs, or wireless access points with access to a local computing server [20], [21].

Regardless of its proximity, offloading tasks for remote computing will consume communication and computation resources. Therefore, there exists a tradeoff between the number of offloaded tasks by multiple users and the user experience of remote computation. Current approaches propose efficient offloading policies from the perspective of a single mobile user such that all of the remote computation resources are allocated to this user [4]–[8]. For instance, in [4], the solver decides whether an entire application should be offloaded to a cloud server or executed by the mobile device. In contrast, others partition the application into tasks or code blocks [5]–[8]. In [5] and [6], a mobile application is represented by a sequence of tasks with a linear topology, and the minimum-energy task scheduling problem is solved. In [6], this is taken a step further by taking into account channel variability and its effect on task deadlines. In [7], instead of a linear topology, a directed graph is used to represent the code blocks, and a genetic algorithm is developed to decide how to offload them among multiple devices. In [8], we propose an adaptive receding-horizon offloading strategy among multiple devices, where the solver can adjust its offloading decision according to environmental dynamics (e.g., fluctuating latency).

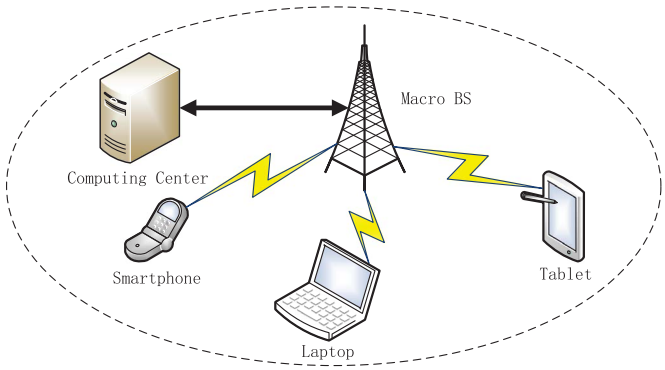


Fig. 1. Multiuser proximate cloud scenario.

Task offloading with multiple mobile users is considered in [9]–[16]. Yang *et al.* in [9] and Cardellini *et al.* in [10] focused on the offloading decision problem among multiple users without optimizing communication and computation resources. In [9], under the constraint of limited cloud resources, both online and offline algorithms are developed for computation partitioning over multiple users. In [10], the offloading decision problem is formulated as a generalized Nash equilibrium problem over a three-tier architecture consisting of local mobile devices, proximate cloud, and the distant cloud. Other works focus on optimizing only computation resources, but may not take into account the effect of communication resources on offloading [13], [14]. For instance, Yang *et al.* [13] partitioned a data stream application such that the data processing speed is maximized, allowing the sharing of computation instances among multiple users. In [14], the mobile applications are modeled as location–time workflows based on user mobility. In contrast, limited communication resources in a wireless interference environment are taken into account in [11] and [12], but infinite computational resources and constant transmission power are assumed. In [15] and [16], communication and computation resources are jointly optimized, however assuming that all tasks are offloaded for remote computation. Therefore, this approach is not expected to scale as the number of users and tasks increases. We confirm this in our simulation study, which also serves as the motivation for optimizing offloading decisions and resources jointly, as we propose in this paper.

III. System Model

In this paper, we consider a multiuser proximate cloud scenario shown in Fig. 1. In this scenario, the single Long-Term Evolution (LTE) macro BS has a computing center with limited computation resources and serves K mobile devices in

its range. These mobile devices may have different computation and energy resources, such as in the case of smart phones, tablets, and laptops.

Each mobile user i has a computation task CT_i [11], [12], which can be described in terms of

1) the input D_i (bit), including system settings, program codes, and input parameters;
2) the number of CPU cycles required to accomplish the computation task, C_i;
3) the output (e.g., the computation result).

Tasks are atomic and cannot be divided into subtasks. Since the size of the output is generally much smaller than the input, we omit it in our computation as in [5], [7], [11], and [12]. The information on D_i and C_i can be obtained by applying program profilers (e.g., as in [6]–[9] and [11]–[13]).[1]

Each computation task can be either executed locally or remotely in a macro computing center.

A. Local Computation

F_i^l denotes the local computation capability of mobile user i in terms of instructions per second. T_i^l is the task completion time and can be written as

$$T_i^l = \frac{C_i}{F_i^l}. \tag{1}$$

According to [11], [12], and [22], CPU power consumption is a superlinear function of execution frequency and is given as

$$P_i^l = \alpha \cdot \left(F_i^l\right)^{\gamma} \tag{2}$$

where P_i^l denotes the local power consumption of mobile user i, and typically, $\alpha = 10^{-11}$, and $\gamma = 2$ [11], [12].

The energy consumption of mobile user i, E_i^l (J) becomes

$$E_i^l = P_i^l \cdot T_i^l = \alpha \left(F_i^l\right)^{\gamma-1} C_i. \tag{3}$$

It should be noted that both T_i^l and E_i^l are determined only by F_i^l and C_i, which are intrinsic features of a mobile device and its computation task and, therefore, are known.

B. Remote Computation

A typical remote computation process consists of three stages.

1) The mobile user uploads CT_i to the macro BS through the uplink channel.
2) The macro computing center allocates f_i computation resources to the task and then executes it on behalf of the mobile user.
3) The macro BS transmits output data back to the mobile user.

As previously mentioned, we ignored the overhead of the output data in the last stage [5], [7], [11], [12].

[1]The program profiler monitors the program parameters, for instance, execution time, acquired memory, thread CPU time, number of instructions, and method calls [2].

Compared with the local computation, remote computation saves mobile users' energy and computation resources for task execution, but spends additional time and energy in uplink transmission. For the uplink transmission, the intracell interference is well mitigated in the LTE network.[2] Therefore, the data rate of user i can be given as [24]

$$R_i(p_i) = W \log_2\left(1 + \frac{p_i h_i}{N_0}\right) \tag{4}$$

where W (Hz) is the user bandwidth, and p_i denotes the transmission power, which can be configured by the mobile user subject to a maximum transmission power constraint [11], [12], [15], [16]. B (Hz) is the system bandwidth, and therefore, at most, $N = B/W$ mobile users are allowed to transmit at the same time. h_i denotes the channel gain from mobile user i to the macro BS including path loss and fading, and N_0 is the noise power. According to (4), mobile user i can adjust its data rate from 0 to $W \log_2(1 + p_0 h_i/N_0)$ by controlling its transmission power, where p_0 denotes maximum transmission power allowed.

The total remote computation completion time for mobile user i, i.e., $T_i^r(f_i, p_i)$, is composed of two parts. Thus

$$T_i^r(f_i, p_i) = T_i^t(p_i) + T_i^e(f_i) \tag{5}$$

where $T_i^t(p_i)$ and $T_i^e(f_i)$ are the uplink transmission time and the remote execution time, respectively. $T_i^t(p_i)$ depends on the size of input and the data rate R_i (b/s) and can be given as

$$T_i^t(p_i) = \frac{D_i}{W \log_2(1 + a_i p_i)} \tag{6}$$

where $a_i = h_i/N_0$. Similar to (1), the remote execution time $T_i^e(f_i)$ can be obtained as

$$T_i^e(f_i) = \frac{C_i}{f_i} \tag{7}$$

where $p_i \neq 0$ and $f_i \neq 0$; hence, infinite transmission and execution times are avoided.

The energy consumption of mobile user i for remote computation is E_i^r (J). We only consider the energy consumption of the upload, since the energy consumption for task execution at the mobile is saved through offloading [11], [12], i.e.,

$$E_i^r(p_i) = \frac{p_i}{\zeta} \cdot T_i^t(p_i) = \frac{p_i}{\zeta} \frac{D_i}{W \log_2(1 + a_i p_i)} \tag{8}$$

where ζ is the power amplifier efficiency.

IV. Problem Formulation

In a mobile cloud computing system, the QoE of user i is determined mainly by task completion time, i.e., T_i, and energy

[2]LTE adopts the single-carrier frequency-division multiple access (SC-FDMA) for the uplink transmission, which orthogonalizes different users' transmissions in the same cell by explicit assignments of groups of discrete-Fourier-transform-precoded orthogonal subcarriers [23].

consumption, i.e., E_i. Specifically, T_i and E_i can be obtained as

$$T_i = s_i \cdot T_i^r + (1 - s_i) \cdot T_i^l \tag{9}$$

$$E_i = s_i \cdot E_i^r + (1 - s_i) \cdot E_i^l \tag{10}$$

where $s_i \in \{0, 1\}$ denotes the offloading decision for CT_i (i.e., the task is offloaded when $s_i = 1$).

