IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 18, 2017, accepted May 24, 2017, date of publication June 2, 2017, date of current version July 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2710056

Energy-Saving Offloading by Jointly Allocating Radio and Computational Resources for Mobile Edge Computing

PENGTAO ZHAO, HUI TIAN, (Member, IEEE), CHENG QIN, (Student Member, IEEE), AND GAOFENG NIE

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Hui Tian (tianhui@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61471060 and in part by the National Science and Technology Major Project under Grant 2017ZX03001003.

ABSTRACT Mobile edge computing (MEC) providing information technology and cloud-computing capabilities within the radio access network is an emerging technique in fifth-generation networks. MEC can extend the computational capacity of smart mobile devices (SMDs) and economize SMDs' energy consumption by migrating the computation-intensive task to the MEC server. In this paper, we consider a multi-mobile-users MEC system, where multiple SMDs ask for computation offloading to a MEC server. In order to minimize the energy consumption on SMDs, we jointly optimize the offloading selection, radio resource allocation, and computational resource allocation coordinately. We formulate the energy consumption minimization problem as a *mixed interger nonlinear programming* (MINLP) problem, which is subject to specific application latency constraints. In order to solve the problem, we propose a *reformulation-linearization-technique-based Branch-and-Bound* (RLTBB) method, which can obtain the optimal result or a suboptimal result by setting the solving accuracy. Considering the complexity of RTLBB cannot be guaranteed, we further design a *Gini coefficient-based greedy heuristic* (GCGH) to solve the MINLP problem in polynomial complexity by degrading the MINLP problem into the *convex* problem. Many simulation results demonstrate the energy saving enhancements of RLTBB and GCGH.

INDEX TERMS Mobile edge computing, computation offloading, energy minimization, branch-and-bound method, reformulation-linearization-technique, Gini coefficient.

I. INTRODUCTION

With the rapid development of the Mobile Internet, the Internet of Things (IoT) and the novel mobile applications (e.g., interactive gaming, virtual reality and natural language processing, etc [1], [2]), the mobile communications technology has explosively increased during the last decade [3]. Smart mobile devices (SMDs) gradually become the major equipments for people's daily life [4]. In addition, large number of IoT terminal equipments are applied to various vertical industries [5]. Meanwhile, the novel mobile applications typically require intensive computation and result in high energy consumption [6]–[8]. Also, the SMD has limited resources (e.g., CPU-cycle frequency, storage, energy, etc). The conflict between resource-intensive applications and resource-constrained SMDs poses a challenge for improving mobile users' QoE. In particular, the limited battery capacity becomes a major bottleneck for SMDs [9]. In order to cope with the limited computing ability and prolong the battery lifetime, Mobile Cloud Computing (MCC) provides an approach for migrating the computations to the infrastructure-based cloud server via *computation offloading* [10]. However, this migration not only increase the network load but also causes the delay fluctuation which influences the latency-sensitive application [11]. In order to increase the bandwidth and decrease the latency, energy consumption and network load for computation offloading, European Telecommunications Standards Institute (ETSI) has proposed a promising approach, *Mobile Edge Computing (MEC).*

In the MEC framework, cloud computing capabilities are provided within the Radio Access Network (RAN) in close proximity to SMDs [12]. In the computation offloading of MEC, a mobile application can be executed on the SMD (*local execution*), or on the MEC server (*edge execution*). Due to the short distance between the MEC server and SMDs, the MEC paradigm can provide low latency, high bandwidth

and computing agility in computation offloading. However, both radio and computational resources are limited in MEC. Particularly, there is limited radio frequency spectrum available in RAN. Meanwhile, compared with the infrastructure-based cloud server, the MEC server has limited computation capabilities considering the economic and scalable deployment constraints [13]. Thus, the MEC server may not be qualified to deal with all SMDs' offloading computations. It is critical to design an efficient offloading scheme. In order to prolong SMDs' battery lifetime, utilize radio and computational resources efficiently, we jointly optimize the selection of offloading SMDs, radio-and-computational resource allocation in the offloading scheme. However, to the best of our knowledge, there are few studies on this challenging work.

Motivated by the differences between MEC and traditional MCC, we dedicate to design a computation offloading mechanism for MEC. In this paper, we investigate an energy minimization problem, which is subject to specified delay constraints, in order to optimizes offloading selection, radio resource allocation and computation resource allocation jointly. We model the problem as a *Mixed Integer Nonlinear Programming* (MINLP) problem, which is NP-hard. We propose an algorithm to solve the problem with adjustable solving accuracies. Furthermore, we design a heuristic algorithm to solve the problem in polynomial complexity. The main contributions of this paper are as follows:

1) To effectively save energy consumption on SMDs, we jointly optimize the offloading selection, radio resource allocation and computational resource allocation coordinately in the energy minimization problem. To the best of our knowledge, there are few works optimizing these three aspects jointly to minimize the energy consumption in a multi-users system.
2) We propose a *Reformulation-Linearization-Technique based Branch-and-Bound method* (RLTBB) with adjustable solving accuracy to solve the energy minimization problem. We use the *Reformulation-Linearization-Technique* (RLT) relaxation technique to convert the original problem to a Mixed Boolean-convex problem. Then, we relax the boolean variables of the *Mixed Boolean-convex problem* to continues variables, thereby obtaining a convex relaxation problem of the original problem. Based on the *convex* relaxation problem, we use the *Branch-and-Bound* (BB) method to solve the original problem under the specified solving accuracy.
3) We design a *Gini coefficient based greedy heuristic* (GCGH) to reduce the solving complexity of the energy minimization problem. In this heuristic, through SMD classification and Gini coefficient calculation, we can obtain a sorted searching set. Then we greedily allocate the radio resource and computational resource among SMDs of the searching set. Thus we obtain a suboptimal solution of the problem in polynomial complexity.

