Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading

Changsheng You, Kaibin Huang, Hyukjin Chae and Byoung-Hoon Kim

Abstract

Mobile-edge computation offloading (MECO) offloads intensive mobile computation to clouds located at the edges of cellular networks. Thereby, MECO is envisioned as a promising technique for prolonging the battery lives and enhancing the computation capacities of mobiles. In this paper, we study resource allocation for a multiuser MECO system based on time-division multiple access (TDMA) and orthogonal frequency-division multiple access (OFDMA). First, for the TDMA MECO system with infinite or finite computation capacity, the optimal resource allocation is formulated as a convex optimization problem for minimizing the weighted sum mobile energy consumption under the constraint on computation latency. The optimal policy is proved to have a threshold-based structure with respect to a derived offloading priority function, which yields priorities for users according to their channel gains and local computing energy consumption. As a result, users with priorities above and below a given threshold perform complete and minimum offloading, respectively. Moreover, for the cloud with finite capacity, a sub-optimal resource-allocation algorithm is proposed to reduce the computation complexity for computing the threshold. Next, we consider the OFDMA MECO system, for which the optimal resource allocation is formulated as a non-convex mixed-integer problem. To solve this challenging problem and characterize its policy structure, a sub-optimal low-complexity algorithm is proposed by transforming the OFDMA problem to its TDMA counterpart. The corresponding resource allocation is derived by defining an average offloading priority function and shown to have close-to-optimal performance in simulation.

Index Terms

Mobile edge computing, resource allocation, mobile computation offloading

C. You and K. Huang are with the Dept. of EEE at The University of Hong Kong, Hong Kong (Email: csyou@eee.hku.hk, huangkb@eee.hku.hk). H. Chae and B.-H. Kim are with LG Electronics, S. Korea (Email: hyukjin.chae@lge.com, bh.kim@lge.com). Part of this work has been submitted to IEEE Globecom 2016.
I. INTRODUCTION

The realization of Internet of Things (IoT) [1] will connect tens of billions of resource-limited mobiles, e.g., mobile devices, sensors and wearable computing devices, to Internet via cellular networks. The finite battery lives and limited computation capacities of mobiles pose challenges for designing IoT. One promising solution is to leverage mobile-edge computing [2] and offload intensive mobile computation to nearby clouds at the edges of cellular networks, called edge clouds, with short latency, referred to as mobile-edge computation offloading (MECO). In this paper, we consider a MECO system with a single edge cloud serving multiple users and investigate the energy-efficient resource allocation.

A. Prior Work

Mobile computation offloading (MCO) [3] (or mobile cloud computing) has been extensively studied in computer science, including system architectures (e.g., MAUI [4] and ThinkAir [5]), virtual machine migration [6] and power management [7]. It is commonly assumed that the implementation of MCO relies on a network architecture with a central cloud (e.g., a data center). This architecture has the drawbacks of high overhead and long backhaul latency [8], and will soon encounter the performance bottleneck of finite backhaul capacity in view of exponential mobile traffic growth. These issues can be overcome by MECO based on a network architecture supporting distributed mobile-edge computing. Among others, designing energy-efficient control policies is a key challenge for a MECO system.

Energy-efficient MECO requires the joint design of MCO and wireless communication techniques. Recent years have seen research progress on this topic for both single-user [9]–[12] and multiuser [13]–[17] MECO systems. For a single-user MECO system, the optimal offloading decision policy was derived in [9] by comparing the energy consumption of optimized local computing (with variable CPU cycles) and offloading (with variable transmission rates). This framework was further developed in [10] and [11] to enable adaptive offloading powered by wireless energy transfer and energy harvesting, respectively. Moreover, dynamic offloading was integrated with adaptive LTE/WiFi link selection in [12] to achieve higher energy efficiency. For multiuser MECO systems, the control policies for energy savings are more complicated. In [13], distributed computation offloading for multiuser MECO at a single cloud was designed using game theory for both energy-and-latency minimization at mobiles. A multi-cell MECO system was considered in [14], where the radio and computation resources were jointly allocated to minimize the mobile energy consumption under offloading latency constraints. With the
coexistence of central and edge clouds, the optimal user scheduling for offloading to different clouds was studied in [15]. In addition to total mobile energy consumption, cloud energy consumption for computation was also minimized in [16] by designing the mapping between clouds and mobiles for offloading using game theory. The cooperation among clouds was further investigated in [17] to maximize the revenues of clouds and meet mobiles’ demand via resource pool sharing. Prior work on MECO resource allocation focuses on complex algorithmic designs and yields little insight into the optimal policy structures. In contrast, for a multiuser MECO system based on time-division multiple access (TDMA), the optimal resource-allocation policy is shown in the current work to have a simple threshold-based structure with respect to a derived offloading priority function. This insight is used for designing the low-complexity resource-allocation policy for a orthogonal frequency-division multiple access (OFDMA) MECO system.

Resource allocation for traditional multiple-access communication systems has been widely studied, including TDMA (see e.g., [18]), OFDMA (see e.g., [19]) and code-division multiple access (CDMA) (see e.g., [20]). Moreover, it has been designed for existing networks such as cognitive radio [21] and heterogenous networks [22]. Note that all of them only focus on the radio resource allocation. In contrast, for the newly proposed MECO systems, both the computation and radio resource allocation at edge clouds are jointly optimized for the maximum mobile energy savings, making the algorithmic design more complex.

B. Contribution and Organization

This paper considers resource allocation in a multiuser MECO system based on TDMA and OFDMA. Multiple mobiles are required to compute different computation loads with the same latency constraint. Assuming that computation data can be split for separate computing, each mobile can simultaneously perform local computing and offloading. Moreover, the edge cloud is assumed to have perfect knowledge of local computing energy consumption, channel gains and fairness factors at all users, which is used for designing centralized resource allocation to achieve the minimum weighted sum mobile energy consumption. In the TDMA MECO system, the optimal threshold-based policy is derived for both the cases of infinite and finite cloud capacities. For the OFDMA MECO system, a sub-optimal low-complexity algorithm is proposed to solve the formulated non-convex mixed-integer resource allocation problem.

The contributions of current work are summarized as follows.

- **TDMA MECO with infinite cloud capacity**: For TDMA MECO with infinite (computation) capacity, a convex optimization problem is formulated to minimize the weighted sum mobile
energy consumption under the time-sharing constraint. To solve this problem, an offloading priority function is derived that yields priorities for users and depends on their channel gains and local computing energy consumption. Based on this, the optimal policy is proved to have a threshold-based structure that determines complete and minimum offloading for users with priorities above and below a given threshold, respectively.

- **TDMA MECO with finite cloud capacity**: The above results are extended to the case of finite capacity. Specifically, the optimal resource allocation policy is derived by defining an effective offloading priority function and modifying the threshold-based policy as derived for the infinite-capacity cloud. To reduce the complexity arising from a two-dimension search for Lagrange multipliers, a simple and low-complexity algorithm is proposed based on the approximated offloading priority order. This reduces the said search to a one-dimension search, which is shown by simulation to have close-to-optimal performance.

- **OFDMA MECO**: For an infinite-capacity cloud based on OFDMA, the insight of priority-based policy structure of TDMA is used for optimizing its resource allocation. Specifically, to solve the corresponding non-convex mixed-integer optimization problem, a sub-optimal low-complexity algorithm is proposed. Using average sub-channel gains, the OFDMA resource allocation problem is transformed into its TDMA counterpart. Based on this, the initial resource allocation and offloaded data allocation can be determined by defining an average offloading priority function. Moreover, the integer sub-channel assignment is performed according to the offloading priority order, followed by adjustments of offloaded data allocation over assigned sub-channels for each user. The proposed algorithm is extended to the finite-capacity cloud case and shown to have close-to-optimal performance by simulation.

The reminder of this paper is organized as follows. Section II introduces the system model. Section III presents the problem formulation for multiuser MECO based on TDMA. The resource allocation policies are characterized in Section IV and Section V for both the cases of infinite and finite cloud capacities, respectively. The above results are extended in Section VI for the OFDMA system. Simulation results and discussion are given in Section VII, followed by the conclusion in Section VIII.

**II. System Model**

Consider a multiuser MECO system shown in Fig. 1 that comprises $K$ single-antenna mobiles, denoted by a set $\mathcal{K} = \{1, 2, \cdots, K\}$, and one single-antenna base station (BS) that is the gateway of an edge cloud. These mobiles are required to compute different computation loads with the
same latency constraint. Assume that the BS has perfect knowledge of multiuser channel gains, local computing energy per bit and sizes of input data at all users, which can be obtained by feedback. Using these information, the BS selects offloading users, determines the offloaded data sizes and allocates radio resource to offloading users with the criterion of minimum weighted sum mobile energy consumption.

A. Multiple-Access Model

Both the TDMA and OFDMA systems are considered as follows. For the TDMA system, time is divided into slots each with a duration of $T$ seconds. As shown in Fig. 1, each slot comprises two sequential phases for 1) mobile offloading or local computing and 2) cloud computing

\footnotetext{For asynchronous computation offloading among users, the maximum additional latency for each user is one time slot. Moreover, this framework can be extended to predictive computing by designing control policies for the coming data.}
and downloading of computation results from the edge cloud to mobiles. Cloud computing has small latency; the downloading consumes negligible mobile energy and furthermore is much faster than offloading due to relative smaller sizes of computation results. For these reasons, the second phase is assumed to have a negligible duration compared with the first phase and not considered in resource allocation. For the OFDMA system, the total bandwidth is divided into multiple orthogonal sub-channels and each sub-channel can be assigned to at most one user. The offloading mobiles will be allocated with one or more sub-channels.