The user preferences on task completion time and energy consumption, i.e., $\beta_i^T, \beta_i^E \in [0, 1]$, can be determined by the remaining battery life and the requirement of task completion time. For instance, a mobile user i of short battery life can increase β_i^E and decrease β_i^T, so as to save more energy at the expense of longer task completion time. Taking user preferences into account, we define the utility function of mobile user i as

$$\begin{aligned} v_i(s_i, f_i, p_i) &= \beta_i^T \frac{T_i^l - T_i}{T_i^l} + \beta_i^E \frac{E_i^l - E_i}{E_i^l} \\ &= s_i \left(\beta_i^T \frac{T_i^l - T_i^r}{T_i^l} + \beta_i^E \frac{E_i^l - E_i^r}{E_i^l} \right). \end{aligned} \tag{11}$$

Note that the user utility, i.e., $v_i(s_i, f_i, p_i)$, measures the improvement in QoE by offloading compared with local execution, where the performance improvement in time and energy consumption can be given as $(T_i^l - T_i)/T_i^l$ and $(E_i^l - E_i)/E_i^l$, respectively. When user i executes its task locally (i.e., $s_i = 0$), its utility v_i is equal to 0. Meanwhile, offloading too many tasks for remote computation will lead to longer remote execution times, due to limited bandwidth and remote computation resources. In this case, the utility function can take negative values, if remote task completion time is much longer than local execution.

Similarly, the resource providers have preferences on different mobile users, i.e., $\rho_i \in [0, 1]$ (e.g., based on the payments offered by the mobile users). For instance, the resource providers can prioritize the users with higher revenues for offloading by increasing the corresponding preferences. The system utility is defined as $\sum_{i=1}^{|\mathbf{K}|} \rho_i v_i(s_i, f_i, p_i)$, which not only measures the overall utility of mobile users but considers the interest of the resource provider as well.

Hence, sharing of remote resources for task offloading becomes a system utility maximization problem. We formulate the problem as follows:

$$\mathbf{P}: \begin{aligned} &\max_{\mathbf{s},\mathbf{f},\mathbf{p}} \sum_{i=1}^{|\mathbf{K}|} \rho_i v_i(s_i, f_i, p_i) \\ &\text{s.t. } C1: \ s_i = \{0, 1\} \ \forall i \in \mathbf{K} \\ &\quad C2: \ 0 < p_i \le p_0 \ \forall i \in \mathbf{S} \\ &\quad C3: \ f_i > 0 \ \forall i \in \mathbf{S} \\ &\quad C4: \ \sum_{i \in \mathbf{S}} f_i \le f_0 \\ &\quad C5: \ \sum_{i \in \mathbf{K}} s_i \le N \end{aligned} \tag{12}$$

where $\mathbf{s}$, $\mathbf{f}$, and $\mathbf{p}$ are the vectors of offloading decisions s_i, the allocation of computation resources f_i, and the uplink transmission power p_i, respectively. $\mathbf{K}$ denotes the set of all the mobile users, and $\mathbf{S}$ represents the set of offloading users, i.e., $\mathbf{S} = \{i | s_i = 1\}$. Constraint $C1$ states that a task can be either locally executed or offloaded. According to constraint $C2$, uplink transmission power must be positive and must not exceed the maximum transmission power p_0. Constraint $C3$ ensures that all mobile users in $\mathbf{S}$ are assigned computation resources. Constraint $C4$ guarantees that the total resources assigned are less than the maximum instructions per second allowed at the macro computing center, denoted as f_0. Constraint $C5$ states that, at most, $N = B/W$ mobile users are allowed to transmit simultaneously.

V. Decomposition-Based Problem Reduction

Offloading decisions are coupled with the optimization of communication and computation resources in problem $\mathbf{P}$. Moreover, since offloading decision $\mathbf{s}$ is an integer vector and the resource allocations $\mathbf{f}, \mathbf{p}$ are continuous vectors, problem $\mathbf{P}$ is formulated as mixed-integer nonlinear programming (MINLP) [25]. To be able to solve $\mathbf{P}$, we decompose it into two dependent subproblems [26]:

1) joint optimization of communication and computation resources, for a particular offloading decision;
2) optimization of the offloading decision based on the results of resource optimization.

Accordingly, we can rewrite the MINLP problem $\mathbf{P}$ as

$$\mathbf{P}: \begin{aligned} &\max_{\mathbf{s}} \max_{\mathbf{f},\mathbf{p}} \sum_{i \in \mathbf{S}} \rho_i v_i(s_i, f_i, p_i) \\ &\text{s.t. } C1, C2, C3, C4, \text{ and } C5. \end{aligned} \tag{13}$$

Note that the constraints on the offloading decision and resource allocations are decoupled from each other. Therefore, problem $\mathbf{P}$ can be totally decomposed into two subproblems and can be rewritten as

$$\begin{aligned} &\max_{\mathbf{S}} v(\mathbf{S}) \\ &\text{s.t. } |\mathbf{S}| \le N \end{aligned} \tag{14}$$

where $v(\mathbf{S})$ is the maximum system utility for offloading decision $\mathbf{S}$ when communication and computation resources are jointly optimized and can be denoted as

$$\mathbf{P1}: \begin{aligned} v(\mathbf{S}) = &\max_{\mathbf{f},\mathbf{p}} \sum_{i \in \mathbf{S}} \rho_i v_i(s_i, f_i, p_i) \\ &\text{s.t. } C2, C3, \text{ and } C4. \end{aligned} \tag{15}$$

In the rest of the section, we first solve the subproblem $\mathbf{P1}$ of resource optimization and then use its results to reduce $\mathbf{P}$ to a submodular maximization problem.

A. Joint Optimization of Resources

Substituting (11) into (15), we can rewrite the objective of $\mathbf{P1}$ as

$$\max_{\mathbf{f},\mathbf{p}} \sum_{i \in \mathbf{S}} \rho_i \left(\beta_i^T + \beta_i^E \right) - \sum_{i \in \mathbf{S}} \rho_i \left(\beta_i^T T_i^r / T_i^l + \beta_i^E E_i^r / E_i^l \right). \tag{16}$$

Since $\sum_{i\in\mathbf{S}}\rho_i(\beta_i^T+\beta_i^E)$ is constant for a particular offloading decision, $\mathbf{P1}$ can be transformed into minimizing the total overhead of offloaded mobile users, i.e., $\min_{\mathbf{f},\mathbf{p}}\sum_{i\in\mathbf{S}}\rho_i(\beta_i^T T_i^r/T_i^l+\beta_i^E E_i^r/E_i^l)$. According to (1)–(11), we can obtain the equivalent optimization problem as

$$\mathbf{P2}: \quad \min_{\mathbf{f},\mathbf{p}} \sum_{i\in S} \frac{(\eta_i+\gamma_i p_i)}{\log_2(1+a_i p_i)} + \frac{\tau_i F_i^l}{f_i} \quad \text{s.t. } C2, C3, \text{ and } C4 \tag{17}$$

where $\eta_i=\rho_i\beta_i^T D_i/W\cdot T_i^l$, $\gamma_i=\rho_i\beta_i^E D_i/WE_i^l\zeta$, and $\tau_i=\rho_i\beta_i^T$ for simplicity.

Note that in $\mathbf{P2}$, the allocation of the uplink transmission power $\mathbf{p}$ and computation resources $\mathbf{f}$ are decoupled from each other in both the objective and the constraints. Thus, $\mathbf{P2}$ can be solved by optimizing communication and computation resources independently.

1) Optimization of Uplink Transmission Power: Each mobile user assigns its transmission power solving the following:

$$\mathbf{P3}: \quad \min_{p_i} f(p_i) \quad \text{s.t. } 0<p_i\le p_0 \tag{18}$$

where

$$f(p_i)=\frac{\eta_i+\gamma_i p_i}{\log_2(1+a_i p_i)}. \tag{19}$$

Note that the second-order derivative of (19), i.e., $f''(p_i)$ in (20), shown at the bottom of the page, is not always positive in the domain of $f(p_i)$, and hence, $f(p_i)$ is not convex in the domain.

Lemma 1: $f(p_i)$ is quasiconvex in the domain.

Proof: $f(p_i)$ is twice differentiable on $\mathbf{R}$. We next check the quasiconvex second-order condition that the point x_0 satisfying $f'(x_0)=0$ also satisfies $f''(x_0)\ge 0$ [27].

The first-order derivative of (19) is

$$f'(p_i)=\frac{\gamma_i\log_2(1+a_i p_i)-a_i/\ln 2\frac{\eta_i+\gamma_i p_i}{1+a_i p_i}}{\log_2^2(1+a_i p_i)} \tag{21}$$

$f'(x_0)=0$, when

$$\phi(x_0)=\gamma_i\log_2(1+a_i x_0)-a_i/\ln 2\cdot\frac{\eta_i+\gamma_i x_0}{1+a_i x_0}=0. \tag{22}$$

Substituting x_0 into (20), we obtain

$$f''(x_0)=\frac{a_i^3}{\gamma_i\ln^2 2}\frac{(\eta_i+\gamma_i x_0)^2}{(1+a_i x_0)^3\log_2^3(1+a_i x_0)}\ge 0.$$

Thus, $f(p_i)$ is quasiconvex in the domain. □

A general approach to the quasiconvex optimization problem is the bisection method, solving a convex feasibility problem each time [27]. However, solving convex feasibility problems by an interior cutting-plane method requires $O(m^2/\varepsilon^2)$ iterations, where m is the dimension of the problem [28]. Note that the first-order derivative of (22) is

$$\phi'(p_i)=\frac{a_i^2}{\ln 2}\frac{\eta_i+\gamma_i p_i}{(1+a_i p_i)^2}>0$$

where

$$\phi(0)=-\frac{a_i\eta_i}{\ln 2}<0.$$

This implies that $\phi(p_i)$ is a monotonically increasing transcendental function and negative at the starting point $p_i=0$. Therefore, we develop a low-complexity bisection method, calculating $\phi(p_i)$ instead of solving a convex feasibility problem each time, to obtain the optimal power allocation p_i^* as shown in Algorithm 1. If $\phi(p_0)>0$, $\lceil\log_2(p_0/\varepsilon)\rceil$ iterations are required before Algorithm 1 terminates [27]. Note that in $\mathbf{P3}$, each mobile user can precalculate the optimal transmission power p_i^* before knowing the offloading decision.