The rest of the paper is organized as follows. In Section II, we review related work. In Section III, we present the system model of multi-device MEC computation offloading and formulate the energy minimization problem as an MINLP problem. In Section IV, we present both the RLTBB and the GCGH in detail. In Section V,we present the simulation results. Finally, the conclusion is drawn in Section VI,.

II. RELATED WORKS

The computation offloading has been attracting significant attention in recent years. Earlier works dedicated to migrate the computation of mobile application to an infrastructure-based cloud server (i.e., MCC). To extend the computation capability of SMD and prolong the lifetime of battery, different code offloading schemes (e.g., MAUI [14], ThinkAir [15], Phone2Cloud [16], etc) were proposed in the single-user scenario. In [17], a potential game based decentralized scheme was proposed to solve the multi-user offloading problem with multiple objectives, i.e., the energy consumption minimization and application latency minimization. The decentralized scheme can achieve a Nash equilibrium and reduce the controlling and signaling overhead. In [18], to improve the offloading performance of mobile tasks, Cao *et al.* [18] posed the optimal radio resource allocation problem for the mobile task offloading in cellular network. An adjustment method, which can adjust a feasible resource block (RB) allocation plan to a candidate optimal plan, was designed to find the optimal solution. However, due to the cloud servers are typically located in the core network, computation offloading in MCC leads to high energy consumption and fluctuant latency.

To reduce the energy consumption and latency in computation offloading, ETSI proposed MEC which can provide Information Technology (IT) and cloud-computing capabilities within the RAN in close proximity to mobile subscribers [19]. Recently, there are some works [20]–[22] on computation offloading in MEC with various objectives. Lin *et al.* [20] developed an offloading framework, named Ternary Decision Maker (TDM), which aimed to shorten response time and reduce energy consumption at the same time. A more flexible execution environment for mobile applications was adopted. On account of the comprehensive modeling and the practical simulation environment, Lin *et al.* [20] gave good contributions on computation offloading. However, [20] considered the single user scenario, and would be better to extend to the multiuser scenario. Wang *et al.* [21] incorporated dynamic voltage scaling (DVS) into computation offloading in a single-user scenario. They investigated partial computation offloading by jointly optimizing the computational speed of SMD, transmit power of SMD, and offloading ratio. An energy-optimal partial computation offloading (EPCO) algorithm was proposed to solve the nonconvex energy consumption minimization problem. Furthermore, a local optimal algorithm was proposed to handle the nonconvex and nonsmooth latency minimization problem. You *et al.* [22] studied resource allocation for a multiuser mobile-edge computation offloading (MECO) system based

on time-division multiple access (TDMA) and orthogonal frequency-division multiple access (OFDMA) to minimize the mobile energy consumption. [22] gave comprehensive modeling analyses. Moreover, for the TDMA MECO system with infinite computation capacity, an optimal policy was designed. For the TDMA MECO system with finite computation capacity and the OFDMA MECO system, respectively, two sub-optimal algorithms were designed. But [21], [22] concentrated on the offloading proportion of users mainly, and ignored the joint optimization of radio and computational resources.

There are some literatures [13], [23]–[25] concentrated on the joint optimization of radio and computational resources in multiuser MEC system. Labidi *et al.* [23] jointly optimized the radio resource scheduling and computation offloading to minimize the average energy consumption on SMDs under the average delay constraints. Zhang *et al.* [24] studied the MEC in 5G heterogeneous networks and jointly optimized the offloading and radio resource allocation to minimize the system energy consumption. In [25], the radio resource[1] and computational resource allocation were jointly optimized to minimize the weighted sum energy consumption in a MIMO system. A successive convex approximation based iterative algorithm was developed. However, study [25] assumed that all SMDs must migrate computation to the MEC server simultaneously. Yu *et al.* [13] jointly selected the offloading SMDs and allocated subcarriers in an Orthogonal Frequency-Division Multiplexing Access (OFDMA) system and CPU time in the MEC server, to minimize the energy consumption on SMDs. A heuristic was designed to obtain a suboptimal result. However, this work allocated the CPU time slots non-preemptively and sorted the execution sequence on MEC server based on the energy saving. This queuing mechanism may cause an improper execution sequence and waste resources. Furthermore, there are some studies investigating to save energy by optimizing content storage and distribution in highly distributed servers. For example, Jalali *et al.* [26] studied energy consumption of nano data center (nDCs). Jalali *et al.* [26] showed that nano servers in Fog computing can complement centralized data centers (DCs) to server applications, and lead to energy saving if the applications (or part of them) are off-loadable from centralized DCs and run on nDCs.