Considering an arbitrary slot in TDMA/OFDMA, the BS schedules a subset of users for complete/partial offloading. The user with partial or no offloading computes a fraction of or all input data, respectively, using a local CPU.

B. Local-Computing Model

Assume that the CPU frequency is fixed at each user and may vary over users. Consider an arbitrary time slot. Following the model in [13], let $C_k$ denote the number of CPU cycles required for computing 1-bit of input data at the $k$-th mobile, and $P_k$ the energy consumption per cycle for local computing at this user. Then the product $C_kP_k$ gives computing energy per bit. As shown in Fig. 2, mobile $k$ is required to compute $R_k$-bit input data within the time slot, out of which $\ell_k$-bit is offloaded and $(R_k - \ell_k)$-bit is computed locally. Then the total energy consumption for local computing at mobile $k$, denoted as $E_{loc,k}$, is given by $E_{loc,k} = (R_k - \ell_k)C_kP_k$. Let $F_k$ denote the computation capacity of mobile $k$ that is measured by the number of CPU cycles per second. Under the computation latency constraint, it has $C_k(R_k - \ell_k)/F_k \leq T$. As a result, the offloaded data at mobile $k$ has the minimum size of $\ell_k \geq m_k^+$ with $m_k = R_k - \frac{F_kT}{C_k}$, where the function $(x)^+ = \max\{x, 0\}$.

C. Computation-Offloading Model

First, consider the TDMA system for an arbitrary time slot. Let $h_k$ denote the channel gain for mobile $k$ that is constant during offloading duration, and $p_k$ its transmission power. Then the achievable rate (in bits/s), denoted by $r_k$, is given by:

$$ r_k = B \log \left(1 + \frac{p_kh_k^2}{N_0}\right) $$  

(1)

where $B$ and $N_0$ are the bandwidth and the variance of complex white Gaussian channel noise, respectively. The fraction of slot allocated to mobile $k$ for offloading is denoted as $t_k$ with $t_k \geq 0$, where $t_k = 0$ corresponds to no offloading. For the case of offloading ($t_k > 0$), the transmission rate is fixed as $r_k = \ell_k/t_k$ since this is the most energy-efficient transmission policy under a
deadline constraint. Define a function \( f(x) = N_0(2\frac{x}{B} - 1) \). It follows from (1) that the energy consumption for offloading at mobile \( k \) is
\[
E_{\text{off},k} = p_k t_k = \frac{t_k}{h_k^2} f\left(\frac{\ell_k}{t_k}\right).
\]
Note that if either \( \ell_k = 0 \) or \( t_k = 0 \), \( E_{\text{off},k} \) is equal to zero.

Next, consider the OFDMA system with \( N \) sub-channels, denoted by a set \( \mathcal{N} = \{1, 2, \cdots, N\} \). Let \( p_{k,n} \) and \( h_{k,n} \) denote the transmission power and channel gain of mobile \( k \) on the \( n \)-th sub-channel. Define \( \rho_{k,n} \in \{0, 1\} \) as the sub-channel assignment indicator variable where \( \rho_{k,n} = 1 \) indicates that sub-channel \( n \) is assigned to mobile \( k \), and verse vice. Then the achievable rate (in bits/s) follows:
\[
r_{k,n} = \rho_{k,n} \bar{B} \log \left(1 + \frac{p_{k,n} h_{k,n}^2}{N_0}\right)
\]
where \( \bar{B} \) and \( N_0 \) are the bandwidth and noise power for each sub-channel, respectively. Let \( \ell_{k,n} = r_{k,n} T \) denote the offloaded data size over the offloading duration time \( T \), which can be set as the OFDMA symbol duration. The corresponding offloading energy consumption can be expressed as below, which is similar to that in [19], namely
\[
E_{\text{off},k,n} = \rho_{k,n} p_{k,n} T = \rho_{k,n} \frac{h_{k,n}^2}{\bar{h}_{k,n}^2} f\left(\frac{\ell_{k,n}}{\rho_{k,n}}\right)
\]
where \( \bar{h}_{k,n}^2 = \frac{h_{k,n}^2}{T} \) and \( f(x) = N_0(2\frac{x}{B} - 1) \).

D. Cloud-Computing Model

For the case of an edge cloud with finite (computation) capacity denoted as \( F \), the finite capacity is reflected in one of the following two constraints. The first one upper-bounds CPU cycles of sum offloaded data that can be handled by the cloud in each time slot: \( \sum_{k=1}^{K} C_k \ell_k \leq F \). This constraint ensures negligible cloud computing latency. The other one considers non-negligible computing time at the cloud as in [23], [24], given as \( t_{\text{comp}} = (\sum_{k=1}^{K} \ell_k C_k)/F \), which is factored into the latency constraint in the sequel.

III. Multiuser MECO for TDMA: Problem Formulation

In this section, resource allocation for multiuser MECO based on TDMA is formulated as an optimization problem. The objective is to minimize the weighted sum mobile energy consumption:
\[
\sum_{k=1}^{K} \beta_k (E_{\text{off},k} + E_{\text{loc},k}),
\]
where the positive weight factors \( \{\beta_k\} \) account for fairness among
mobiles. Under the constraints on time-sharing, cloud computation capacity and computation latency, the resource allocation problem is formulated as follows:

\[
\min_{\{\ell_k, t_k\}} \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f \left( \frac{\ell_k}{t_k} \right) + (R_k - \ell_k)C_kP_k \right]
\]

s.t. \[\sum_{k=1}^{K} t_k \leq T, \sum_{k=1}^{K} C_k \ell_k \leq F,\]

\[t_k \geq 0, \ m_k^+ \leq \ell_k \leq R_k, \; k \in K.\]  \hfill (P1)

Some basic characteristics of Problem P1 are given in the following two lemmas, proved in Appendix A and B, respectively.

**Lemma 1.** Problem P1 is a convex optimization problem.

**Lemma 2.** The feasibility condition for Problem P1 is: \[\sum_{k=1}^{K} m_k^+ C_k \leq F.\]

Lemma 2 shows that whether the cloud capacity constraint is satisfied determines the feasibility of this optimization problem, while the time-sharing constraint can always be satisfied and only affects the mobile energy consumption.

Assume that Problem P1 is feasible. The direct solution for Problem P1 using the dual-decomposition approach (the Lagrange method) requires iterative computation and yields little insight into the structure of the optimal policy. To address these issues, we adopt a two-stage solution approach that requires first solving Problem P2 below, which follows from Problem P1 by relaxing the constraint on cloud capacity:

\[
\min_{\{\ell_k, t_k\}} \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f \left( \frac{\ell_k}{t_k} \right) + (R_k - \ell_k)C_kP_k \right]
\]

s.t. \[\sum_{k=1}^{K} t_k \leq T, \]

\[t_k \geq 0, \ m_k^+ \leq \ell_k \leq R_k, \; k \in K.\]  \hfill (P2)

If the solution for Problem P2 violates the constraint on cloud capacity, Problem P1 is then incrementally solved building on the solution for Problem P2. This approach allows the optimal policy to be shown to have the said threshold-based structure and also facilitates the design of low-complexity close-to-optimal resource-allocation algorithm. It is interesting to note that Problem P2 corresponds to the case where the edge cloud has infinite capacity. The detailed procedures for solving Problems P1 and P2 are presented in the two subsequent sections.
In this section, by solving Problem P2 using the Lagrange method, we derive a threshold-based policy for the optimal resource allocation. Moreover, the policy is simplified for several special cases.

To solve Problem P2, the partial Lagrange function is defined as

\[ L = \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f \left( \frac{\ell_k}{t_k} \right) + (R_k - \ell_k)C_kP_k \right] + \lambda \left( \sum_{k=1}^{K} t_k - T \right) \]

where \( \lambda \geq 0 \) is the Lagrange multiplier associated with the time-sharing constraint. For ease of notation, define a function

\[ g(x) = f(x) - xf'(x). \]

Let \( \{\ell_k^{\ast(2)}, t_k^{\ast(2)}\} \) denote the optimal solution for Problem P2 that always exists according to Lemma 2. Then applying KKT conditions leads to the following necessary and sufficient conditions:

\[ \frac{\partial L}{\partial \ell_k^{\ast(2)}} = \beta_k f' \left( \frac{\ell_k^{\ast(2)}}{t_k^{\ast(2)}} \right) - \beta_k C_kP_k \begin{cases} > 0, & \ell_k^{\ast(2)} = m_k^+ \\ = 0, & \ell_k^{\ast(2)} \in (m_k^+, R_k), \forall k \in \mathcal{K} \\ < 0, & \ell_k^{\ast(2)} = R_k \end{cases} \]

\[ \frac{\partial L}{\partial t_k^{\ast(2)}} = \frac{\beta_k g \left( \frac{\ell_k^{\ast(2)}}{t_k^{\ast(2)}} \right)}{h_k^2} + \lambda^* \begin{cases} > 0, & t_k^{\ast(2)} = 0 \\ = 0, & t_k^{\ast(2)} > 0 \end{cases}, \forall k \in \mathcal{K} \]

\[ \sum_{k=1}^{K} t_k^{\ast(2)} \leq T, \quad \lambda^* \left( \sum_{k=1}^{K} t_k^{\ast(2)} - T \right) = 0. \]

Note that for \( \ell_k^{\ast(2)} \in (m_k^+, R_k) \) and \( t_k^{\ast(2)} > 0 \), it can be derived from (5a) and (5b) that

\[ \frac{\ell_k^{\ast(2)}}{t_k^{\ast(2)}} = f^{-1} \left( C_kP_kh_k^2 \right) = g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right). \]

Based on these conditions, the optimal policy for resource allocation is characterized in the following sub-sections.