Algorithm 1 Uplink Transmission Power Bisection

```
Calculate φ(p_0) = γ_i log(1 + a_i p_0) − (a_i/ln 2) · (η_i + γ_i p_0)/(1 + a_i p_0)
if φ(p_0) ≤ 0 then
    p_i^* = p_0
else
    Initialize p_s = 0 and p_t = p_0
    repeat
        p_l = (p_t + p_s)/2
        if φ(p_l) ≤ 0 then
            p_s = p_l
        else
            p_t = p_l
        end if
    until (p_t − p_s) ≤ ε
    p_i^* = (p_t + p_s)/2
end if
```

2) Optimization of Computation Resources: We formulate the problem of optimizing computation resources as

$$\mathbf{P4}: \quad \min_{\mathbf{f}} g(\mathbf{f}) \quad \text{s.t. } C3 \text{ and } C4 \tag{23}$$

where

$$g(\mathbf{f})=\sum_{i\in\mathbf{S}}\frac{\tau_i F_i^l}{f_i}. \tag{24}$$

Note that the domain of $g(\mathbf{f})$ is convex. The Hessian matrix of (24) is composed of elements either $\partial^2 g/\partial f_i^2=(2\tau_i F_i^l/f_i^3)>0$

$$f''(p_i)=\frac{a_i\{[a_i(\eta_i+\gamma_i p_i)-2\gamma_i(1+a_i p_i)]\log_2(1+a_i p_i)+2a_i(\eta_i+\gamma_i p_i)/\ln 2\}}{\ln 2(1+a_i p_i)^2\log_2^3(1+a_i p_i)} \tag{20}$$

or $\partial g^2/\partial f_i \partial f_j = 0$ $(i \neq j)$. Hence, the Hessian matrix is positive definite, and therefore, $g(\mathbf{f})$ is convex [27].

The Lagrangian of $\mathbf{P4}$ is

$$L(\mathbf{f}, \lambda) = \sum_{i \in \mathbf{S}} \frac{\tau_i F_i^l}{f_i} + \lambda \left(\sum_{i \in \mathbf{S}} f_i - f_0 \right). \tag{25}$$

Note that constraint $C3$ is slack based on Karush–Kuhn–Tucker conditions, which has already been eliminated in (25) [27], [29]. Thus, we can obtain the dual problem of $\mathbf{P4}$ as

$$\max_{\lambda > 0} \min_{\mathbf{f} \succ 0} L(\mathbf{f}, \lambda) = \sum_{i \in \mathbf{S}} \frac{\tau_i F_i^l}{f_i} + \lambda \left(\sum_{i \in \mathbf{S}} f_i - f_0 \right). \tag{26}$$

Note that f_i^* satisfies $\partial L/\partial f_i|_{f_i^*} = 0$ and can be obtained as

$$f_i^* = \frac{\sqrt{\tau_i F_i^l}}{\sqrt{\lambda}}. \tag{27}$$

Substituting (27) into (26), we can obtain the Lagrangian dual function of $\mathbf{P4}$ as

$$\varphi(\lambda) = \min_{\mathbf{f} \succ 0} L(\mathbf{f}, \lambda) = 2\sqrt{\lambda} \sum_{i \in \mathbf{S}} \sqrt{\tau_i F_i^l} - \lambda f_0. \tag{28}$$

By solving $\varphi'(\lambda^*) = 0$, we obtain the optimal Lagrangian multiplier λ^* as

$$\lambda^* = \left(\sum_{i \in \mathbf{S}} \sqrt{\tau_i F_i^l} \Big/ f_0 \right)^2. \tag{29}$$

Substituting (29) into (27), we can obtain the optimal allocation of computation resources as

$$f_i^* = \frac{\sqrt{\tau_i F_i^l}}{\sum\limits_{i \in \mathbf{S}} \sqrt{\tau_i F_i^l}} f_0. \tag{30}$$

Substituting (30) into (24), the minimum value of $g(\mathbf{f})$ is

$$g(\mathbf{f})^{\min} = \frac{\left(\sum\limits_{i \in \mathbf{S}} \sqrt{\tau_i F_i^l} \right)^2}{f_0}. \tag{31}$$

B. Submodular Maximization Problem

For a particular offloading decision, the computation and communication resources are optimized as described in Section V-A. Specifically, according to (15)–(31), the system utility, i.e., $v(\mathbf{S})$, can be obtained as

$$v(\mathbf{S}) = \sum_{i \in \mathbf{S}} \rho_i \left(\beta_i^T + \beta_i^E \right) - \sum_{i \in \mathbf{S}} f\left(p_i^*\right) - g(\mathbf{f})^{\min} \tag{32}$$

where p_i^* can be precalculated through Algorithm 1, and $g(\mathbf{f})^{\min}$ can be obtained by the closed-form expression (31). Substituting (32) into (14), $\mathbf{P}$ is reduced to an integer nonlinear programming problem, i.e.,

$$\mathbf{P5}: \quad \begin{aligned} &\max_{\mathbf{S}} \sum_{i \in \mathbf{S}} \rho_i \left(\beta_i^T + \beta_i^E \right) - \sum_{i \in \mathbf{S}} f\left(p_i^*\right) - g(\mathbf{f})^{\min} \\ &\text{s.t.} \quad |\mathbf{S}| \leq N. \end{aligned} \tag{33}$$

Next, we focus on solving the reduction problem $\mathbf{P5}$ instead of the original MINLP problem $\mathbf{P}$.[3]

Lemma 2: The system utility $v(\mathbf{S})$ is submodular.

Proof: The marginal value $\Delta_i v(\mathbf{A})$ denotes the increasing system utility by adding element i into $\mathbf{A}$, i.e.,

$$\begin{aligned} \Delta_i v(\mathbf{A}) &= v\left(\mathbf{A} \cup \{i\}\right) - v(\mathbf{A}) \\ &= \rho_i \left(\beta_i^T + \beta_i^E \right) - \left(\Delta(i) + \Delta(i|\mathbf{A}) \right) \end{aligned} \tag{34}$$

where $\Delta(i)$ and $\Delta(i|\mathbf{A})$ represent the constant and variable parts of the marginal value in (34). These are

$$\Delta(i) = \frac{\eta_i + \gamma_i p_i^*}{\log_2\left(1 + a_i p_i^*\right)} + \frac{\tau_i F_i^l}{f_0} \tag{35}$$

$$\Delta(i|\mathbf{A}) = \frac{2\sqrt{\tau_i F_i^l} \sum\limits_{j \in \mathbf{A}} \sqrt{\tau_j F_j^l}}{f_0}. \tag{36}$$

While $\Delta(i)$ does not depend on the offloading decision set, $\Delta(i|\mathbf{A})$ monotonically increases with the offloading decision set $\mathbf{A}$. Hence, according to (34), $\Delta_i v(\mathbf{A})$ is a monotonically decreasing function of the current offloading decision $\mathbf{A}$, i.e., for all $\mathbf{A} \subset \mathbf{B}$ and $i \notin \mathbf{B}$, $\Delta_i v(\mathbf{A}) \geq \Delta_i v(\mathbf{B})$ is satisfied. Thus, $v(\mathbf{S})$ is submodular [30]. □

Note that the marginal value $\Delta_i v(\mathbf{A})$ in (34) can be negative, and therefore, $v(\mathbf{S})$ is nonmonotonic, i.e., $\mathbf{A} \supset \mathbf{B}$ cannot imply $v(\mathbf{A}) \geq v(\mathbf{B})$. Moreover, $v(\mathbf{S})$ is not symmetric, i.e., $v(\mathbf{A}) \neq v(\mathbf{A}^c)$. Thus, $v(\mathbf{S})$ is an asymmetric nonmonotonic submodular function. Problem $\mathbf{P5}$ is an asymmetric nonmonotonic submodular function maximization problem with a cardinality constraint and is NP-hard according to [30] and [31].

Theorem 1: Problem $\mathbf{P}$ is NP-hard.

Proof: See Appendix A. □

VI. Heuristic Offloading Decision Algorithm

The optimal offloading decision of $\mathbf{P5}$ can be computed by enumerating and comparing all possible offloading decisions. However, the complexity of such enumeration is $O(2^K)$. Note that problem $\mathbf{P}$ is NP-hard. Therefore, in the rest of this section, we propose the HODA to obtain the local optimum in polynomial complexity.

A. Locally Optimal Offloading Decision

Definition 1: Offloading decision $\mathbf{S}$ is locally optimal, if and only if neither supersets nor subsets of $\mathbf{S}$ result in higher system utility than $\mathbf{S}$, i.e., for all $\mathbf{A} \subsetneq \mathbf{S}$ or $\mathbf{A} \supsetneq \mathbf{S}$ and $|\mathbf{A}| \leq N$, $v(\mathbf{S}) \geq v(\mathbf{A})$.