Different from the previous works, our paper concentrates on the computation offloading for MEC by jointly optimizing the offloading, subchannel allocation and CPU-cycle frequency assignment.

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. SYSTEM MODEL

Fig. 1 shows the system model, where SMDs can offload their computation tasks to the MEC server through a cellular network. The set of SMDs can be denoted as $\mathcal{N} = \{1, 2, \ldots, N\}$.

[1]In [25], the radio resources are the transmit precoding matrices, and the computational resources are the CPU-cycle frequencies.

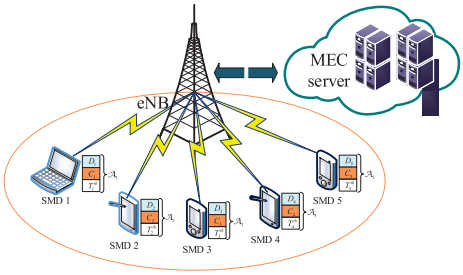


FIGURE 1. The system model.

SMD i has a computation task $\mathcal{A}_i \triangleq (D_i, C_i, T_i^{th})$, where D_i denotes the size of computation input data (e.g., the program codes and input parameters) involving in the computation task $\mathcal{A}_i$. C_i denotes the total number of CPU cycles required to accomplish the computation task $\mathcal{A}_i$, and T_i^{th} denotes the corresponding delay constraint. Each SMD can execute its task by *local execution*, or by *edge execution*. We define the offloading vector as $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]$. If SMD i executes its task by *local execution*, $\alpha_i = 0$, otherwise, $\alpha_i = 1$.

1) LOCAL EXECUTION MODEL

We define F_i^l as the maximal CPU-cycle frequency (i.e., CPU cycles per second) of SMD i, and define f_i^l as the CPU-cycle frequency to compute task $\mathcal{A}_i$. When task $\mathcal{A}_i$ is executed by *local execution* (i.e., $\alpha_i = 0$), the required time is

$$t_i^l = \frac{C_i}{f_i^l} \tag{1}$$

and corresponding energy consumption of SMD i is

$$e_i^l = \kappa \left(f_i^l\right)^2 C_i \tag{2}$$

where κ is the effective switched capacitance depending on the chip architecture [27]. We set $\kappa = 10^{-26}$ according to the practical measurement in [28]. Considering that the energy consumption grows with the allocated CPU-cycle frequency, we can minimize the energy consumption by controlling the CPU-cycle frequency with DVS technique [21]. The allocated CPU-cycle frequency is controlled as

$$f_i^l = \min\left\{\frac{C_i}{T_i^{th}}, F_i^l\right\} \tag{3}$$

2) EDGE EXECUTION MODEL

The wireless channel is constituted of L orthogonal frequency subchannels. The achievable uplink rate for SMD i in subchannel n can be obtain as

$$r_i^n = W \log\left(1 + \frac{p_i^n h_i^n}{WN_0}\right). \tag{4}$$

Where W is the bandwidth of a subchannel, p_i^n is the transmit power of SMD i in subchannel n, h_i^n is the channel gain of SMD i in subchannel n, and N_0 is the noise power spectral

density. We consider an average scenario of long term.[2] In order to concentrate more on the algorithm design, we simplify the communication model and make an assumption.

Assumption 1: We assume the subchannels to be homogeneous for each SMD (i.e., the channel gains of different subchannels are the same for a SMD, while they can be different for different SMDs). Accordingly, equal power is allocated to each assigned subchannel.

Assumption 1 is adopted in literatures [29] and [30], and is reasonable since the small-scale fading are average out and only the large-scale fading (e.g., path-loss and shadowing) affects. Accordingly, the SNR is constant with respect to frequency sine the slow fading coefficients are independent of frequency [31]. Based on Assumption 1, the achievable uplink rate for SMD i in each subchannel can be obtain as

$$r_i = W \log\left(1 + \frac{ph_i}{WN_0}\right). \tag{5}$$

Where p is the transmit power of each SMD in each assigned subchannel, h_i is the channel gain of SMD i in each subchannel. We denote the number of subchannels assigned to SMD i as θ_i . When $\alpha_i = 0, \theta_i = 0$. Otherwise, $\theta_i > 0$.[3] Accordingly, the achievable uplink rate for SMD i can be given as

$$R_i = r_i\theta_i. \tag{6}$$

We define F as the maximal CPU-cycle frequency of the MEC server, and define f_i as the assigned CPU-cycle frequency to compute task $\mathcal{A}_i$ on the MEC server [6], [25]. When task $\mathcal{A}_i$ is executed by *edge execution* (i.e., $\alpha_i = 1$), the required time of $\mathcal{A}_i$ is

$$t_i^c = \frac{D_i}{R_i} + \frac{C_i}{f_i} \tag{7}$$

and corresponding energy consumption of SMD i is

$$e_i^c = P_i^T \frac{D_i}{R_i} + P_i^I \frac{C_i}{f_i} \tag{8}$$

where P_i^T is the transmit power of SMD i, P_i^I is the power consumption in idle state. We ignore the time and energy consumption of receiving the result, because the data size of result is much smaller than the input data size [6], [13], [24].