### A. Offloading Priority Function

Define a (mobile) offloading priority function, which is essential for the optimal resource allocation, as follows:

\[ \varphi(\beta_k, C_k, P_k, h_k) = \begin{cases} \frac{\beta_k N_0}{h_k^2} \left( v_k \ln v_k - v_k + 1 \right), & v_k \geq 1 \\ 0, & v_k < 1 \end{cases} \]

with the constant \( v_k \) defined as

\[ v_k = \frac{BC_kP_kh_k^2}{N_0 \ln 2}. \]
This function is derived by solving a useful equation as shown in the following lemma.

**Lemma 3.** Given \( v_k \geq 1 \), the offloading priority function \( \varphi(\beta_k, C_k, P_k, h_k) \) in (7) is the root of following equation with respect to \( x \):

\[
f'(C_k P_k h_k^2) = g^{-1}\left(\frac{-h_k^2 x}{\beta_k}\right),
\]

Lemma 3 is proved in Appendix C. The function generates an offloading priority value, \( \varphi_k = \varphi(\beta_k, C_k, P_k, h_k) \), for mobile \( k \), depending on corresponding variables quantifying fairness, local computing and channel. The amount of offloaded data by a mobile grows with an increasing offloading priority as shown in the next sub-section. It is useful to understand the effects of parameters on the offloading priority that are characterized as follows.

**Lemma 4.** Given \( v \geq 1 \), \( \varphi(\beta, C, P, h) \) is a *monotone increasing function* for \( \beta, C, P \) and \( h \).

Lemma 4 can be easily proved by deriving the first derivatives of \( \varphi \) with respect to each parameter. Moreover, it is consistent with the intuition that, to reduce energy consumption by offloading, the BS should schedule those mobiles having high computing energy consumption per bit (i.e., large \( C \) and \( P \)) or good channels (i.e., large \( h \)).

**Remark 1** (Effects of Parameters on the Offloading Priority). It can be observed from (7) and (8) that the offloading priority scales with local computing energy per bit \( CP \) approximately as \( (CP) \ln(CP) \) and with the channel gain \( h \) approximately as \( \ln h \). The former scaling is much faster than the latter. This shows that the computing energy per bit is dominant over the channel on determining whether to offload.

**B. Optimal Resource-Allocation Policy**

Based on conditions in (5a)-(5c) and Lemma 3, the main result of this section is derived, given in the following theorem.

**Theorem 1** (Optimal Resource-Allocation Policy). Consider the case of infinite cloud computation capacity. The optimal policy solving Problem P2 has the following structure.

1) If \( v_k \leq 1 \) and the minimum offloaded data size \( m_k^+ = 0 \) for all \( k \), none of these users performs offloading, i.e.,

\[
\ell_k^{(2)} = t_k^{(2)} = 0 \quad k \in \mathcal{K}.
\]
2) If there exists mobile $k$ such that $\nu_k > 1$ or $m^+_k > 0$, for $k \in \mathcal{K}$,

$$
\ell_k^{(2)} = \begin{cases} 
m^+_k, & \varphi_k < \lambda^* \\
[ m^+_k, R_k], & \varphi_k = \lambda^* \\
R_k, & \varphi_k > \lambda^*
\end{cases}
$$

and

$$
t_k^{(2)} = \frac{\ln 2}{B \left[ W_0 \left( \frac{\lambda^* h_k^2 / \beta_k - N_0}{N_0 e} \right) + 1 \right]} \times \ell_k^{(2)}
$$

where $W_0(x)$ is the Lambert function and $\lambda^*$ is the optimal value of the Lagrange multiplier satisfying the active time-sharing constraint: $\sum_{k=1}^K t_k^{(2)} = T$.

**Proof:** See Appendix D.

Theorem 1 reveals that the optimal resource-allocation policy has a threshold-based structure when offloading saves energy. In other words, since the exact case of $\varphi_k = \lambda^*$ rarely occurs in practice, the optimal policy makes a binary offloading decision for each mobile. Specifically, if the corresponding offloading priority exceeds a given threshold, namely $\lambda^*$, the mobile should offload all input data to the edge cloud; otherwise, the mobile should offload only the minimum amount of data under the computation latency constraint. This result is consistent with the intuition that the greedy method can lead to the optimal resource allocation. Note that there are two groups of users selected to perform the minimum offloading. One is users for which offloading cannot save energy since they have bad channels or low local computing energy such that $\nu_k \leq 1$ and $\varphi_k = 0$. The second group are those with relatively low offloading priorities, i.e., $\varphi_k < \lambda^*$; they cannot perform complete offloading due to the limited radio resource.

**Remark 2 (Offloading or Not?).** For a conventional TDMA communication system, continuous transmission by at least one mobile is always advantageous under the criterion of minimum sum energy consumption [18]. However, this does not always hold for a TDMA MECO system where no offloading for all users may be preferred as shown in Theorem 1. Offloading is not necessary expect for two cases. First, there exists at least one mobile whose input-data size is too large such that complete local computing fails to meet the latency constraint. Second, some mobile has a sufficient high value for the product $C_k P_k h_k^2$, indicating that energy savings can be achieved by offloading because of high channel gain or large local computing energy consumption.

**Remark 3 (Offloading Rate).** It can be observed from Theorem 1 that the offloading rate, defined as $\ell_k^{(2)}/t_k^{(2)}$ for mobile $k$, is determined only by the channel gain and fairness factor while other factors, namely $C_k$ and $P_k$, affect the offloading decision. The rate increases with a growing channel gain and vice versa since a large channel gain supports a higher transmission rate or reduces transmission power, making offloading desirable for reducing energy consumption.
Algorithm 1 Optimal Algorithm for Solving Problem P2.

- **Step 1** [Initialize]: Let $\lambda_\ell = 0$ and $\lambda_h = \lambda_{\text{max}}$. According to Theorem 1, obtain $T_\ell = \sum_{k=1}^{K} t_{k,\ell}^{(2)}$ and $T_h = \sum_{k=1}^{K} t_{k,h}^{(2)}$, where $\{t_{k,\ell}^{(2)}\}$ and $\{t_{k,h}^{(2)}\}$ are the allocated fractions of slot for the cases of $\lambda_\ell$ and $\lambda_h$, respectively.

- **Step 2** [Bisection search]: While $T_\ell \neq T$ and $T_h \neq T$, update $\{\lambda_\ell, \lambda_h\}$ as follows.
  1. Define $\lambda_m = (\lambda_\ell + \lambda_h)/2$ and compute $T_m$.
  2. If $T_m = T$, then $\lambda^* = \lambda_m$ and the optimal policy can be determined. Otherwise, if $T_m < T$, let $\lambda_h = \lambda_m$ and if $T_m > T$, let $\lambda_\ell = \lambda_m$.

**Remark 4 (Low-Complexity Algorithm).** The traditional method for solving Problem P2 is the block-coordinate descending algorithm which performs iterative optimization of the two sets of variables, $\{\ell_k\}$ and $\{t_k\}$, resulting in high computation complexity. In contrast, by exploiting the threshold-based structure of the optimal resource-allocation policy in Theorem 1, the proposed solution approach, described in Algorithm 1, needs to perform only a one-dimension search for $\lambda^*$, reducing the computation complexity significantly. To facilitate the search, next lemma gives the range of $\lambda^*$, which can be easily proved from Theorem 1 and omitted for simplicity.

**Lemma 5.** When there is at least one offloading mobile, the optimal Lagrange multiplier $\lambda^*$ satisfies: $0 \leq \lambda^* \leq \lambda_{\text{max}} = \max_k \varphi_k$.

Furthermore, with the assumption of infinite cloud capacity, the effects of finite radio resource (i.e., the TDMA time-slot duration) are characterized as given below in terms of the number of offloading users, which can be easily derived according to Theorem 1.

**Proposition 1** (Exclusive Mobile Computation Offloading). For TDMA MECO with offloading users, only one mobile can offload computation if $T \leq \frac{R_m}{B \log_2 \left( \frac{BC_m P_m h_m^2}{N_0 \ln 2} \right)}$ where $m = \arg \max_k \varphi_k$.

It indicates that short time slot limits the number of offloading users. From another perspective, it means that if the winner user $m$ has excessive data, it will take up all the resource.

**Proposition 2** (Inclusive Mobile Computation Offloading). All offloading-desired mobiles (defined as for which, it has $\varphi_k > 0$) will completely offload computation if

$$T \geq \sum_{k \in \mathcal{O}_1} \frac{R_k \ln 2}{B \left( W_0 \left( \frac{\lambda_{\text{min}} h_k^2 / \beta_k - N_0}{N_0 e} \right) + 1 \right)} + \sum_{k \in \mathcal{O}_2} \frac{m^+_k \ln 2}{B \left( W_0 \left( \frac{\lambda_{\text{min}} h_k^2 / \beta_k - N_0}{N_0 e} \right) + 1 \right)}$$
where $O_1 = \{k \mid \varphi_k > 0\}$, $O_2 = \{k \mid \varphi_k = 0\}$ and $\lambda_{\text{min}} = \min_{k \in O_1} \varphi_k$.

Proposition 2 reveals that when $T$ exceeds a given threshold, the offloading-desired mobiles for which offloading brings energy savings, will offload all computation to the cloud. In this case, the radio resource is not the bottleneck in terms of offloading more users. Moreover, the longer the time slot is, the less the weighted sum mobile energy consumption is.

Remark 5 (Which Resource is Bottleneck?). Proposition 1 and 2 suggest that as the radio resource continuously increases, the cloud will become the performance bottleneck and the assumption of infinite cloud capacity will not hold. For a short time-slot duration, only a few users can offload computation. This requires a fraction of computation such that the cloud can be regarded as having infinite capacity. However, when the time-slot duration is large, it not only saves energy by offloading but also allows more users to offload computation, which potentially exceeds the cloud capacity. The case of finite-cloud capacity will be considered in the sequel.