Note that determining whether an offloading decision is locally optimal needs comparing all the supersets or subsets with the original decision, which, again, has high complexity of $O(2^K)$. Next, we discuss how the offloading decision expands or contracts to increase system utility and exploit the submodularity of system utility to reduce the complexity of determining the locally optimal offloading decision to $O(K)$.

[3] Appendix A demonstrates the equivalence of $\mathbf{P}$ and $\mathbf{P5}$ in terms of the same answers to corresponding problems in decision forms.

Definition 2: Offloading decision $\mathbf{A}$ is better than offloading decision $\mathbf{B}$, which is denoted as $\mathbf{A} \triangleright \mathbf{B}$, if and only if the utility of $\mathbf{A}$ is higher than that of $\mathbf{B}$, i.e., $v(\mathbf{A}) \geq v(\mathbf{B})$. The relation $\triangleright$ is transitive, i.e., $\mathbf{A} \triangleright \mathbf{B}$ and $\mathbf{B} \triangleright \mathbf{C}$ imply $\mathbf{A} \triangleright \mathbf{C}$.

Based on Definition 2, the offloading decision set can expand or contract while increasing the system utility. Specifically, if $\mathbf{B} \supsetneq \mathbf{A}$, $|\mathbf{B}| \leq N$, and $\mathbf{B} \triangleright \mathbf{A}$, offloading decision $\mathbf{A}$ can expand to $\mathbf{B}$. Similarly, if $\mathbf{C} \subsetneq \mathbf{A}$ and $\mathbf{C} \triangleright \mathbf{A}$, offloading decision $\mathbf{A}$ can contract to $\mathbf{C}$.

Definition 3: An offloading decision $\mathbf{A}$ is inextensible, if and only if $|\mathbf{A}| = N$ or there exists no set $\mathbf{B} \supsetneq \mathbf{A}$ satisfying $\mathbf{B} \triangleright \mathbf{A}$. An offloading decision $\mathbf{A}$ is indecomposable, if and only if there exists no subset $\mathbf{C} \subsetneq \mathbf{A}$ satisfying $\mathbf{C} \triangleright \mathbf{A}$.

If an offloading decision is both inextensible and indecomposable, it is also locally optimal. To find a locally optimal offloading decision, we can repeat extending the original offloading decision until it cannot be extended, while ensuring that the decision is indecomposable at the same time. However, again, determining whether an offloading decision is inextensible or indecomposable also has a complexity level of $O(2^K)$. In the following, we prove two lemmas showing that we can reduce this complexity to $O(K)$, when we take into account the submodularity of the utility function.

Lemma 3: Offloading decision $\mathbf{A}$, $|\mathbf{A}| < N$ is inextensible, if and only if there exists no element $i \in \mathbf{A}^c$ satisfying $(\mathbf{A} \cup \{i\}) \triangleright \mathbf{A}$.

Proof: First, let us assume that $\mathbf{A}$ is inextensible. If there exists an element $i \in \mathbf{A}^c$ satisfying $(\mathbf{A} \cup \{i\}) \triangleright \mathbf{A}$, $\mathbf{A}$ can expand to set $\mathbf{A} \cup \{i\}$ at least, where a contradiction with $\mathbf{A}$ is inextensible.

Second, let us assume that there does not exist any element $i \in \mathbf{A}^c$ satisfying $(\mathbf{A} \cup \{i\}) \triangleright \mathbf{A}$, i.e., $\Delta_i v(\mathbf{A}) \leq 0$. For any set $\mathbf{I} \supsetneq \mathbf{A}$, let $\mathbf{A} = \mathbf{T}_1 \subset \mathbf{T}_2 \subset \cdots \subset \mathbf{T}_k = \mathbf{I}$ be a chain of sets, where $\mathbf{T}_i \setminus \mathbf{T}_{i-1} = \{e_i\}$. For $2 \leq i \leq k$, due to submodularity, $v(\mathbf{T}_i) - v(\mathbf{T}_{i-1}) = \Delta_{e_i} v(\mathbf{T}_{i-1}) \leq \Delta_{e_i} v(\mathbf{A}) \leq 0$. Summing up all these inequalities, we obtain that $v(\mathbf{I}) - v(\mathbf{A}) \leq 0$, i.e., for all $\mathbf{I} \supsetneq \mathbf{A}$, $\mathbf{A} \triangleright \mathbf{I}$, and therefore, $\mathbf{A}$ is inextensible. □

Lemma 4: Offloading decision $\mathbf{A}$ is indecomposable if, and only if, there exists no element $i \in \mathbf{A}$ satisfying $(\mathbf{A} \setminus \{i\}) \triangleright \mathbf{A}$.

Proof: This can be proven similar to Lemma 3 and, therefore, is omitted for the sake of brevity. □

Lemmas 3 and 4 show that instead of evaluating all the supersets or subsets of the original decision, we can evaluate only adding elements from $\mathbf{A}^c$ or removing elements from $\mathbf{A}$. This has a complexity level of $O(K)$. Specifically, if all $i \in \mathbf{A}^c$ satisfy $\mathbf{A} \triangleright (\mathbf{A} \cup \{i\})$ and all $i \in \mathbf{A}$ satisfy $\mathbf{A} \triangleright (\mathbf{A} \setminus \{i\})$, offloading decision $\mathbf{A}$ is locally optimal.

B. User Classification

Here, we discuss how to build the initial offloading set. Mobile users vary in terms of the capabilities of their mobile devices and the current network conditions they experience. These differences may make offloading more beneficial for some users and local execution for others. For example, a mobile user with a better uplink channel condition and lower computation capability will benefit more from offloading.

Next, we define two conditions to determine the users for local execution and offloading, respectively.

Condition 1: If $\Delta_i v(\mathbf{A})^{\max} = \rho_i(\beta_i^T + \beta_i^E) - \Delta(i) \leq 0$, mobile user i executes tasks locally.

As mentioned in the proof of Lemma 3, $\Delta_i v(\mathbf{A}) > 0$ indicates that $\mathbf{A}$ can expand to $\mathbf{A} \cup \{i\}$. Note that both (35) and (36) are positive, and $\Delta(i|\mathbf{A}) = 0$ if and only if $\mathbf{A} = \emptyset$. Therefore, $\Delta_i v(\mathbf{A})$ in (34), i.e., the marginal value when i is added, cannot be positive when Condition 1 is satisfied. Therefore, users that satisfy this condition choose to execute tasks locally. $\mathbf{S}_{\text{local}} = \{i | \Delta_i v(\mathbf{A})^{\max} \leq 0\}$ denotes the set of mobile users that satisfy Condition 1.

Next, the macrocell checks the following condition to decide whether mobile user i should be offloaded. After determining $\mathbf{S}_{\text{local}}$, $\Delta_i v(\mathbf{A})$ gets the minimum value, when $\mathbf{A}$ is the maximum set that does not include i, i.e., $\mathbf{A}_{\max} = \mathbf{K} \setminus (\mathbf{S}_{\text{local}} \cup \{i\})$.

Condition 2: If $\Delta_i v(\mathbf{A})^{\min} = \rho_i(\beta_i^T + \beta_i^E) - (\Delta(i) + \Delta(i|\mathbf{A}_{\max})) \geq 0$, the macrocell selects mobile user i for offloading.

Note that if this condition is satisfied, adding mobile user i increases utility, even if $\mathbf{A}$ reached its largest possible size. $\mathbf{S}_{\text{remote}} = \{i | \Delta_i v(\mathbf{A})^{\min} \geq 0\}$ is the set of mobile users that satisfy Condition 2 and is indecomposable, according to Definition 3.

Based on this classification, $\mathbf{S}_{\text{remote}}$ should be included in the locally optimal offloading decision set, and $\mathbf{S}_{\text{local}}$ must be excluded from this set. Finally, $\mathbf{S}_{\text{search}} = \mathbf{K} \setminus (\mathbf{S}_{\text{local}} \cup \mathbf{S}_{\text{remote}})$ constitutes the remaining mobile users, which cannot be predetermined to run local or remote executions.

C. Heuristic Offloading Decision Algorithm

Based on the discussions and results in Section VI-A and B, we adopt a semidistributed offloading approach composed of two stages, shown in Algorithm 2.