B. PROBLEM FORMULATION

The objective of the paper is to minimize the total energy consumption on SMDs under specified latency constraints. To this end, the problem can be formulated as follows:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\theta},\boldsymbol{f}} \sum_{i=1}^{N} \alpha_i \left(P_i^T \frac{D_i}{R_i} + P_i^I \frac{C_i}{f_i}\right) + (1-\alpha_i)\kappa\left(f_i^l\right)^2 C_i \\
&s.t.\ C1: 0 \le f_i \le \alpha_i F \\
&\quad C2: \sum_{i=1}^{N} f_i \le F \\
&\quad C3: 0 \le \theta_i \le \alpha_i L \\
&\quad C4: \sum_{i=1}^{N} \theta_i \le L \\
&\quad C5: \alpha_i \left(\frac{D_i}{R_i} + \frac{C_i}{f_i}\right) + (1-\alpha_i)\frac{C_i}{f_i^l} \le T_i^{th} \\
&\quad C6: \alpha_i \in \{0, 1\} \quad \forall i \in \mathcal{N}
\end{aligned} \tag{9}$$

where $\boldsymbol{\alpha}$ is the execution indicator vector, $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_N]$ is the radio resource allocation and $\boldsymbol{f} = [f_1, f_2, \ldots, f_N]$ is the computational resource allocation. In addition, we set $\alpha_i(P_i^T \frac{D_i}{R_i} + P_i^I \frac{C_i}{f_i}) = 0$ and $\alpha_i(\frac{D_i}{R_i} + \frac{C_i}{f_i}) = 0$ when $\alpha_i = 0$ [32].
$C1$ is the constraint of the available computational resource to be allocated for user i.
$C2$ indicates that the total allocated computational resource cannot exceed F at the MEC server.
$C3$ represents the constraints of available radio resource to be allocated for user i.
$C4$ is the constraint of total radio resource in RAN.
$C5$ indicates that each task $\mathcal{A}_i$ must meet the specified latency constraint T_i^{th}.
$C6$ specifies that each SMD completes its task either by *local execution* or by *edge execution*.

Lemma 1: When $T_i^{th} \ge \frac{C_i}{F_i^l}, \forall i \in \mathcal{N}$, problem (9) has an optimal solution.

To guarantee that problem (9) has an optimal solution, we restrict $T_i^{th} \ge \frac{C_i}{F_i^l}, \forall i \in \mathcal{N}$. Due to the Knapsack problem is a NP complete problem and Problem (9)is extended from the Knapsack problem, problem (9) is NP-hard [29].

IV. PROBLEM SOLUTION

In this section, we introduce an accuracy-adjustable algorithm, RLTBB, and a suboptimal algorithm, GCGH.

A. REFORMULATION-LINEARIZATION-TECHNIQUE BASED BRANCH-AND-BOUND METHOD

1) PROBLEM REFORMULATION

To avoid the *divide-by-zero* error, we introduce two microscales, ε_1 and ε_2, to convert (9) as given by

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\theta},\boldsymbol{f}} \sum_{i=1}^{N} \left\{\alpha_i \left[\frac{P_i^T D_i}{r_i(\varepsilon_1+\theta_i)} + \frac{P_i^I C_i}{\varepsilon_2+f_i}\right] + (1-\alpha_i)\kappa\left(f_i^l\right)^2 C_i\right\} \\
&s.t.\ C1 - C4, C6 \\
&\quad C7: \alpha_i \left[\frac{D_i}{r_i(\varepsilon_1+\theta_i)} + \frac{C_i}{\varepsilon_2+f_i}\right] + (1-\alpha_i)\frac{C_i}{f_i^l} \le T_i^{th}, \\
&\qquad \forall i \in \mathcal{N}
\end{aligned} \tag{10}$$

[2]We consider that there is a long period comprised of many time slots. The channel state and resource allocation in each time slot may be different. We use the mean value of all time slots to express the average state of the long period.

[3]We assume the subchannel number is a continuous variable in the average scenario of a long term. Although there is a specific allocated subchannel number (discrete variable) in each time slot, the average allocated subchannel number in a long period can be conducted as a continuous variable.

Lemma 2: Problem (10) is sensible to ε_1 and ε_1 for obtaining a lower bound of problem (9).

Problem (10) is sensible to ε_1 and ε_1 for obtaining a lower bound of problem (9). We define two auxiliary variables, $\beta_i = (\varepsilon_1 + \theta_i)^{-1}$ and $\gamma_i = (\varepsilon_2 + f_i)^{-1}$, and reformulate (10) as follows:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \sum_{i=1}^{N}\left[\alpha_i\left(\frac{P_i^T D_i}{r_i}\beta_i + P_i^I C_i\gamma_i\right) + (1-\alpha_i)\kappa\left(f_i^l\right)^2 C_i\right]\\
&s.t.\ C7: \alpha_i\left(\frac{D_i}{r_i}\beta_i + C_i\gamma_i\right) + (1-\alpha_i)\frac{C_i}{f_i^l} \le T_i^{th}\\
&\quad C8: \frac{1}{\alpha_i L + \varepsilon_1} \le \beta_i \le \frac{1}{\varepsilon_1}\\
&\quad C9: \frac{1}{\alpha_i F + \varepsilon_2} \le \gamma_i \le \frac{1}{\varepsilon_2}\\
&\quad C10: \sum_{i=1}^{N}\frac{1}{\beta_i} \le L + N\varepsilon_1\\
&\quad C11: \sum_{i=1}^{N}\frac{1}{\gamma_i} \le F + N\varepsilon_2\\
&\quad C6 \quad \forall i \in \mathcal{N}
\end{aligned} \tag{11}$$