C. Special Cases

The optimal resource-allocation policies for several special cases considering equal fairness factors are discussed as follows.

1) Uniform Channels and Local Computing: Consider the simplest case where $\{h_k, C_k, P_k\}$ are identical for all $k$. Then all mobiles have uniform offloading priorities. In this case, for the optimal resource allocation, all mobiles can offload arbitrary data sizes so long as the sum offloaded data size satisfies the following constraint: $\sum_{k=1}^{K} \ell_k^{(2)} \leq TB \log_2 \left( \frac{BCP h^2}{N_0 \ln 2} \right)$.

2) Uniform Channels: Consider the case of $h_1 = h_2 \cdots = h_K = h$. The offloading priority for each mobile, say mobile $k$, is only affected by the corresponding local-computing parameters $P_k$ and $C_k$. Without loss of generality, assume that $P_1 C_1 \leq P_2 C_2 \cdots \leq P_K C_K$. Then the optimal resource-allocation policy is given in the following corollary of Theorem 1.

**Corollary 1.** Assume infinite cloud capacity, $h_1 = h_2 \cdots = h_K = h$ and $P_1 C_1 \leq P_2 C_2 \cdots \leq P_K C_K$. Let $k_t$ denote the index such that $\varphi_k < \lambda^*$ for all $k < k_t$ and $\varphi_k > \lambda^*$ for all $k \geq k_t$, neglecting the rare case where $\varphi_k = \lambda^*$. The optimal resource-allocation policy is given as follows: for $k \in K$,

\[
\ell_k^{(2)} = \begin{cases} 
R_k, & k \geq k_t \\
m_k^+, & \text{otherwise}
\end{cases}
\]

and

\[
l_k^{(2)} = \frac{\ln 2}{B \left[ W_0 \left( \frac{\lambda \beta h^2}{N_0 c} \right) + 1 \right]} \times \ell_k^{(2)}.
\]
The result shows that the optimal resource-allocation policy follows a greedy approach that selects mobiles in a descending order of energy consumption per bit for complete offloading until the time-sharing duration is fully utilized.

3) Uniform Local Computing: Consider the case of $C_1 P_1 = C_2 P_2 \cdots = C_K P_K$. Similar to the previous case, the optimal resource-allocation policy can be shown to follow the greedy approach that selects mobiles for complete offloading in the descending order of channel gains.

V. MULTIUSER MECO FOR TDMA: FINE CLOUD CAPACITY

In this section, we consider the case of finite cloud capacity and analyze the optimal resource-allocation policy for solving Problem P1. The policy is shown to also have a threshold-based structure as the infinite-capacity counterpart derived in the preceding section. Both the optimal and sub-optimal algorithms are presented for policy computation. The results are extended to a cloud with non-negligible computing time.

A. Optimal Resource-Allocation Policy

To solve the convex Problem P1, the corresponding partial Lagrange function is written as

$$
\tilde{L} = \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f \left( \frac{\ell_k}{t_k} \right) + (R_k - \ell_k)C_k P_k \right] + \lambda \left( \sum_{k=1}^{K} t_k - T \right) + \mu \left( \sum_{k=1}^{K} C_k \ell_k - F \right)
$$

where $\mu \geq 0$ is the Lagrange multiplier corresponding to the cloud capacity constraint. Using the above Lagrange function, it is straightforward to show that the corresponding KKT conditions can be modified from their infinite-capacity counterparts in (5a)-(5c) by replacing $P_k$ with $\tilde{P}_k = P_k - \mu$, called the effective computation energy per cycle. The resultant effective offloading priority function, denoted as $\tilde{\phi}_k$, can be modified accordingly from that in (7) as

$$
\tilde{\phi}(\beta_k, C_k, P_k, h_k, \tilde{\mu}^*) = \begin{cases} 
\frac{\beta_k N_0}{h_k^2} (\tilde{v}_k \ln \tilde{v}_k - \tilde{v}_k + 1), & \tilde{v}_k \geq 1 \\
0, & \tilde{v}_k < 1
\end{cases},
$$

where

$$
\tilde{v}_k = \frac{BC_k (P_k - \tilde{\mu}^*) h_k^2}{N_0 \ln 2}.
$$

The relationship between cloud capacity and the optimal Lagrange multiplier $\tilde{\mu}^*$ is given below.

Lemma 6. A cloud with smaller capacity $F$ leads to a larger Lagrange multiplier $\tilde{\mu}^*$.

This lemma indicates that compared with $\phi_k$ in (7) for the case of infinite-capacity cloud, the effective offloading priority function here is also determined by the cloud capacity. Moreover,
both the effective local computing energy and priority function decrease when the cloud has less computation capacity.

Based on above discussion, the main result of this section follows as shown below.

**Theorem 2.** Consider the finite-capacity cloud with upper-bounded offloaded computation. The optimal policy solving Problem P1 has the same structure as that in Theorem 1 and is expressed in terms of the priority function $\tilde{\varphi}_k$ in (9) and optimized Lagrange multipliers $\{\tilde{\lambda}^*, \tilde{\mu}^*\}$.

**Remark 6 (Variation of Offloading Priority Order).** Since $\tilde{\mu}^* > 0$, it has $\tilde{\varphi}_k < \varphi_k$ for all $k$. Therefore, the offloading priority order may be different with that in infinite-capacity cloud, due to the varying decreasing rates of offloading priorities. The reason is that the finite-capacity cloud should make the tradeoff between energy savings and computation burden. To this end, it will select mobiles for offloading that can save significant energy and require less computation for each bit of data.

Computing the threshold for the optimal resource-allocation policy requires a two-dimension search over the Lagrange multipliers $\{\tilde{\lambda}^*, \tilde{\mu}^*\}$, described in Algorithm 2. For an efficient search, it is useful to limit the range of $\tilde{\lambda}^*$ and $\tilde{\mu}^*$ shown as below, which can be easily proved.

**Lemma 7.** When there is at least one offloading mobile, the optimal Lagrange multipliers $\{\tilde{\lambda}^*, \tilde{\mu}^*\}$ satisfy: $0 \leq \tilde{\lambda}^* \leq \lambda_{\text{max}}$ and $0 \leq \tilde{\mu}^* \leq \mu_{\text{max}} = \max_k \left\{ P_k - \frac{N_0 \ln 2}{BC_k h_k^2} \right\}$, where $\lambda_{\text{max}}$ is defined in Lemma 5.

Note that $\tilde{\mu}^* = 0$ corresponds to the case of infinite-capacity cloud and $\tilde{\mu}^* = \mu_{\text{max}}$ to the case where offloading yields no energy savings for any mobile.

**B. Sub-Optimal Resource-Allocation Policy**

To reduce the computation complexity of Algorithm 2 due to the two-dimension search, one simple sub-optimal policy is proposed as shown in Algorithm 3. The key idea is to decouple the computation and radio resource allocation. In Step 2, based on the approximated offloading priority in (7) for the case of infinite-capacity cloud, we allocate the computation resource to mobiles with high offloading priorities. Step 3 optimizes the corresponding fractions of slot given offloaded data. This sub-optimal algorithm has low complexity requiring only a one-dimension search. Moreover, its performance is shown by simulation to be close-to-optimal in the sequel.
Algorithm 2 Optimal Algorithm for Solving Problem P1.

- **Step 1** [Check solution for Problem P2]: Perform Algorithm 1. If \( \sum_{k=1}^{K} \ell^*_k(2) \leq F \), the optimal policy is given in Theorem 1. Otherwise, go to Step 2.

- **Step 2** [Initialize]: Let \( \mu_\ell = 0 \) and \( \mu_h = \mu_{\text{max}} \). Based on Theorem 2, obtain \( F_\ell = \sum_{k=1}^{K} C_k \ell^*_{k,\ell} \) and \( F_h = \sum_{k=1}^{K} C_k \ell^*_{k,h} \), where \( \{\ell^*_{k,\ell}\} \) and \( \{\ell^*_{k,h}\} \) are the offloaded data sizes for \( \mu_\ell \) and \( \mu_h \), respectively, involving the one-dimension search for \( \tilde{\lambda}^* \).

- **Step 3** [Bisection search]: While \( F_\ell \neq F \) and \( F_h \neq F \), update \( \{\mu_\ell, \mu_h\} \) as follows.
  1. Define \( \mu_m = (\mu_\ell + \mu_h)/2 \) and compute \( F_m \).
  2. If \( F_m = F \), then \( \tilde{\mu}^* = \mu_m \) and the optimal policy can be determined. Otherwise, if \( F_m < F \), let \( \mu_h = \mu_m \) and if \( F_m > F \), let \( \mu_\ell = \mu_m \).

Algorithm 3 Sub-optimal Algorithm for Solving Problem P1.

- **Step 1**: Perform Algorithm 1. If \( \sum_{k=1}^{K} \ell^*_k(2) \leq F \), Theorem 1 gives the optimal policy. Otherwise, go to Step 2.

- **Step 2**: Based on offloading priorities in (7), offload the data from mobiles in the descending order of offloading priority until the cloud computation capacity is fully occupied, i.e., \( \sum_{k=1}^{K} C_k \ell^*_k = F \).

- **Step 3**: With \( \{\ell^*_k\} \) derived in Step 2, perform one-dimension search for \( \lambda^* \) such that \( \sum_{k=1}^{K} \ell_k^* = T \) where \( t_k^* = \frac{l_k^* \ln 2}{B[W_0(\frac{\lambda^* h_k^2/B_k - N_0}{N_0 e}) + 1]} \).

C. Extension: MECO with Non-Negligible Computing Time

Consider another finite-capacity cloud for which the computing time is non-negligible. Surprisingly, the resultant optimal policy is also threshold based, with respect to a different offloading priority function.