Algorithm 2 Heuristic Offloading Decision Algorithm (HODA)

Stage I: at each mobile user i
1: i computes p_i^* (Algorithm 1)
2: **if** $i \notin \mathbf{S}_{\text{local}}$ $(\Delta_i v(\mathbf{A})^{\max} > 0)$ **then**
3: send an offloading request
4: **else**
5: send NULL
6: **end if**
Stage II: at the macro BS
7: Wait until all the requests are received
8: Determine $\mathbf{S}_{\text{remote}}$ and $\mathbf{S}_{\text{search}}$ (Condition 2)
9: $\mathbf{S} \leftarrow \mathbf{S}_{\text{remote}}$
10: **if** $|\mathbf{S}| \geq N$ **then**
11: **while** $|\mathbf{S}| > N$ **do**
12: $i \leftarrow \arg\min_{i \in \mathbf{S}} \{v_i(s_i, f_i, p_i)\}$
13: $\mathbf{S} \leftarrow \mathbf{S} \setminus \{i\}$
14: **end while**
15: **else**
16: **repeat**
17: $i \leftarrow \arg\max_{i \in \mathbf{S}_{\text{search}}} \{v_i(s_i, f_i, p_i)\}$ s.t. $\Delta_i v(\mathbf{S}) > 0$ **and** $(\mathbf{S} \cup \{i\})$ is indecomposable

18: $\mathbf{S}_{\text{search}} \leftarrow \mathbf{S}_{\text{search}} \setminus \{i\}$
19: $\mathbf{S} \leftarrow \mathbf{S} \cup \{i\}$
20: **until** $\mathbf{S}$ is inextensible
21: **end if**
22: Send offloading decision $\mathbf{S}$ to mobile users

1) Algorithmic Process: In Stage I, mobile users run Algorithm 1 to compute optimal uplink transmission power p_i^* based on their current conditions and device characteristics (e.g., device computation capacity and power consumption, wireless channel conditions, user preferences, and computation tasks). Moreover, the users check whether they satisfy Condition 1, deciding on the set $\mathbf{S}_{\text{local}}$. The mobile users that are not in $\mathbf{S}_{\text{local}}$ send their offloading requests to the macro BS, including information on their current conditions and characteristics. The rest of the users send NULL messages to indicate that they will not be offloading.

In Stage II, the macro BS waits until it collects all the requests. Based on Condition 2, mobile users are classified into $\mathbf{S}_{\text{remote}}$ and $\mathbf{S}_{\text{search}}$. The offloading decision set is initialized to $\mathbf{S} = \mathbf{S}_{\text{remote}}$. If the initialized offloading users require more than the available bandwidth, i.e., $|\mathbf{S}| > N$, the macro BS repeats deleting the user of minimum utility in $\mathbf{S}$ until $|\mathbf{S}| = N$. Otherwise, the macro BS heuristically selects the best mobile user in $\mathbf{S}_{\text{search}}$, which has the maximum utility, i.e., $v_i(s_i, f_i, p_i)$, while ensuring $\Delta_i v(\mathbf{S}) > 0$ and $\mathbf{S} \cup \{i\}$ is indecomposable. The macro BS repeats adding the best user in $\mathbf{S}_{\text{search}}$ into $\mathbf{S}$ until $\mathbf{S}$ is inextensible. Then, decision $\mathbf{S}$ is sent back to the mobile users, and on receiving this information, the mobile users can offload their tasks accordingly.

Theorem 2: HODA obtains the local optimum of problem $\boldsymbol{P}$.

Proof: As in Sections V and VI-A, the offloading solution is locally optimal in $\mathbf{P}$, if and only if it is both inextensible and indecomposable. HODA initializes the offloading set $\mathbf{S} = \mathbf{S}_{\text{remote}}$, which is indecomposable. Next, the offloading set contracts to the maximum size due to the bandwidth constraint or expands to the inextensible set while ensuring it is indecomposable. The result of HODA is both inextensible and indecomposable and, therefore, is locally optimal of $\mathbf{P}$. □

2) Complexity Analysis: To reduce complexity, the proposed approach migrates the computation of p_i^* and $\mathbf{S}_{\text{local}}$ to the mobile users. At the macrocell, the size of searching users, i.e., $\mathbf{S}_{\text{search}}$, is further reduced by building the initial offloading set, i.e., $\mathbf{S}_{\text{remote}}$, through Condition 2. Moreover, Lemmas 3 and 4 reduce the complexity of determining inextensible and indecomposable sets to $O(K)$.

Specifically, at the mobile side, the complexity of Stage I is determined by Algorithm 1. Here, the complexity of the bisection method is $O(\log(p_0/\varepsilon))$. At the macrocell, the complexity of Stage II is determined by the complexity of building set $\mathbf{S}$, which takes $O(K)$ iterations. If $|\mathbf{S}| \geq N$, selecting the user with maximum utility in $\mathbf{S}$ requires the complexity of $O(K)$. Otherwise, within each iteration, by storing the result of $\sum_{i\in\mathbf{S}} \sqrt{\beta_i^T F_i^l}$, finding the maximum utility user with $\Delta_i v(\mathbf{S}) > 0$ while checking that $\mathbf{S} \cup \{i\}$ remains indecomposable has a complexity level of $O(K^2)$. Hence, the complexity of Stage II is $O(K^3)$.

3) Discussion: As described in Section VI-C1, HODA adopts a semidistributed request-and-decision framework: 1) The mobile users send offloading requests independently in Stage I, and 2) the macro computing center decides the offloaded users and preallocates the computation and communication resources for offloading in Stage II. Such framework can easily adapt to other scenarios.

1) **Resource multiplexing**: In the general case when computation tasks arrive dynamically, we can execute HODA as an online algorithm to achieve resource multiplexing in the time dimension. The mobile users can apply personal offloading policies to determine whether to send offloading requests, while the macro computing center always collects the requests and then allows resource sharing based on the current available resources. Therefore, the released communication and computation resources could be reused by the upcoming computation tasks.
2) **Multiple computing centers**: The availability of the computing center, i.e., f_0, varies over time due to the stochastic nature of users' offloading requests. Thus, the single macro computing center can become a bottleneck. To address this problem, the resource providers can either increase the capability of the macro computing center or deploy multiple computing centers. In the case of multiple computing centers, the mobile users can apply the offloading policies that make offloading decisions among multiple devices independently (e.g., [7]) and send offloading requests to the selected computing center.
3) **Fairness**: The macro computing center cannot accommodate all the offloading requests from all users, and the users with bad wireless channels may seldom be allowed for offloading. To achieve fairness among users, the resource provider can apply fair scheduling (e.g., proportional fairness) by adjusting its preferences. For instance, the resource provider can increase ρ_i to prioritize user i for offloading.

D. Offloading in Ultra-dense Network (UDN)

In this paper, we have considered multiuser offloading in macro LTE networks. Recently, the ultra-dense network (UDN) is envisioned as a key technique to address the coverage and capacity problem in next-generation mobile networks. However, the UDN faces critical intercell interference due to the unsystematic and ultradense deployment of small cells, which significantly limits system capacity [32]. Therefore, offloading in the UDN should take interference into account to further improve user experience.

We consider the scenario where each mobile user i is served by its corresponding small cell i and all the small cells share the same bandwidth. Different from (4), the achievable data rate of user i becomes

$$R_i(\mathbf{p}) = W \log_2 \left(1 + \frac{p_i h_i^i}{N_0 + \sum_{j\in\mathbf{K}, j\neq i} p_k h_j^i}\right) \tag{37}$$

where h_j^i denotes the channel gain from mobile user j to small cell i. Due to the interferences among users and the contiguity

constraint of the SC-FDMA technique, even the problem of optimizing transmission power is nonconvex [33].

In the literature, the authors apply either machine learning [23] or dynamic programming [33] to mitigate the interference. However, the problem of jointly optimizing offloading and resource allocation as in this paper can be even more challenging, since the data rate of a user also depends on the transmission power of others. We will leave this problem for future work.

VII. Evaluation

We evaluate the efficiency of our HODA based on Monte Carlo simulations. Since no work jointly optimizes the offloading decision together with communication and computation resources, we compare HODA against the following.

1) **Local execution**: There is no offloading. All tasks are always locally executed.
2) **Enumeration algorithm**: All 2^K offloading decisions are enumerated and compared to find the global optimum.
3) **All remote joint optimization algorithm (ARJOA)**: All tasks are offloaded for remote computing. Then, we add joint optimization of communication and computation resources proposed in Section V-A, as in [15] and [16].
4) **Independent offloading and joint optimization algorithm (IOJOA)**: The mobile users make offloading decisions independently, as in [4]–[8]. Moreover, the communication and computation resources are jointly optimized, as in Section V-A.

Note that when all the system bandwidth is already occupied, mobile users in ARJOA and IOJOA choose to execute their computation tasks locally to avoid the unexpected latency from both the communication and computation sides.

We simulate 5000 runs in a network composed of a single macrocell with a radius of 500 m, where the mobile users are randomly distributed. We set the system bandwidth $B = 20$ MHz and user equipment (UE) bandwidth $W = 1$ MHz, and therefore, $N = 20$. The radio communication parameters follow the Third-Generation Partnership Project specification [34]. As an example of a complex application, we adopt the face recognition application in [1], [11], and [12], where the input data size $D = 420$ kB and the required number of CPU cycles $C = 1000$ MCycles. The communication and computation parameters used in our simulations are summarized in Table I.