(11) is a nonconvex problem because of the discrete variables and the second order terms in the form of $x \cdot y$. RLT can linearize the second order terms in the form of $x \cdot y$ [33], [34]. Therefore, we can get a convex relaxation problem of (9) based on RLT and the relaxation $0 \le \alpha_i \le 1$. Particularly, we adopt the RLT to linearize the objective function and constraint $C7$ in Problem (11). For the second order term $\alpha_i \cdot \beta_i$, we define $\mu_i = \alpha_i \cdot \beta_i$. α_i is bounded as $0 \le \alpha_i \le 1$ and β_i is bounded as $\frac{1}{L+\varepsilon_1} \le \beta_i \le \frac{1}{\varepsilon_1}$. We can obtain the *RLT bound-factor product constraints* for μ_i as

$$\begin{cases}
\left\{[\alpha_i - 0]\cdot\left[\beta_i - \dfrac{1}{L+\varepsilon_1}\right]\right\}_{LS} \ge 0\\
\left\{[1-\alpha_i]\cdot\left[\beta_i - \dfrac{1}{L+\varepsilon_1}\right]\right\}_{LS} \ge 0\\
\left\{[\alpha_i - 0]\cdot\left[\dfrac{1}{\varepsilon_1} - \beta_i\right]\right\}_{LS} \ge 0\\
\left\{[1-\alpha_i]\cdot\left[\dfrac{1}{\varepsilon_1} - \beta_i\right]\right\}_{LS} \ge 0
\end{cases} \quad \forall i \in \mathcal{N} \tag{12}$$

$\{\cdot\}_{LS}$ represents a linearization step under $\mu_i = \alpha_i \cdot \beta_i$. By substituting $\mu_i = \alpha_i \cdot \beta_i$, we obtain

$$\begin{cases}
\mu_i - \dfrac{1}{L+\varepsilon_1}\alpha_i \ge 0\\
\beta_i - \dfrac{1}{L+\varepsilon_1} - \mu_i + \dfrac{1}{L+\varepsilon_1}\alpha_i \ge 0\\
\dfrac{1}{\varepsilon_1}\alpha_i - \mu_i \ge 0\\
\dfrac{1}{\varepsilon_1} - \beta_i - \dfrac{1}{\varepsilon_1}\alpha_i + \mu_i \ge 0
\end{cases} \quad \forall i \in \mathcal{N} \tag{13}$$

For the second order term $\alpha_i \cdot \gamma_i$, we define $\omega_i = \alpha_i \cdot \gamma_i$. Since $0 \le \alpha_i \le 1$ and $\frac{1}{F+\varepsilon_2} \le \gamma_i \le \frac{1}{\varepsilon_2}$, the *RLT bound-factor product constraints* for ω_i are

$$\begin{cases}
\omega_i - \dfrac{1}{F+\varepsilon_2}\alpha_i \ge 0\\
\gamma_i - \dfrac{1}{F+\varepsilon_2} - \omega_i + \dfrac{1}{F+\varepsilon_2}\alpha_i \ge 0\\
\dfrac{1}{\varepsilon_2}\alpha_i - \omega_i \ge 0\\
\dfrac{1}{\varepsilon_2} - \gamma_i - \dfrac{1}{\varepsilon_2}\alpha_i + \omega_i \ge 0
\end{cases} \quad \forall i \in \mathcal{N} \tag{14}$$

After substituting μ_i and ω_i into the objective and $C7$ in (11), we obtain a convex optimization problem relaxation (15) as

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\mu},\boldsymbol{\omega}} \sum_{i=1}^{N}\left[\frac{P_i^T D_i}{r_i}\mu_i + P_i^I C_i\omega_i + (1-\alpha_i)\kappa\left(f_i^l\right)^2 C_i\right]\\
&s.t.\ C8 - C11\\
&\quad C12: 0 \le \alpha_i \le 1\\
&\quad C13: \left(\frac{D_i}{r_i}\mu_i + C_i\omega_i\right) + (1-\alpha_i)\frac{C_i}{f_i^l} \le T_i^{th}\\
&\quad C14: (13)\\
&\quad C15: (14) \quad \forall i \in \mathcal{N}
\end{aligned} \tag{15}$$

The optimal value of (15), $\underline{E}$ ($\underline{E} = +\infty$, if problem (15) is infeasible), is a *lower bound* of problem (9).