Assume that the edge cloud performs load balancing for the uploaded computation as in [23], [24]. In other words, the CPU cycles are proportionally allocated for each user such that all users experience the same computing time \( (\sum_{k=1}^{K} C_k \ell_k) / F \) (see Section II-D). Then, considering both the data transmission and computing time reformulates the latency constraint as:

\[
\sum_{k=1}^{K} \frac{C_k \ell_k}{F} + \sum_{k=1}^{K} t_k \leq T.
\]  

The resultant optimization problem for minimizing weighted sum mobile energy consumption
is re-written by
\[
\min_{\{\ell_k, t_k\}} \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f\left(\frac{\ell_k}{t_k}\right) + (R_k - \ell_k)C_kP_k \right] \\
\text{s.t.} \quad \frac{\sum_{k=1}^{K} C_k\ell_k}{F} + \sum_{k=1}^{K} t_k \leq T,
\] (P3)

The key challenge of Problem P3 is that the amount of offloaded data size of each user has effects on offloading energy consumption, offloading duration and cloud computing time, complicating the problem. The following lemma gives the feasibility condition, proved in Appendix E.

**Lemma 8.** The feasibility condition for Problem P3 is \( \sum_{k=1}^{K} \frac{C_k m_k^+}{F} < T \).

Note that in the above lemma, the case \( \sum_{k=1}^{K} \frac{C_k m_k^+}{F} = T \) makes Problem P3 infeasible since the resultant offloading time \( (t_k = 0) \) cannot enable computation offloading.

Similarly, to solve Problem P3, the partial Lagrange function is written as
\[
\hat{L} = \sum_{k=1}^{K} \beta_k \left[ \frac{t_k}{h_k^2} f\left(\frac{\ell_k}{t_k}\right) + (R_k - \ell_k)C_kP_k \right] + \lambda \left( \frac{\sum_{k=1}^{K} C_k\ell_k}{F} + \sum_{k=1}^{K} t_k - T \right).
\]

Define \( a_k = \frac{F\ln 2}{BC_k} \) and \( b_k = \frac{FP_kh_k^2}{N_0} \). Using KKT conditions, we can obtain the following offloading priority function
\[
\hat{\varphi}(\beta_k, C_k, P_k, h_k, F) = \begin{cases} 
\frac{\beta_k N_0}{h_k^2} (\hat{\nu}_k \ln \hat{\nu}_k - \hat{\nu}_k + 1), & \hat{\nu}_k \geq 1 \\
0, & \hat{\nu}_k < 1
\end{cases},
\] (11)

where
\[
\hat{\nu}_k = \frac{b_k - 1}{W_0((b_k - 1)e^{(a_k - 1)})}.
\] (12)

This function is derived by solving an equation in the following lemma, proved in Appendix F.

**Lemma 9.** Given \( \hat{\nu}_k \geq 1 \), the offloading priority function \( \hat{\varphi}_k = \hat{\varphi}(\beta_k, C_k, P_k, h_k, F) \) in (11) is the root of the following equation with respect to \( x \):
\[
f'^{-1}\left( C_kP_kh_k^2 - \frac{x C_kh_k^2}{\beta_k} \right) = g^{-1}\left( \frac{-h_k^2x}{\beta_k} \right).
\] (13)

Recall that for a cloud that upper-bounds the offloaded computation, the offloading priority \( \hat{\varphi}_k \) in (9) is function of a Lagrange multiplier \( \bar{\mu}^* \), determined by \( F \). However, for the cloud with non-negligible computing time, the offloading priority function \( \hat{\varphi}_k \) in (11) is directly affected
by the finite cloud capacity $F$ via $\tilde{\nu}_k$. In the following, the properties of $\tilde{\nu}_k$, which is the key component of $\tilde{\varphi}_k$, are characterized.

**Lemma 10.** $\tilde{\nu} > 1$ if and only if $\nu > 1$, where $\nu$ is defined in (8).

It is proved in Appendix G and indicates that the condition that offloading saves energy for this kind of finite-capacity cloud is same as that of infinite-capacity cloud.

**Lemma 11.** Given $\tilde{\nu} \geq 1$, $\tilde{\varphi}(\beta, C, P, h, F)$ is a monotone increasing function for $\beta$, $C$, $P$, $h$ and $F$, respectively.

Lemma 11 can be easily proved by deriving the first derivatives of $\tilde{\varphi}$ with respect to each parameter. It shows that enhancing the cloud capacity will increase the offloading priority for all users that is same as the result of a cloud with upper-bounded computation shown in Lemma 6. However, this may change the offloading priority order due to their varying increasing rates.

Based on above discussion, the main result of this section are presented in the following theorem.

**Theorem 3.** Consider a finite-capacity cloud with non-negligible computing time. The optimal resource allocation policy solving Problem P3 has the same structure as that in Theorem 1 and is expressed in terms of the priority function $\tilde{\varphi}_k$ in (11) and optimized Lagrange multipliers $\hat{\lambda}^*$.

The optimal policy can be computed with a one-dimension search for $\hat{\lambda}^*$, following a similar procedure in Algorithm 1. The key difference is that $\hat{\lambda}^*$ in this case also depends on the cloud capacity due to the time-sharing constraint in Problem P3, which is given in the lemma below.

**Lemma 12.** A cloud with smaller capacity $F$ leads to a larger Lagrange multiplier $\hat{\lambda}^*$.

VI. **MULTIUSER MECO FOR OFDMA**

In this section, considering a cloud with infinite computation capacity based on OFDMA, both the OFDM sub-channels and offloaded data sizes are optimized for energy-efficient multi-user MECO. To solve the formulated non-convex mixed-integer optimization problem, a low-complexity sub-optimal algorithm is proposed by defining an average offloading priority function from its TDMA counterpart and shown to have close-to-optimal performance in simulation.

A. **Multiuser MECO for OFDMA: Infinite Cloud Capacity**

Consider an OFDMA system described in Section II with $K$ mobiles and $N$ sub-channels. Given time-slot duration $T$, the latency constraint for local computing is rewritten as $C_k(R_k -$
\[ \sum_{n=1}^{N} \frac{\ell_{k,n}}{F_k} \leq T. \] Moreover, the time-sharing constraint is replaced by sub-channel constraints, expressed as \( \sum_{k=1}^{K} \rho_{k,n} \leq 1 \) for all \( n \). Then the corresponding optimization problem for the minimum weighted sum mobile energy consumption based on OFDMA is readily reformulated as follows.

\[
\min_{\{\ell_{k,n}, \rho_{k,n}\}} \sum_{k=1}^{K} \sum_{n=1}^{N} \beta_k \left[ \sum_{n=1}^{N} \frac{\rho_{k,n} f\left(\frac{\ell_{k,n}}{\rho_{k,n}}\right)}{h_{k,n}} \left( R_k - \sum_{n=1}^{N} \ell_{k,n} \right) C_k P_k \right] \\
\text{s.t.} \quad \sum_{k=1}^{K} \rho_{k,n} \leq 1, \quad n \in \mathcal{N}, \\
\quad m_k^+ \leq \sum_{n=1}^{N} \ell_{k,n} \leq R_k, \quad k \in \mathcal{K}, \\
\quad \rho_{k,n} \in \{0, 1\}, \quad n \in \mathcal{N} \text{ and } k \in \mathcal{K}.
\]

(\text{P4})

Observe that Problem P4 is a non-convex mixed-integer programming problem that is difficult to solve. It involves the joint optimization of both continuous variables \( \{\ell_{k,n}\} \) and integer variables \( \{\rho_{k,n}\} \). One common solution method is convex relaxation by allowing time sharing for each sub-channel as in [19] and then applying convex optimization techniques. An alternative method is using dual decomposition as in [25], which has been proved to be optimal when the number of sub-channels goes to infinity. However, both algorithms performing extensive iterations shed little insight on the policy structure.

To reduce the computation complexity and characterize the policy structure, a low-complexity sub-optimal algorithm is proposed below by decomposition method. Specifically, the initial resource and offloaded data allocation is determined by defining an average offloading priority function. Then, the integer sub-channel assignment is performed according to the offloading priority order, followed by the adjustment of offloaded data allocation over assigned sub-channels for each user. The main procedures of this sequential algorithm are summarized as follows.

- **Phase 1** [Sub-Channel Reservation for Offloading-Required Users]: Consider the offloading-required users that have \( m_k^+ > 0 \). The offloading priorities for these users are ordered in the descending manner. Based on this, the available sub-channels with high priorities are assigned to users sequentially and each user is allocated with one sub-channel.

- **Phase 2** [Initial Resource and Offloaded Data Allocation]: For the unassigned sub-channels, using average channel gains over these sub-channels for all users, the OFDMA resource allocation problem is transformed into its TDMA counterpart. Then, by defining an average offloading priority function, the optimal total sub-channel number and offloaded data size for each user are derived. Note that the resultant sub-channel numbers may not be integer.
Sub-channel Reservation for Offloading-Required Users.

While $K_1 \neq \emptyset$, reserve sub-channels as follows.

1. Let $\rho_{k',n'} = 1$ where $\{k', n'\} = \arg \max_{k \in K_1, n \in N_2} \varphi_{k,n}$.
2. Update sets: $S_{k'} = S_{k'} \cup \{n'\}$; $K_1 = K_1 \setminus \{k'\}$; $N_1 = N_1 \cup \{n'\}$; $N_2 = N \setminus N_1$.

Phase 3 [Integer Sub-channel Assignment]: Given constraints on the rounded total sub-channel numbers for each user derived in Phase 2, specific integer sub-channel assignment is determined by the offloading priority order. Specifically, each sub-channel is assigned to the user that requires sub-channel assignment and has higher offloading priority than others.