TABLE I
Simulation Parameters

Parameters	Assumptions
Macrocell Radius	500m
System Bandwidth B	20MHz
UE Bandwidth W	1MHz
Pathloss from Mobile User to Macro BS	$128.1 + 37.5\log_{10}(r)$
Thermal Noise Density	-174dBm/Hz
Max UE Tx Power p_0	23dBm
Lognormal Shadowing Standard Deviation	10dB
Input Data Size D	420kB
Total Number of CPU Cycles C	1000MCycles
Local Computation Capability f^l	0.5GHz−1.5GHz
Mobile Users' Preferences β_i^T, β_i^E	$0.25 - 0.75$
Resource Providers' Preferences ρ_i	1
Maximum Remote Computation Resources f_0	20GHz

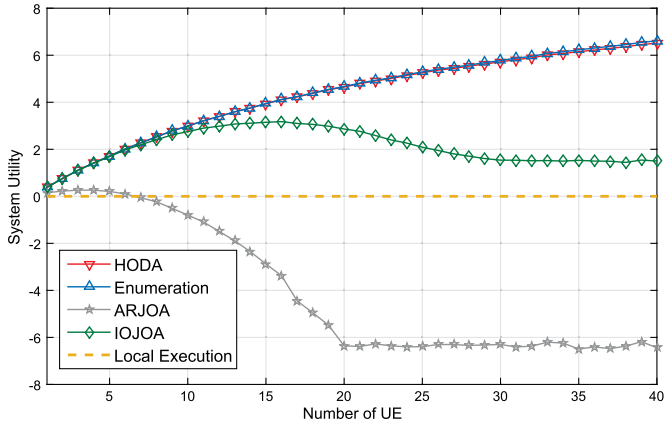


Fig. 2. Comparison of system utility as K varies from 1 to 40. HODA performs very close to the optimal solution.

A. Analysis of System Utility

1) System Utility With Varying Number of Users: We first evaluate the system utility as the number of users, i.e., K, increases from 1 to 40. Fig. 2 shows a comparison of the system utility achieved by HODA, local execution, enumeration, IOJOA, and ARJOA. In the case of local execution, the system utility is always 0, and therefore, its performance does not change with increasing K. HODA performs very close to the optimal solution computed by the enumeration algorithm. Its performance slightly degrades when K gets larger and remains within 5% of the optimal. Furthermore, as the number of mobile users increases, both the enumeration algorithm and the HODA have a wider choice of mobile users with varying diversity, so that they are able to continuously improve the system utility. However, the performance of IOJOA and ARJOA starts degrading as K gets larger since all users compete for the limited communication and computation resources. With approximately seven users, ARJOA's performance drops below local execution. IOJOA performs similarly to HODA when $K < 10$, but loses more than 65% system utility when $K > 30$. Since the limited communication resources prevent from offloading more tasks, both IOJOA and ARJOA reach the stable system utility as K gets larger. While IOJOA performs better than ARJOA due to selecting better users to offload and, hence, alleviating the competition on resources, it never reaches the performance of HODA or enumeration. This shows that optimizing offloading decisions is essential to improve system utility.

2) Number of Offloaded Users With Varying Number of Users: Fig. 3 shows how many users HODA, IOJOA, ARJOA, and the enumeration algorithm decide to offload as K varies from 1 to 40. As K increases, since ARJOA offloads all the users, the number of offloaded users linearly increases at first. However, due to the constraint of communication resources, ARJOA cannot offload more users when $K > 20$. IOJOA selects fewer better computation tasks to offload but still faces the bottleneck

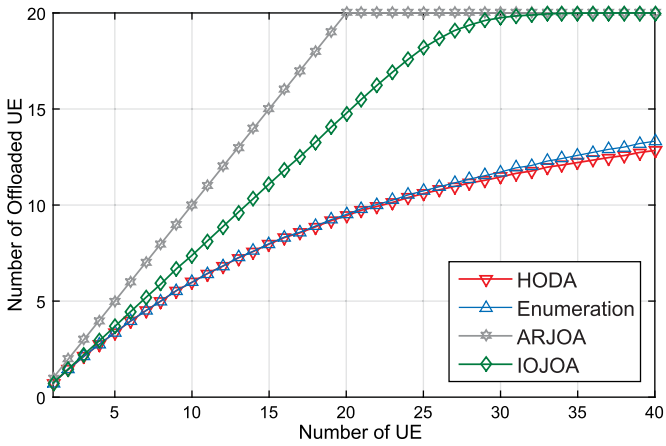


Fig. 3. Average number of offloaded users as K increases from 1 to 40. Both HODA and the enumeration algorithm offload fewer users as K increases, improving system utility.

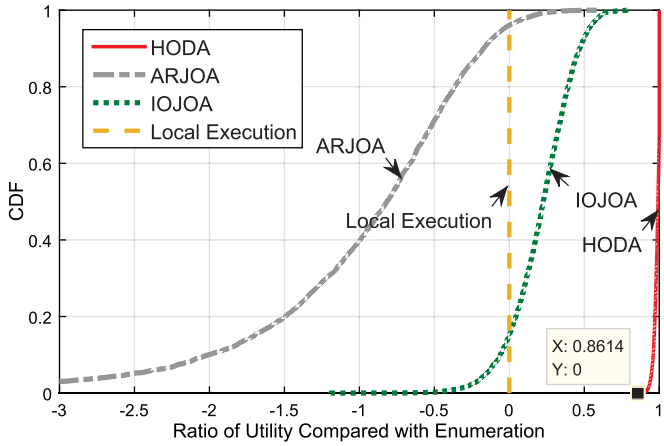


Fig. 4. Comparison of the cdf of the ratio of the system utility of a solution compared with the enumeration algorithm when $K = 40$. HODA does not exhibit high variations.

of communication resources when $K > 30$. In contrast, both the enumeration algorithm and the HODA maintain a similar and lower number of offloaded users, due to the limited computation resources. For HODA as K increases, the size of $\mathbf{S}_{\text{local}}$ linearly increases. This is expected as device characteristics are assigned uniformly random in simulation, and the mobile users that satisfy Condition 1 execute their tasks locally. As K increases, computation resources also become scarcer, and users cannot satisfy Condition 2 easily; hence, the size of $\mathbf{S}_{\text{remote}}$ gets smaller. Hence, $\mathbf{S}_{\text{search}}$ becomes larger, providing a wide selection of users to choose from. Compared with the enumeration algorithm, HODA selects slightly fewer users for offloading, as it prioritizes users with higher utility rather than total system utility.

3) Variation of System Utility: Next, we mainly evaluate the scenario of 40 mobile users, as the significant resource bottlenecks show the benefit of joint decision and offloading optimization. To understand how the performance of the different algorithms varies across different simulation runs, we plot the cumulative distribution function (cdf) of the ratio of the system utility of a given solution compared with the enumeration algorithm when $K = 40$ in Fig. 4. The ratio of system utility is the ratio of the utility of a solution and the optimal utility. Since ARJOA and IOJOA offload more UE devices, they have negative system utility, even worse than local execution. HODA performs significantly better in achieving comparable performance to the optimal. Fig. 4 shows that HODA does not exhibit high variations, where the worst case performs 86% of the optimal.

B. Analysis of Per-User Metrics

1) Distribution of User Utility: Fig. 5 shows the cdf of the user utility achieved by HODA, local execution, enumeration, IOJOA, and ARJOA when $K = 40$. In local execution, all the users have the same utility of 0 according to (11). As in Fig. 3, both IOJOA and ARJOA offload 20 computation tasks when $K = 40$ and, as expected, have 50% of the users with 0 utility. On the other hand, HODA and enumeration have about

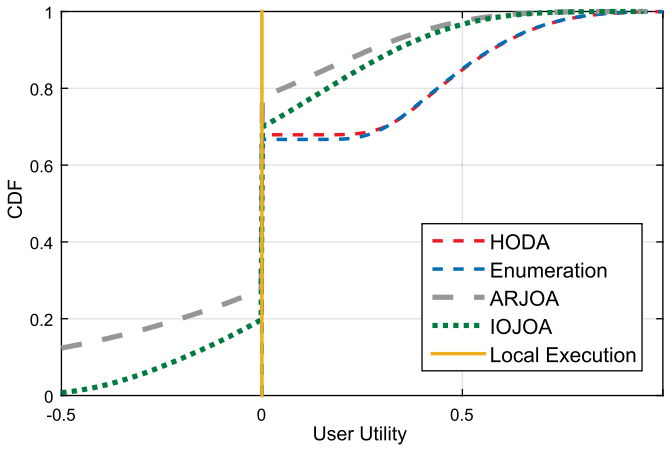


Fig. 5. Comparison of the cdf of user utility when $K = 40$. All users have nonnegative utility with HODA.

68% of its users with 0 utility and, hence, offload about 32% users. Both algorithms perform similarly, although HODA may achieve better system utility for remote users as it prioritizes maximum utility users when building the offloading set. Note that with 40 mobile users, the resources are extremely limited in this scenario, and therefore, in ARJOA, randomly selecting 20 users for offloading leads to 28% of the mobile users achieving a utility lower than local execution. IOJOA also suffers, but less drastically, with 20% users achieving negative utility.