We define $\mathcal{N}_1 \triangleq \{i \mid i \in \mathcal{N}, \alpha_i = 1\}$ and $\mathcal{N}_0 \triangleq \{i \mid i \in \mathcal{N}, \alpha_i = 0\}$. Obviously, when $\boldsymbol{\alpha}$ is determined, (9) can be converted to problem (16) as

$$\begin{aligned}
&\min_{\boldsymbol{\theta},\boldsymbol{f}} \sum_{j\in\mathcal{N}_0}\kappa\left(f_j^l\right)^2 C_j + \sum_{i\in\mathcal{N}_1}\left(\frac{P_i^T D_i}{r_i\theta_i} + \frac{P_i^I C_i}{f_i}\right)\\
&s.t.\ C1 - C4, \quad \forall i \in \mathcal{N}\\
&\quad C16: \frac{D_i}{r_i\theta_i} + \frac{C_i}{f_i} \le T_i^{th}, \quad \forall i \in \mathcal{N}_1
\end{aligned} \tag{16}$$

The optimal value of (16), $\overline{E}$ ($\overline{E} = +\infty$, if (16) is infeasible), is a *upper bound* of (9).

Lemma 3: Problem (15)(16) are convex optimization problems.

Therefore, we adopt the BB[4] method based on (15), (16), which can be solved by the state-of-the-art convex optimization algorithms, to solve the (9).

2) PROCESS OF THE RLTBB

In order to implement the BB method, we build a search tree, which is generated based on the *depth-first* strategy. The root node of the tree represents problem (9). A lower bound of E^*, the optimal result of (9), is $L_1 = \underline{E}$. We define $\{\underline{\boldsymbol{\alpha}}, \underline{\boldsymbol{\beta}}, \underline{\boldsymbol{\gamma}}, \underline{\boldsymbol{\mu}}, \underline{\boldsymbol{\omega}}\}$ as the optimal solution of (15). Then we obtain an $\overline{\boldsymbol{\alpha}}$ by the method

$$\overline{\boldsymbol{\alpha}} = [\overline{\alpha}_i \mid \overline{\alpha}_i = \begin{cases} 1, & \underline{\alpha}_i > 0.5 \\ 0, & \underline{\alpha}_i \le 0.5 \end{cases} \quad \forall \underline{\alpha}_i \in \underline{\boldsymbol{\alpha}}] \tag{17}$$

[4]BB method is an algorithm design paradigm for discrete and combinatorial optimization problems [35]. The method was first proposed by A. H. Land and A. G. Doig in 1960 for discrete programming [36].

We determine $\mathcal{N}_0$ and $\mathcal{N}_1$ based on $\overline{\boldsymbol{\alpha}}$ and calculate the optimal value, $\overline{E}$, of (16). A upper bound of E^* is $U_1 = \overline{E}$. The allocation strategy corresponding to U_1 is $SU_1 = \{\overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\theta}}, \overline{\boldsymbol{f}}\}$ ($SU_1 = \emptyset$, if $U_1 = +\infty$). We define $\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\}$ as the optimal solution of (9). If $U_1 - L_1 \leq \varepsilon$, where ε is the required tolerance[5] (i.e., the solving accuracy), we can terminate the search and $\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\} = SU_1$. Otherwise, we select an unpruned leaf node with the maximal depth and the lowest lower bound for further branching.

Now we are going to branch. At this time, we assume that the branching process is the b^{th} branching. Pick the node k, with the maximal depth and the lowest lower bound, and form two problems: The first problem is

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\theta},\boldsymbol{f}} \sum_{i=1}^{N} \alpha_i \left(P_i^T \frac{D_i}{R_i} + P_i^I \frac{C_i}{f_i} \right) + (1-\alpha_i)\kappa \left(f_i^l\right)^2 C_i \\
&s.t.\ C1 - C6, \quad \forall i \in \mathcal{N} \\
&\quad C17: \{\alpha_1, \alpha_2, \ldots, \alpha_{|d(k)|}, \alpha_{|d(k)|+1}\} = \{d(k), 0\} \quad (18)
\end{aligned}$$

and the second problem is

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\boldsymbol{\theta},\boldsymbol{f}} \sum_{i=1}^{N} \alpha_i \left(P_i^T \frac{D_i}{R_i} + P_i^I \frac{C_i}{f_i} \right) + (1-\alpha_i)\kappa \left(f_i^l\right)^2 C_i \\
&s.t.\ C1 - C6, \quad \forall i \in \mathcal{N} \\
&\quad C18: \{\alpha_1, \alpha_2, \ldots, \alpha_{|d(k)|}, \alpha_{|d(k)|+1}\} = \{d(k), 1\} \quad (19)
\end{aligned}$$

Where $d(k)$ is a parameter of node k, and denotes the determined execution strategy. $|d(k)|$ indicates the element number of $d(k)$. For the root node, $k = 1$, $d(k) = \emptyset$ and $|d(k)| = 0$. Problem (18) and (19) represent the left and right child node of node k, respectively. We define $\mathcal{N}_d$ as the determined execution strategy set (i.e., each element in $\mathcal{N}_d$ equals either 0 or 1), $\mathcal{N}_{d0} \triangleq \{i \mid i \in \mathcal{N}_d, \alpha_i = 0\}$ and $\mathcal{N}_{d1} \triangleq \{i \mid i \in \mathcal{N}_d, \alpha_i = 1\}$. The corresponding convex relaxation based on the RLT technique of problem (18) and (19) are