Phase 4 [Adjustment of Offloaded Data Allocation]: For each user, given the sub-channel assignment in Phase 3, the specific offloaded data allocation is optimized by a water-filling type algorithm.

Before stating the algorithm, let $\varphi_{k,n}$ define the offloading priority function for user $k$ at sub-channel $n$, given by

$$\varphi_{k,n} = \begin{cases} \frac{\beta_k \bar{N}_0}{h_{k,n}^2} (v_{k,n} \ln v_{k,n} - v_{k,n} + 1), & v_{k,n} \geq 1 \\ 0, & v_{k,n} < 1 \end{cases},$$

where $v_{k,n} = \frac{\bar{B}T C_k P_k h_{k,n}^2}{\bar{N}_0 \ln 2}$. Let $\Phi$ reflect the offloading priority order, which is constituted by $\{\varphi_{k,n}\}$ arranged in the descending manner, e.g., $\{\varphi_{2,3} \geq \varphi_{1,4} \geq \cdots \geq \varphi_{5,2}\}$. The set of offloading-required users is denoted as $K_1$, given by $K_1 = \{k, |m_k^+ > 0\}$. The sets of assigned and unassigned sub-channels are denoted by $N_1$ and $N_2$, initialized as $N_1 = \emptyset$ and $N_2 = N$. For each user, say user $k$, the assigned sub-channel set is represented by $S_k$, initialized as $S_k = \emptyset$.

In addition, the sub-channel assignment indicators are set as $\{\rho_{k,n} = 0\}$ at the beginning.

Using these definitions, the detailed control policies are elaborated as follows.

1) Sub-Channel Reservation for Offloading-Required Users: The purpose of this phase is to guarantee that the computation latency constraints are satisfied for all users. This can be achieved by reserving one sub-channel for each offloading-required user as presented in Algorithm 4.

Observe that Step 1 in the loop searches for the highest offloading priority $\varphi_{k',n'}$ over unassigned sub-channels $N_2$ for the remaining offloading-required users $K_1$; and then allocates sub-channel $n'$ to user $k'$. This sequential sub-channel assignment follows the descending offloading priority order. Moreover, the condition for the loop ensures that all offloading-required users will be allocated with one sub-channel. This phase only has a complexity of $O(K)$ since it just
Algorithm 5 Integer Sub-channel Assignment.

Step 1: While $\tilde{\mathcal{K}} \neq \emptyset$, assign sub-channels as follows.

1. Let $\rho_{k',n'} = 1$ where \( \{k',n'\} = \arg\max_{k \in \tilde{\mathcal{K}}, n \in \mathcal{N}} \varphi_{k,n} \).
2. Update sets: $\mathcal{S}_{k'} = \mathcal{S}_{k'} \cup \{n'\}$; $\mathcal{N}_1 = \mathcal{N}_1 \cup \{n'\}$; $\mathcal{N}_2 = \mathcal{N} \setminus \mathcal{N}_1$.
3. If $|\mathcal{S}_{k'}| = \tilde{n}_{k'}$, then $\tilde{\mathcal{K}} = \tilde{\mathcal{K}} \setminus \{k'\}$.

Step 2: If $\mathcal{N}_2 \neq \emptyset$, assign remaining sub-channels as follows. For each $n \in \mathcal{N}_2$, let $\rho_{k',n} = 1$ where $k' = \arg\max_{k \in \mathcal{K}} \varphi_{k,n}$.

performs the max operation for at most $K$ iterations.

2) Initial Resource and Offloaded Data Allocation: This phase determines the total allocated sub-channel number and offloaded data size for each user. Note that the integer constraint on sub-channel allocation makes Problem P4 challenging, which requires an exhaustive search. To reduce the computation complexity, we first derive the non-integer total number of sub-channels for each user as below.

Using a similar method in [26], for each user, say user $k$, let $H_k$ denote its average sub-channel gain, given by $H_k = \sqrt{\frac{\sum_{n \in \mathcal{N}_2} \bar{h}_{k,n}^2}{|\mathcal{N}_2|}}$ where $|\mathcal{N}_2|$ gives the cardinality of unassigned sub-channel set $\mathcal{N}_2$ resulted from Phase 1. Then, the MECO OFDMA resource allocation Problem P4 is transformed into its TDMA counterpart Problem P5 as below.

\[
\min_{\{\ell_k, n_k\}} \sum_{k=1}^{K} \beta_k \left[ \frac{n_k \ell_k}{H_k^2} \bar{f} \left( \frac{\ell_k}{n_k} \right) + (R_k - \ell_k)C_k P_k \right]
\]

\[
s.t. \quad \sum_{k=1}^{K} n_k \leq |\mathcal{N}_2|,
\]

\[
n_k \geq 0, \quad m_k^+ \leq \ell_k \leq R_k, \quad k \in \mathcal{K}
\]

where $\{\ell_k, n_k\}$ are the allocated total sub-channel numbers and offloaded data sizes.

Define an average offloading priority function as in (7) by replacing $h_k$ as $H_k$. The optimal control policy $\{\ell_k^*, n_k^*\}$ can be directly obtained following the same method as for Theorem 1. Note that this phase only invokes the bisection search, for which the computation complexity is independent of $K$ and $N$.

3) Integer Sub-channel Assignment: Given the non-integer total sub-channel number allocation obtained in Phase 2, in this phase, users are assigned with specific integer sub-channels based on offloading priority order. Specifically, it includes the following two steps as in Algorithm 5.

In the first step, to guarantee that sub-channels are enough for allocation, each user is allocated
with $\tilde{n}_k^* = \lfloor n_k^* \rfloor$ sub-channels. However, allocating specific sub-channels to users given the rounded numbers is still hard, for which the optimal solution can be obtained using the Hungarian Algorithm [27] that has the complexity of $O(N^3)$. To further reduce the complexity, a priority-based sub-channel assignment is proposed as follows. Let $\tilde{K}$ denote the set of users that require sub-channel assignment, which is initialized as $\tilde{K} = \{k, |\tilde{n}_k^* > 0\}$ and will be updated as in Step 1.(3), by deleting the user that has been allocated with the maximum sub-channels. During the loop, for users in set $\tilde{K}$ and available sub-channels $N_2$, we search for the highest offloading priority function, indexed as $\varphi_{k',n'}$, and assign sub-channel $n'$ to user $k'$.

In the second step, all users compete for remaining sub-channels since $\tilde{n}_k^*$ is the lower-rounding of $n_k^*$ in the first step. In particular, each unassigned sub-channel in $N_2$ is assigned to the user with highest offloading priority. In total, the computation complexity of this phase is $O(N)$.

4) Adjustment of Offloaded Data Allocation: Based on results from Phase 1–3, for each user, say $k$, this phase allocates the total offloaded data $\ell_k^*$ over assigned sub-channels $S_k$ for minimizing the individual mobile energy consumption. The corresponding optimization problem is formulated as below with the solution given in Proposition 3.

$$\begin{align*}
\min_{\ell_{k,n}} \sum_{n \in S_k} \frac{1}{h_{k,n}^2} f(\ell_{k,n}) \\
\text{s.t.} \sum_{n \in S_k} \ell_{k,n} = \ell_k^*, \quad \ell_{k,n} \geq 0, \quad n \in S_k.
\end{align*}$$

(\textbf{P6})

**Proposition 3.** For user $k$, the optimal offloaded data allocation solving Problem P6 is

$$\ell_{k,n}^* = \left[ BT \log_2 \left( \frac{\xi_k BT h_{k,n}^2}{N_0 \ln 2} \right) \right]^+$$

for $n \in S_k$ where $\xi_k$ satisfies $\sum_{n \in S_k} \ell_{k,n}^* = \ell_k^*$.

Note that it is possible that some sub-channels are allocated to user $k$ but without offloaded data allocation due to its poor sub-channel gain. Since the complexity of such water-filling type algorithm is independent of $K$ and $N$, the total complexity of this phase is $O(K)$, considering offloaded data allocation for each user.

**Remark 7 (Low-Complexity Algorithm).** Based on above discussion, the total complexity for the proposed sequential sub-optimal algorithm is $O(K + N)$, which significantly reduces the computation complexity compared with that of optimal policies (e.g., using dual decomposition has complexity of $O(K^2)$ [28] for convergence).
B. Multiuser MECO for OFDMA: Finite Cloud Capacity

For the case of finite-capacity cloud based on OFDMA, the corresponding sub-optimal low-complexity algorithm can be derived by modifying that for infinite-capacity cloud as follows.

Recall that, for TDMA MECO, modifying the offloading priority function of infinite-capacity cloud leads to the optimal resource allocation for the finite-capacity cloud. Therefore, by the similar method, modifying Phase 2 to account for the finite computation capacity will give the new optimal non-integer sub-channel numbers and total offloaded data sizes for all users. Other phases in Section VI-A can be straightforwardly extended to the current case and are omitted for simplicity.

VII. Simulation Results

In this section, the performance of the proposed resource-allocation algorithms for both the TDMA and OFDMA systems is evaluated by simulation. The simulation settings are as follows unless specified otherwise. There are 30 users in the system with equal fairness factors, i.e., $\beta_k = 1$ for all $k$ such that the weighted sum mobile energy consumption represents the total mobile energy consumption. The time slot $T = 100$ ms. Both channels $h_k$ in TDMA and sub-channels $h_{k,n}$ in OFDAM are modeled as independent Rayleigh fading with average power loss set as $10^{-3}$. The variance of complex white Gaussian channel noise $N_0 = 10^{-9}$ W. Consider mobile $k$. The CPU computation capacity $F_k$ is uniformly selected from the set $\{0.1, 0.2, \cdots, 1.0\}$ GHz and the local computing energy per cycle $P_k$ follows a uniform distribution in the range $(0, 20 \times 10^{-11})$ J/cycle similar to [13]. For the computing task, both the data size and required number of CPU cycles per bit follow the uniform distribution with $R_k \in [100, 500]$ KB and $C_k \in [500, 1500]$ cycles/bit. All random variables are independent for different mobiles, modeling heterogeneous
A. Multiuser MECO for TDMA

Consider a MECO system where the bandwidth $B = 10$ MHz. For performance comparison, a baseline equal resource-allocation policy is considered, which allocates equal offloading time duration for mobiles that satisfy $\nu_k > 1$ and based on this, the offloaded data sizes are optimized.