2) Distribution of Task Completion Time: Fig. 6 shows the cdf of task completion time of any user achieved by HODA, local execution, enumeration, IOJOA, and ARJOA when $K = 40$. HODA, local execution, and enumeration algorithm perform closely with each other. For the majority of mobile users, the task completion time of HODA and the enumeration algorithm is only slightly shorter than that of local execution. On the other hand, as IOJOA and ARJOA offload more users, the task computation times are much longer when the resources are limited. In this case, ARJOA again suffers the worst task completion time. Specifically, the task completion time of the

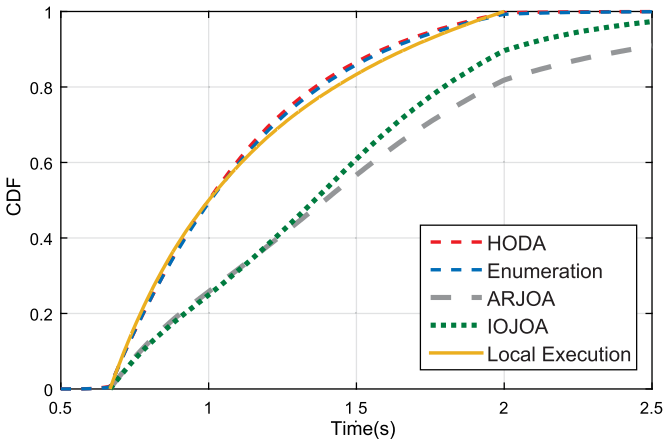


Fig. 6. Comparison of the cdf of task completion time when $K = 40$. Offloading too many users leads to an increase in task completion time.

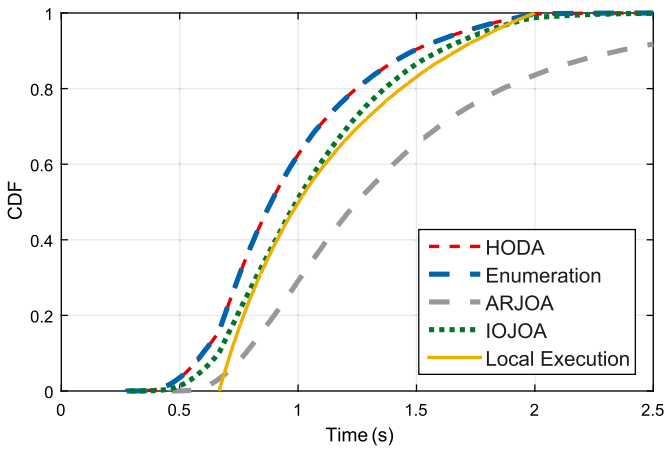


Fig. 7. Comparison of the cdf of task completion time when $K = 10$. HODA improves task completion time.

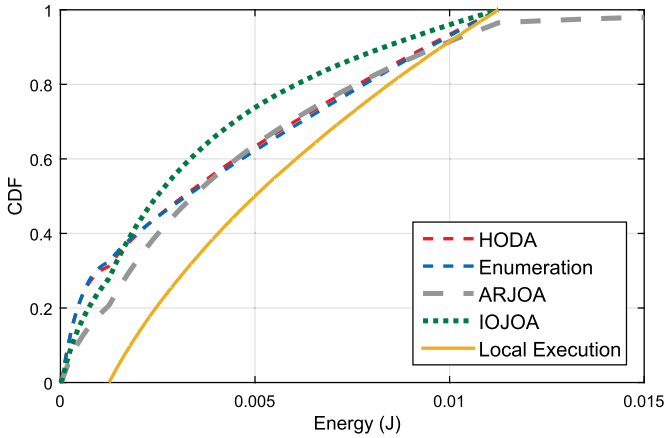


Fig. 8. Comparison of the cdf of energy consumption when $K = 40$. Offloading selective tasks improves energy efficiency.

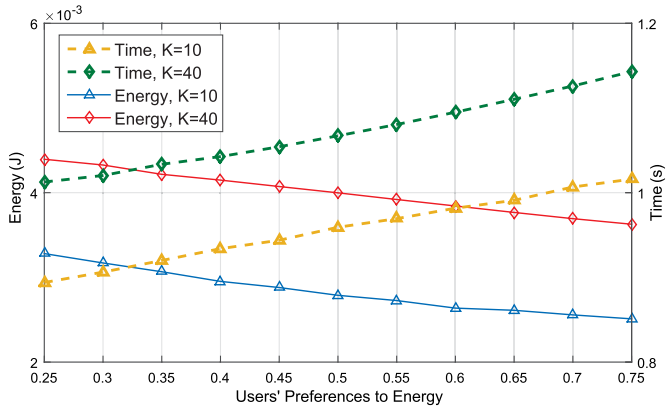


Fig. 9. Average latency and energy consumption of HODA as $\beta_i^E = 0.25$–0.75 with $\beta_i^T = 0.5$ when $K = 10, 40$. Increasing β_i^E reduces the energy consumption at the expense of longer latency.

50% mobile users of HODA is less than 1 s, which is a 30% improvement over ARJOA.

When we decrease the number of users to $K = 10$ as shown in Fig. 7, for a small number of users, both HODA and the enumeration algorithm are able to improve task completion times compared with local execution and IOJOA, which perform similarly. The worst performance is experienced by ARJOA, which offloads all tasks creating a resource bottleneck.

3) Distribution of Energy Consumption: Fig. 8 shows the cdf of energy consumption achieved by HODA, local execution, enumeration, IOJOA, and ARJOA when $K = 40$. Since offloading saves energy in general, local execution has the worst performance with the highest energy consumption. The most energy-efficient algorithm is IOJOA as it offloads each user independently, and for these selected tasks, remote computation has better energy performance than local computation. The energy consumption of HODA is close to the enumeration algorithm. For the 50% mobile users, the energy consumption of HODA is less than 3.5 mJ, saving 30% energy compared with local execution. While IOJOA offloads 50% of its users, compared with the 32% in HODA and enumeration (see Fig. 5), IOJOA fails to select users with better utility to offload, which results in a crossing point among IOJOA, HODA, and enumeration in Fig. 8.

4) Analysis of Users' Preferences: Fig. 9 shows the users' average latency and energy consumption of HODA as the users' preferences to energy consumption, i.e., β_i^E, vary from 0.25 to 0.75 with $\beta_i^T = 0.5$ when $K = 10, 40$. With fixed $\beta_i^T = 0.5$ and increasing β_i^E, mobile users prefer lower energy consumption. Hence, the users' average energy consumption decreases approximately linearly with β_i^E at the cost of suffering longer latency. In the case of $K = 40$, mobile users compete for the limited resources and have less probability to offload their tasks. Therefore, the users experience more average latency and energy consumption than in the case of $K = 10$.

In summary, simulation results show that the limited resources in proximate clouds necessitate the joint optimization of offloading decision and communication and computation resources. HODA is able to approximate well the performance of the optimal enumeration algorithm in terms of system utility, task completion time, and energy consumption. While the enumeration algorithm always aims to maximize the system utility, HODA serves user preferences better, by prioritizing users with maximum utility in its offloading decisions and, hence, achieves better per-user metrics at the expense of fewer offloaded users.

VIII. Conclusion

In this paper, we have argued that, due to the resource bottlenecks in proximate clouds, the joint optimization of communication and computation resources without offloading decision degrades system performance. In particular, as the number of mobile users increases, the system utility and task completion time can be worse than local execution. This is because limited resources at the remote computing unit create a tradeoff between the number of offloaded mobile users and their QoE. In this paper, we took into account user QoE through user utility and total system utility functions. We have proposed a semidistributed HODA, which ensures that mobile users with better utility are offloaded, while others perform local executions. Essentially, the complexity of HODA is $O(K^3)$, in comparison with the $O(2^K)$ complexity of the optimal. Simulation results show that HODA performs very close to the optimal solution.

Appendix A
Proof of NP-Hardness of Problem $\mathbf{P}$

We reduce the submodular maximization problem $\mathbf{P5}$ to problem $\mathbf{P}$. Since the submodular maximization problem $\mathbf{P5}$ is NP-hard [30], [31], problem $\mathbf{P}$ is at least as hard as $\mathbf{P5}$.

The decision form of $\mathbf{P}$ is as follows: Given an instance $I_{\mathbf{P}} = \{D_i, C_i, T_i^l, E_i^l, h_i, \beta_i^T, \beta_i^E, \rho_i, f_0, p_0, N\}$, are there vectors $\mathbf{s}$, $\mathbf{f}$, and $\mathbf{p}$ such that $\sum_{i=1}^{|\mathbf{K}|} \rho_i v_i \geq k$? As described in Section V-A, given an instance of $\mathbf{P}$, the instance of $\mathbf{P5}$ can be formed by obtaining p_i^* through the bisection Algorithm 1. Thus, the decision form of $\mathbf{P5}$ is as follows: Given an instance $I_{\mathbf{P5}} = I_{\mathbf{P}} \cup \{p_i^*\}$, is there a vector $\mathbf{s}$ such that $v(\mathbf{S}) \geq k$?

Next, given an instance of $\mathbf{P5}$, we construct the instance of $\mathbf{P}$ by removing the power allocation p_i^*. In this case, we prove that the answers to $\mathbf{P}$ and $\mathbf{P5}$ are the same in the decision form.

Proof: First, if vector $\mathbf{s}$ satisfies $v(\mathbf{S}) \geq k$ for $\mathbf{P5}$, we can construct the vectors $\mathbf{s}$, $\{\mathbf{f}|f_i = (\sqrt{\tau_i F_i^l} / \sum_{i \in \mathbf{S}} \sqrt{\tau_i F_i^l}) f_0\}$, and $\{\mathbf{p}|p_i = p_i^*\}$ satisfying $\sum_{i=1}^{|\mathbf{K}|} \rho_i v_i \geq k$ for $\mathbf{P}$.