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\dot{\boldsymbol{\theta}},\dot{\boldsymbol{f}}^c,\ddot{\boldsymbol{\beta}},\ddot{\boldsymbol{\gamma}},\ddot{\boldsymbol{\mu}},\ddot{\boldsymbol{\omega}}} \sum_{j\in\mathcal{N}_{d0}} \kappa \left(f_j^l\right)^2 C_j + \sum_{i\in\mathcal{N}_{d1}} \left(\frac{P_i^T D_i}{r_i \dot{\theta}_i} + \frac{P_i^I C_i}{\dot{f}_i^c} \right) \\
&\qquad + \sum_{g\in\mathcal{N}\setminus\mathcal{N}_d} \left[\frac{P_g^T D_g}{r_g} \ddot{\mu}_g + P_g^I C_g \ddot{\omega}_g \right. \\
&\qquad\qquad \left. + (1-\alpha_g)\kappa \left(f_g^l\right)^2 C_g \right] \\
&s.t.\ C1, C3, \quad \forall i \in \mathcal{N}_d \\
&\quad C8, C9, C12 - C15, \quad \forall i \in \mathcal{N}\setminus\mathcal{N}_d \\
&\quad C16, \quad \forall i \in \mathcal{N}_{d1} \\
&\quad C19: \sum_{i\in\mathcal{N}_{d1}} \dot{\theta}_i + \sum_{g\in\mathcal{N}\setminus\mathcal{N}_d} \left(\frac{1}{\ddot{\beta}_g} - \varepsilon_1 \right) \leq L \\
&\quad C20: \sum_{i\in\mathcal{N}_{d1}} \dot{f}_i^c + \sum_{g\in\mathcal{N}\setminus\mathcal{N}_d} \left(\frac{1}{\ddot{\gamma}_g} - \varepsilon_2 \right) \leq F \\
&\quad C21: \mathcal{N}_d = \{d(k), 0\} \quad (20)
\end{aligned}$$

[5]If $\varepsilon = 0$, RLTBB is an optimal algorithm; if $\varepsilon > 0$, RLTBB is a suboptimal algorithm.

and

$$\begin{aligned}
&\min_{\boldsymbol{\alpha},\dot{\boldsymbol{\theta}},\dot{\boldsymbol{f}}^c,\ddot{\boldsymbol{\beta}},\ddot{\boldsymbol{\gamma}},\ddot{\boldsymbol{\mu}},\ddot{\boldsymbol{\omega}}} \sum_{j\in\mathcal{N}_{d0}} \kappa \left(f_j^l\right)^2 C_j + \sum_{i\in\mathcal{N}_{d1}} \left(\frac{P_i^T D_i}{r_i \dot{\theta}_i} + \frac{P_i^I C_i}{\dot{f}_i^c} \right) \\
&\qquad + \sum_{g\in\mathcal{N}\setminus\mathcal{N}_d} \left[\frac{P_g^T D_g}{r_g} \ddot{\mu}_g + P_g^I C_g \ddot{\omega}_g \right. \\
&\qquad\qquad \left. + (1-\alpha_g)\kappa \left(f_g^l\right)^2 C_g \right] \\
&s.t.\ C1, C3, \quad \forall i \in \mathcal{N}_d \\
&\quad C8, C9, C12 - C15, \quad \forall i \in \mathcal{N}\setminus\mathcal{N}_d \\
&\quad C16, \quad \forall i \in \mathcal{N}_{d1} \\
&\quad C19, C20 \\
&\quad C22: \mathcal{N}_d = \{d(k), 1\} \quad (21)
\end{aligned}$$

problem (20) and (21) are convex problems. In order to obtain the lower bound of the left and right child node, we calculate the optimal values, $\underline{E}_{2b}$ and $\underline{E}_{2b+1}$, corresponding to convex problems (20) and (21), respectively. The corresponding execution indicators of problem (20) and (21) are $\underline{\boldsymbol{\alpha}}_{2b}$ and $\underline{\boldsymbol{\alpha}}_{2b+1}$, respectively. Then we calculate the corresponding $\overline{\boldsymbol{\alpha}}_{2b}$ and $\overline{\boldsymbol{\alpha}}_{2b+1}$ based on (17). We calculate the upper bound $\overline{E}_{2b}$ of problem (18) based on problem (16) and $\overline{\boldsymbol{\alpha}}_{2b}$, and define the corresponding solution as $\overline{SU}_{2b} = \{\overline{\boldsymbol{\alpha}}_{2b}, \overline{\boldsymbol{\theta}}_{2b}, \overline{\boldsymbol{f}}_{2b}\}$. In the same way, we calculate the upper bound $\overline{E}_{2b+1}$ of problem (19) based on problem (16) and $\overline{\boldsymbol{\alpha}}_{2b+1}$, and define the corresponding solution as $\overline{SU}_{2b+1} = \{\overline{\boldsymbol{\alpha}}_{2b+1}, \overline{\boldsymbol{\theta}}_{2b+1}, \overline{\boldsymbol{f}}_{2b+1}\}$.

Then we calculate the lower and upper bound, L_{b+1} and U_{b+1}, of E^*. L_{b+1} equals the minimal lower bound of all the unpruned leaf nodes. In the same way, U_{b+1} equals the minimal upper bound of all the unpruned leaf nodes, and SU_{b+1} equals the solution $\overline{SU}$ corresponding to U_{b+1}. We prune unnecessary leaf nodes whose lower bounds are greater than U_{b+1}. After that, we update $b = b + 1$ and continue the above process until $U_b - L_b \leq \varepsilon$. The results are $E^* = U_b$, $\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\} = SU_b$. Algorithm 1 shows the process of RLTBB algorithm.