Fig. 3 shows the curves of total mobile energy consumption versus the time slot duration $T$. Several observations can be made. First, the total mobile energy consumption reduces as the time-slot duration grows. Next, the sub-optimal policy computed using Algorithm 3 is found to have close-to-optimal performance and yields total mobile energy consumption less than half of that for the equal resource-allocation policy. The energy reduction is more significant for a shorter time slot duration. The reason is that without the optimization on time fractions, the offloading energy of baseline policy grows exponentially as the allocated time fractions decreases.

The curves of total mobile energy consumption versus the cloud computation capacity are displayed in Fig. 4(a). It can be observed that the performance of the sub-optimal policy approaches to that of the optimal one when the cloud computation capacity increases and achieves substantial energy savings gains over the equal resource-allocation policy. Furthermore, the total mobile energy consumption is invariant after the cloud computation capacity exceeds

\footnote{The performance of finite-capacity cloud with non-negligible computing time has similar observations and is omitted due to limited space.}
some threshold (about $6 \times 10^9$). This suggests that there exists some critical value for the cloud computation capacity, above which increasing the capacity yields no reduction on the total mobile energy consumption.

Last, Fig. 4(b) plots the curves of total energy consumption versus the number of mobiles. It shows the total energy consumption of the proposed policy grows with the number of mobiles at a much slower rate than that of the equal-allocation policy. Again, the designed sub-optimal policy is observed to have close-to-optimality.

B. Multiuser MECO for OFDMA

Consider the clouds with infinite computation capacity for OFDMA MECO systems where $B = 1$ MHz and $N_0 = 10^{-9}$ W for each sub-channel. The proposed low-complexity sub-optimal resource allocation policy is compared with two baseline policies. One is the time-sharing resource-allocation policy that is the lower bound of minimum total mobile energy consumption and computed by a convex problem solver: CVX [29]. The other is the greedy resource-allocation policy which allocates each sub-channel to the user with highest offloading priority without considering the heterogeneous computation loads, followed by the optimal data allocation over assigned sub-channels.

Fig. 5(a) depicts the curves of total mobile energy consumption versus the number of sub-channels in a OFDMA MECO system with 8 users. It can be observed that the proposed sub-optimal resource allocation have close-to-optimal performance as the time-sharing policy, especially when the number of sub-channels is large (e.g., 256). Moreover, the proposed policy has significant energy-savings gain over the greedy policy. The reason is that it considers the
varying computation loads over users and allocates more sub-channels for heavy-loaded users, while the greedy policy only offloads computation from users with high priorities. In addition, it suggests that increasing the number of sub-channels has little effect on the energy savings for the system if the number is above a threshold (about 64), but otherwise it decreases the total mobile energy consumption significantly.

Fig. 5(b) shows the curves of total mobile energy consumption versus the number of users for a OFDMA system with 128 sub-channels. It is observed that the total mobile energy consumptions for three policies increase with more users in the same trend that is almost linear. However, the proposed policy has much smaller increasing rate than the greedy one and approaches the performance of the time-sharing policy.

VIII. CONCLUSION

This work studies resource allocation for a multiuser MECO system based on TDMA/OFDMA, accounting for both the cases of infinite and finite cloud computation capacities. For the TDMA MECO system, it shows that, to achieve the minimum weighted sum mobile energy consumption, the optimal resource allocation policy should have a threshold-based structure. Specifically, we derive an offloading priority function that depends on the local computing energy and channel gains. Based on this, the BS makes a binary offloading decision for each mobile, where users with priorities above and below a given threshold will perform complete and minimum offloading. Furthermore, a simple sub-optimal algorithm is proposed to reduce the complexity for computing the threshold for finite-capacity cloud. Then, we extend this threshold-based policy structure to the OFDMA system and design a low-complexity algorithm to solve the formulated non-convex mixed-integer optimization problem, which has close-to-optimal performance in simulation.

There are several potential directions for extending the current work. First, the single time slot can be extended to case of multiple time slots, where the computation load distribution over these slots can be integrated with offloading policies in each slot studied in the current work. Next, considering the queue delay in the cloud, the joint user scheduling and resource allocation will achieve the energy savings and satisfy the latency requirements simultaneously. Last, cooperative computing among edge clouds is expected to further improve the network performance.

APPENDIX

A. Proof of Lemma 1

Since $f(x)$ is a convex function, its perspective function [28], i.e., $t_k f\left(\frac{x}{t_k}\right)$, is still convex. Using the same technique in [18], jointly considering the cases for $t_k = 0$ and $t_k > 0$, $f(x)$ is
still convex. Thus, the objective function, the summation of a set of convex functions, preserves
the convexity. Combining it with the linear convex constraints leads to the desired result.

B. Proof of Lemma 2

Whether Problem P1 is feasible depends on the following two key constraints: \( \sum_{k=1}^{K} C_k \ell_k \leq F \) and \( m_k^+ \leq \ell_k \leq R_k \). Assume \( m_k^+ \leq \ell_k \leq R_k \) is satisfied. Then it has \( \sum_{k=1}^{K} C_k m_k^+ \leq \sum_{k=1}^{K} C_k \ell_k \leq \sum_{k=1}^{K} C_k R_k \). Thus, if and only if \( \sum_{k=1}^{K} C_k m_k^+ \leq F \), there exists feasible solution such that Problem P1 is feasible.

C. Proof of Lemma 3

First, we derive a general result that is the root of equation: \( f'(p) = g^{-1}(y) \) with respect to
\( y \) as follows. According to the definitions of \( f(x) \) and \( g(x) \), it has
\[
\begin{align*}
f'(x) &= \frac{N_0 \ln 2}{B - 2^x} \quad \text{and} \quad f'^{-1}(y) = B \log_2 \left( \frac{B y}{N_0 \ln 2} \right). 
\end{align*}
\]
Therefore, the solution for the general equation is
\[
y = g(f'^{-1}(p)) = f(f'^{-1}(p)) - f'^{-1}(p) \times f(f'^{-1}(p))
= f(f'^{-1}(p)) - f'^{-1}(p) \times p
= \frac{Bp}{\ln 2} - N_0 - pB \log_2 \left( \frac{B p}{N_0 \ln 2} \right).
\]
Note that to ensure \( \ell_k^{*(2)} \geq 0 \) in Problem P1, it requires \( f'^{-1}(C_k P_k h_k^2) \geq 0 \) from (6). Combining this with (15), it leads to \( v_k \geq 1 \) where \( v_k \) is defined in (8). Then, substituting \( p = C_k P_k h_k^2 \) and
\( y = -\frac{h_k^2}{\beta_k} \) to (16) and making arithmetic operations gives the desired result as in (7).

D. Proof of Theorem 1

First, to prove this theorem, we need the following two lemmas which can be easily proved
using the definition of Lambert function and its property (omitted due to limited space).

Lemma 13. The function \( g^{-1}(y) \) can be expressed as \( g^{-1}(y) = \frac{B \left[ W_0 \left( \frac{y+N_0}{N_0 e} \right) + 1 \right]}{\ln 2} \).

Lemma 14. The function \( g^{-1}(y) \) is a monotone decreasing function for \( y < 0 \).

Then, consider case 1) in Theorem 1. Note that for mobile \( k \), if \( m_k^+ = 0 \) and \( v_k \leq 1 \), it results
in \( \ell_k^{(2)} = 0 \) derived from (5a). Thus, if these two conditions are satisfied for all \( k \), it leads
to \( \ell_k^{(2)} = \ell_k^{(2)} = 0 \).
Furthermore, if \( \psi_k > 1 \) or \( m_k^+ > 0 \), it leads to \( t_k^{*(2)} > 0 \). And the time-sharing constraint should be active since remaining time can be used for extending offloading duration so as to reduce transmission energy. Moreover, consider each user, say user \( k \). If \( \psi_k > 1 \), then from (5a) and (5b), \( \{t_k^{*(2)}, \ell_k^{*(2)}\} \) should satisfy the following:

\[
\ell_k^{*(2)} = \min \left\{ \max \left[ \frac{m_k^+}{t_k^{*(2)}}, f^{-1}(C_k P_k h_k^2) \right], \frac{R_k}{t_k^{*(2)}} \right\}
\]

(17a)

\[
= \max \left\{ \frac{m_k^+}{t_k^{*(2)}}, \min \left[ f^{-1}(C_k P_k h_k^2), \frac{R_k}{t_k^{*(2)}} \right] \right\}
\]

(17b)

\[
= g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right).
\]

(17c)

Using Lemma 3 and Lemma 14, we have the following:

1) If \( \varphi_k > \lambda^* \geq 0 \), it has \( -h_k^2 \varphi_k < -h_k^2 \lambda^* \leq 0 \). Then, from (17a), it gives

\[
\max \left[ \frac{m_k^+}{t_k^{*(2)}}, f^{-1}(C_k P_k h_k^2) \right] \geq f^{-1}(C_k P_k h_k^2) = g^{-1} \left( \frac{-h_k^2 \varphi_k}{\beta_k} \right) > g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right).
\]

From (17a), (17c) and (18), it follows that \( t_k^{*(2)} = R_k \).