Second, if no vector $\mathbf{s}$ satisfies $v(\mathbf{S}) \geq k$ for $\mathbf{P5}$, there cannot be any vectors satisfying $\sum_{i=1}^{|\mathbf{K}|} \rho_i v_i \geq k$. We prove this by contradiction. Suppose there are vectors $\mathbf{s}$, $\mathbf{f}$, and $\mathbf{p}$ satisfying $\sum_{i=1}^{|\mathbf{K}|} \rho_i v_i \geq k$ for $\mathbf{P}$. Then, vector $\mathbf{s}$ also satisfies $v(\mathbf{S}) \geq k$ for the instance $I_{\mathbf{P5}} = I_{\mathbf{P}} \cup \{p_i^*\}$. □

Thus, the submodular maximization problem $\mathbf{P5}$ is polynomial reducible to problem $\mathbf{P}$, i.e., $\mathbf{P5} \leq_p \mathbf{P}$. Then, problem $\mathbf{P}$ is NP-hard because of the NP-hardness of $\mathbf{P5}$.

References

[1] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proc. IEEE ISCC*, 2012, pp. 59–66.

[2] A. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Commun. Surveys Tut.*, vol. 16, no. 1, pp. 393–413, 1st Quart. 2014.

[3] 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*, 2013, pp. 1285–1293.

[4] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[5] W. Zhang, Y. Wen, and D. Wu, "Energy-efficient scheduling policy for collaborative execution in mobile cloud computing," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 190–194.

[6] W. Zhang, Y. Wen, and D. O. Wu, "Collaborative task execution in mobile cloud computing under a stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 14, no. 1, pp. 81–93, Jan. 2015.

[7] Z. Cheng, P. Li, J. Wang, and S. Guo, "Just-in-time code offloading for wearable computing," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 74–83, Mar. 2015.

[8] X. Lyu and H. Tian, "Adaptive receding horizon offloading strategy under dynamic environment," *IEEE Commun. Lett.*, vol. 20, no. 5, pp. 878–881, May 2016.

[9] L. Yang, J. Cao, H. Cheng, and Y. Ji, "Multi-user computation partitioning for latency sensitive mobile cloud applications," *IEEE Trans. Comput.*, vol. 64, no. 8, pp. 2253–2266, Aug. 2015.

[10] V. Cardellini *et al.*, "A game-theoretic approach to computation offloading in mobile cloud computing," *Math. Program.*, vol. 157, pp. 1–29, 2013.

[11] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[12] 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.

[13] L. Yang, J. Cao, S. Tang, T. Li, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," in *Proc. 5th Int. Conf. CLOUD*, Jun. 2012, pp. 794–802.

[14] M. Rahimi, N. Venkatasubramanian, and A. Vasilakos, "Music: Mobility-aware optimal service allocation in mobile cloud computing," in *Proc. 6th Int. Conf. CLOUD*, Jun. 2013, pp. 75–82.

[15] S. Sardellitti, G. Scutari, and S. Barbarossa, "Distributed joint optimization of radio and computational resources for mobile cloud computing," in *Proc. 3rd Int. Conf. CloudNet*, Oct. 2014, pp. 211–216.

[16] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[17] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surveys Tut.*, vol. 16, no. 1, pp. 337–368, 1st Quart. 2014.

[18] H. Beyranvand, W. Lim, M. Maier, C. Verikoukis, and J. A. Salehi, "Backhaul-aware user association in FiWi enhanced LTE-A heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 6, pp. 2992–3003, Jun. 2015.

[19] S. Tombaz, P. Monti, F. Farias, M. Fiorani, L. Wosinska, and J. Zander, "Is backhaul becoming a bottleneck for green wireless access networks?" in *Proc. IEEE ICC*, Jun. 2014, pp. 4029–4035.

[20] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct.–Dec. 2009.

[21] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: Taxonomy and open challenges," *IEEE Commun. Surveys Tut.*, vol. 16, no. 1, pp. 369–392, 1st Quart. 2014.

[22] X. Lin, Y. Wang, Q. Xie, and M. Pedram, "Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment," *IEEE Trans. Serv. Comput.*, vol. 8, no. 2, pp. 175–186, Mar. 2015.

[23] S. Deb and P. Monogioudis, "Learning-based uplink interference management in 4 G LTE cellular systems," *IEEE/ACM Trans. Netw.*, vol. 23, no. 2, pp. 398–411, Feb. 2015.

[24] A. Goldsmith, *Wireless Communications*. Cambridge, MA, USA: Cambridge Univ. Press, 2005.

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

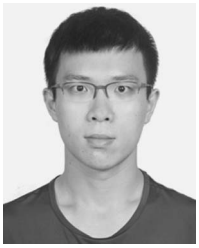
[26] Y. Cheng, M. Pesavento, and A. Philipp, "Joint network optimization and downlink beamforming for CoMP transmissions using mixed integer conic programming," *IEEE Trans. Signal Process.*, vol. 61, no. 16, pp. 3972–3987, Aug. 2013.

[27] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, MA, USA: Cambridge Univ. Press, 2004.

[28] J. Goffin, Z. Lu, and Y. Ye, "Complexity analysis of an interior cutting plane method for convex feasibility problems," *SIAM J. Optim.*, vol. 6, no. 3, pp. 638–652, 1996.

[29] Z. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1426–1438, Aug. 2006.

[30] S. Fujishige, *Submodular Functions and Optimization*, vol. 58. Amsterdam, The Netherlands: Elsevier, 2005.
[31] U. Feige, V. S. Mirrokni, and J. Vondrak, "Maximizing non-monotone submodular functions," *SIAM J. Comput.*, vol. 40, no. 4, pp. 1133–1153, 2011.
[32] X. Ge, S. Tu, G. Mao, C. X. Wang, and T. Han, "5G ultra-dense cellular networks," *IEEE Wireless Commun.*, vol. 23, no. 1, pp. 72–79, Feb. 2016.
[33] M. Kalil, A. Shami, A. Al-Dweik, and S. Muhaidat, "Low-complexity power-efficient schedulers for LTE uplink with delay-sensitive traffic," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4551–4564, Oct. 2015.
[34] "Further advancements for E-UTRA physical layer aspects," 3rd Generation Partnership Project, Sophia Antipolis Cedex, France, 3GPP TR 36.814, E-UTRA Access, Tech. Rep., 2010.



Xinchen Lyu received the B.E. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014, where he is currently working toward the Ph.D. degree.

His research interests include mobile cloud computing and radio resource management.

Hui Tian received the M.S. degree in microelectronics and the Ph.D. degree in circuits and systems from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1992 and 2003, respectively.

She is currently a Professor with BUPT, the Network Information Processing Research Center Director of the State Key Laboratory of Networking and Switching Technology, and the MAT Director of Wireless Technology Innovation Institute. Her current research interests include radio resource management, cross-layer optimization, mobile cloud computing, mobile-to-mobile communication, cooperative communication, and mobile social networks.

Cigdem Sengul received the M.S. and Ph.D. degrees in computer science from the University of Illinois at Urbana—Champaign (UIUC), Champaign, IL, USA, in 2003 and 2007, respectively.

Since 2002, she has been a Researcher on mobile and wireless networks in various research and development environments in different countries. Between 2012 and 2015, she was a Senior Lecturer in Communication Networks with Oxford Brookes University, Oxford, U.K. From 2008 to 2012, she was a Senior Research Scientist with Telekom Innovation Laboratories, Technical University of Berlin (main research unit of Deutsche Telekom), Berlin, Germany, and co-led the Berlin Open Wireless Lab Project. Between 2007 and 2008, she was a Researcher with the ASAP Research Group at INRIA, France, where she specialized in routing in delay-tolerant networks. She is currently a Senior Researcher with Nominet, Oxford, working on privacy in the Internet of things. Her work on wireless interference management and modeling and energy management has been published and demonstrated in leading conferences and journals, including the IEEE Conference on Computer Communications, the IEEE International Conference on Distributed Computing Systems, the IEEE International Conference on Communications, the *Association for Computing Machinery (ACM) Transactions on Sensor Networks*, and the ACM Special Interest Group on Data Communication. Her research was funded by Fulbright Scholarship and Department of Computer Science, UIUC fellowship, and Vodafone fellowship. Her current research interest is in the very exciting area of Internet of things, specifically privacy management, network management, and visualization.

Ping Zhang (SM'15) received the M.S. degree in electrical engineering from Northwestern Polytechnical University, Xi'an, China, in 1986 and the Ph.D. degree in electric circuits and systems from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1990.

He is currently a Professor with BUPT and the Director of the State Key Laboratory of Networking and Switching Technology, China. His research interests include cognitive wireless networks, fourth-generation mobile communication, fifth-generation mobile networks, communications factory test instruments, universal wireless signal detection instruments, and mobile Internet.

Dr. Zhang is the Executive Associate Editor-in-Chief on information sciences of the *Chinese Science Bulletin*, a Guest Editor of the *IEEE Wireless Communications Magazine*, and an Editor of *China Communications*. He received the First and Second Prizes from the National Technology Invention and Technological Progress Awards, as well as the First Prize Outstanding Achievement Award of Scientific Research in College.