2) If \( \varphi_k = \lambda^* \), it has \( f^{-1}(C_k P_k h_k^2) = g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right) \).

3) If \( 0 \leq \varphi_k < \lambda^* \), it has \( -h_k^2 \varphi_k > -h_k^2 \lambda^* \). Combining it with (17b) leads to

\[
\min \left[ f^{-1}(C_k P_k h_k^2), \frac{R_k}{t_k^{*(2)}} \right] \leq f^{-1}(C_k P_k h_k^2) = g^{-1} \left( \frac{-h_k^2 \varphi_k}{\beta_k} \right) < g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right).
\]

From (17b), (17c) and (19), it follows that \( t_k^{*(2)} = m_k^+ \).

Furthermore, if \( \psi_k < 1 \), it has \( t_k^{*(2)} = m_k^+ \). Note that this case can be included in the scenario of \( \varphi_k < \lambda^* \) with the definition of \( \varphi_k \) in (7).

Last, from (17c), it follows that

\[
t_k^{*(2)} = \frac{\ell_k^{*(2)}}{g^{-1} \left( \frac{-h_k^2 \lambda^*}{\beta_k} \right)} = \frac{\ell_k^{*(2)} \ln 2}{B W_0 \left( \frac{\lambda^* h_k^2 / \beta_k - N_0}{N_0 e} \right) + 1}
\]

(20)

where (20) is obtained using Lemma 13, completing the proof.

\[\blacksquare\]

E. Proof of Lemma 8

Assume the constraints \( m_k^+ \leq \ell_k \leq R_k \) and \( t_k \geq 0 \) are satisfied. Then it has the following:

\[
\sum_{k=1}^{K} \frac{C_k m_k^+}{F} \leq \sum_{k=1}^{K} \frac{C_k \ell_k}{F} + \sum_{k=1}^{K} t_k.
\]

Therefore, to ensure there exists at least one feasible solution that also satisfies the latency constraint (10), the necessary and sufficient condition is \( \sum_{k=1}^{K} C_k m_k^+ / F \leq T \). However, the equality
cannot be achieved. The reason is that if \( \sum_{k=1}^{K} C_k m_k^+ = T \), it results in \( t_k = 0 \) for all \( k \) such that \( \ell_k = 0 \) for all \( k \), which contradicts with \( \frac{\sum_{k=1}^{K} C_k \ell_k}{F} \geq \sum_{k=1}^{K} C_k m_k^+ = T \).

\[ \blacksquare \]

**F. Proof of Lemma 9**

First, by arithmetic operations with the Lambert function, it can be proved that the solution for a general equation \( x \ln x + px = q \) is \( x = \frac{q}{W_0(q \times e^p)} \).

Next, to solve equation (13), let \( y_k = C_k P_k h_k^2 - \frac{x C_k h_k^2}{\beta_k F} \) and use the derivation method in Lemma 3, it has

\[
\frac{F y_k}{C_k} - \frac{F P_k h_k^2}{\beta_k F} = \frac{B y_k}{\ln 2} - N_0 - y_k B \log_2 \left( \frac{B y_k}{N_0 \ln 2} \right). \quad (21)
\]

Defining \( z_k = \frac{B y_k}{N_0 \ln 2} \), (21) can be rewritten as

\[
z_k \ln z_k + (a_k - 1) z_k = b_k - 1, \quad (22)
\]

where \( a_k \) and \( b_k \) are defined in Lemma 9. Using Lambert function, the solution for (22) can be obtained:

\[
z_k = \frac{B y_k}{N_0 \ln 2} = \frac{B y_k}{N_0 \ln 2} = \frac{B y_k}{N_0 \ln 2} = \frac{B y_k}{N_0 \ln 2} = \frac{B y_k}{N_0 \ln 2} \quad (24)
\]

Using the monotone increasing property of Lambert function, (24) is equivalent to \( b \geq a \). \( \blacksquare \)

**G. Proof of Lemma 10**

It is equivalent to proved as below that when \( \hat{\nu} > 1 \), it has \( b \geq a \).

According to the definition of Lambert function, it has \( b - 1 = W_0((b - 1)e^{(b-1)}) \). Then, it leads to

\[
\hat{\nu} = \frac{b - 1}{W_0((b - 1)e^{(b-1)})} = \frac{W_0((b - 1)e^{(b-1)})}{W_0((b - 1)e^{(b-1)})} \geq 1. \quad (24)
\]

Using the monotone increasing property of Lambert function, (24) is equivalent to \( b \geq a \). \( \blacksquare \)

**References**

[1] M. Swan, “Sensor mania! The internet of things, wearable computing, objective metrics, and the quantified self 2.0,” *J. Sens. Actuator Networks*, vol. 1, pp. 217–253, 2012.

[2] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, *et al.*, “Mobile-edge computing introductory technical white paper,” White Paper, Mobile-edge Computing (MEC) industry initiative, 2014.

[3] K. Kumar and Y.-H. Lu, “Cloud computing for mobile users: Can offloading computation save energy?,” *IEEE Computer*, no. 4, pp. 51–56, 2010.
[4] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, “MAUI: Making smartphones last longer with code offload,” in *Proc. ACM MobiSys*, pp. 49–62, Jun. 2010.

[5] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, “ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading,” in *Proc. IEEE INFOCOM*, pp. 945–953, 2012.

[6] Z. Xiao, W. Song, and Q. Chen, “Dynamic resource allocation using virtual machines for cloud computing environment,” *IEEE Trans. Parallel and Distributed Systems*, vol. 24, pp. 1107–1117, Sep. 2013.

[7] H. N. Van, F. D. Tran, and J.-M. Menaud, “Performance and power management for cloud infrastructures,” in *Proc. IEEE Cloud Computing*, pp. 329–336, 2010.

[8] A. Ahmed and E. Ahmed, “A survey on mobile edge computing,” in *Proc. IEEE Intl. Conf. Intel. Sys and Cont*, 2016.

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

[10] C. You, K. Huang, and H. Chae, “Energy efficient mobile cloud computing powered by wireless energy transfer,” *IEEE J. Select. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, 2016.

[11] Y. Mao, J. Zhang, and K. B. Letaief, “Dynamic computation offloading for mobile-edge computing with energy harvesting devices,” [Online]. Available: http://arxiv.org/pdf/1605.05488v1.pdf.

[12] X. Xiang, C. Lin, and X. Chen, “Energy-efficient link selection and transmission scheduling in mobile cloud computing,” *IEEE Wireless Commun. Letters*, vol. 3, pp. 153–156, Jan. 2014.

[13] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient multi-user computation offloading for mobile-edge cloud computing,” *IEEE Trans. Networking*, vol. PP, pp. 1–1, Oct. 2015.

[14] S. Sardellitti, G. Scutari, and S. Barbarossa, “Joint optimization of radio and computational resources for multicell mobile-edge computing,” *IEEE Trans. Signal and Info. Processing over Networks*, Jun. 2015.

[15] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, “A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing,” in *Proc. IEEE Globecom*, pp. 1–6, 2015.

[16] Y. Ge, Y. Zhang, Q. Qiu, and Y.-H. Lu, “A game theoretic resource allocation for overall energy minimization in mobile cloud computing system,” in *Proc. IEEE ISLPED*, pp. 279–284, 2012.

[17] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, “A framework for cooperative resource management in mobile cloud computing,” *IEEE J. Select. Areas Commun.*, vol. 31, no. 12, pp. 2685–2700, 2013.

[18] X. Wang and G. B. Giannakis, “Power-efficient resource allocation for time-division multiple access over fading channels,” *IEEE Trans. Info. Theory*, vol. 54, pp. 1225–1240, Mar. 2008.

[19] C. Y. Wong, R. S. Cheng, K. B. Lataief, and R. D. Murch, “Multiuser OFDM with adaptive subcarrier, bit, and power allocation,” *IEEE J. Select. Areas Commun.*, vol. 17, pp. 1747–1758, Oct. 1999.

[20] S.-J. Oh, D. Zhang, and K. M. Wasserman, “Optimal resource allocation in multiservice CDMA networks,” *IEEE Trans. Wireless Commun.*, vol. 2, pp. 811–821, Jul. 2003.

[21] L. B. Le and E. Hossain, “Resource allocation for spectrum underlay in cognitive radio networks,” *IEEE Trans. Wireless Commun.*, vol. 7, pp. 5306–5315, Dec. 2008.

[22] Y. Choi, H. Kim, S.-w. Han, and Y. Han, “Joint resource allocation for parallel multi-radio access in heterogeneous wireless networks,” *IEEE Trans. Wireless Commun.*, vol. 9, pp. 3324–3329, Nov. 2010.

[23] S. Borst, “User-level performance of channel-aware scheduling algorithms in wireless data networks,” *IEEE Trans. Networking*, vol. 13, no. 3, pp. 636–647, 2005.

[24] S.-C. Wang, K.-Q. Yan, W.-P. Liao, and S.-S. Wang, “Towards a load balancing in a three-level cloud computing network,” in *Proc. IEEE Intl. Conf. Comput. Sci. and Info. Tech.*, vol. 1, pp. 108–113, 2010.

[25] M. Tao, Y.-C. Liang, and F. Zhang, “Resource allocation for delay differentiated traffic in multiuser OFDM systems,” *IEEE Trans. Wireless Commun.*, vol. 7, no. 6, pp. 2190–2201, 2008.
[26] D. Kivanc, G. Li, and H. Liu, “Computationally efficient bandwidth allocation and power control for OFDMA,” *IEEE Trans. Wireless Commun.*, vol. 2, no. 6, pp. 1150–1158, 2003.

[27] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[28] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[29] M. Grant, S. Boyd, and Y. Ye, “CVX: Matlab software for disciplined convex programming,” 2008.