Theorem 1: RLTBB can converge and its computation complexity is exponential.

Proof: See Appendix D. ■

B. GINI COEFFICIENT BASED GREEDY HEURISTIC

Although RLTBB can solve problem (9) with adjustable solving accuracy (i.e., optimally or approximately), RLTBB can not guarantee the time complexity [37]. In order to reduce the solving complexity, we design the GCGH to obtain a suboptimal result of (9) in polynomial complexity. The major concept is the *Gini Coefficient* which is an efficient index, between 0 and 1, for assessment on regional income inequality. The smaller is the Gini Coefficient, the more equal is the income distribution (i.e., there are more SMDs' incomes consisting the majority of total income), and vice versa. Motivated by the indicative function of Gini Coefficient, we design a index function based on the Gini Coefficient to obtain the SMD set $\mathcal{S}_o^{s1}$ where its SMDs can contribute to the majority of energy saving. Then we allocate resources

Algorithm 1 RLTBB Algorithm

Initialization:
Set required tolerance ε
Initialize $L_1 = \underline{E}$, $U_1 = \overline{E}$, $SU_1 = \{\overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\theta}}, \overline{\boldsymbol{f}}\}$, $b = 1$ and $\mathcal{L}_1^{up} = \mathcal{L}_1^{s} = \{1\}$
Calculate the total energy consumption, E_{all}^{local}, when all the SMDs execute their application locally.
while $((U_b - L_b) > \varepsilon \cdot (E_{all}^{local} - L_b))$ **do**
  Select $k = \arg\min_{i \in \mathcal{L}_b^s} \{\underline{E}_i\}$, and split the leaf node k into two subproblems
  Solve the problem (20)(21) to obtain $\underline{E}_{2b}$, $\underline{E}_{2b+1}$, $\underline{\boldsymbol{\alpha}}_{2b}$ and $\underline{\boldsymbol{\alpha}}_{2b+1}$[6]
  Calculate $\overline{\boldsymbol{\alpha}}_{2b}$ and $\overline{\boldsymbol{\alpha}}_{2b+1}$ based on method (17) to obtain $\overline{E}_{2b}$, $\overline{E}_{2b+1}$, $\overline{SU}_{2b}$ and $\overline{SU}_{2b+1}$[7]
  form $\mathcal{L}_{b+1}^{up}$ from $\mathcal{L}_b^{up}$ by removing k and adding $2b$ and $2b+1$
  $\mathcal{L}_{b+1}^{p} := \{i \mid i \in \mathcal{L}_{b+1}^{up}, \underline{E}_i > U_b\}$
  $\mathcal{L}_{b+1}^{up} := \mathcal{L}_{b+1}^{up} \backslash \mathcal{L}_{b+1}^{p}$
  $\mathcal{L}_{b+1}^{s} := \{i \mid i = \arg\max_{j \in \{k \mid k \in \mathcal{L}_{b+1}^{up}, |d(k)| \neq N\}} |d(j)|\}$
  $L_{b+1} := \min_{i \in \mathcal{L}_{b+1}^{up}} \underline{E}_i$
  $U_{b+1} := \min_{i \in \mathcal{L}_{b+1}^{up}} \overline{E}_i$, $SU_{b+1} := \overline{SU}_{\arg\min_{i \in \mathcal{L}_{b+1}^{up}} \overline{E}_i}$
  $b := b + 1$
end while
Output:
$E^* = U_b$
$\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\} = SU_b$

to SMDs belonging to $\mathcal{S}_o^{s1}$ greedily. Generally, the GCGH scheme is divided into three stages, which are stated as follows.

- **Stage 1: SMD classification.** The SMDs are classified into two types according to the *Basic Offloading Condition*.
- **Stage 2: Gini Coefficient calculation.** Sort the SMDs, which satisfy the Basic Offloading Condition, according to the *Income function*. Calculate the *Gini Coefficient* and divide the set.
- **Stage 3: Resource allocation.** Allocate the radio resource and computational resource to the SMDs belonging to the searching set, which is obtained in stage 2, by a greedy algorithm.

1) SMD CLASSIFICATION

The less $|\mathcal{N}|$ in (9) is, the easier solving (9) is. Motivated by this relation, we reduce the problem scale of (9) by a *Basic Offloading Condition*. We define the Basic Offloading Condition as

Definition 1 (Basic Offloading Condition): If the task of SMD i is executed by the edge execution*, SMD i should satisfy the following two conditions,* $\frac{D_i}{r_i L} + \frac{C_i}{F} \leq T_i^{th}$ *and* $\frac{P_i^T D_i}{r_i L} + \frac{P_i^I C_i}{F} < \kappa (f_i^l)^2 C_i$.

Theorem 2: The SMDs, which do not satisfy the Basic Offloading Condition, must execute their tasks by local execution.

Proof: See Appendix E. ■

So we classify SMDs into two types, the locally executing set $\mathcal{S}_l$ and optionally executing set $\mathcal{S}_o$, according to the Basic Offloading Condition. SMDs not satisfying the Basic Offloading Condition are classified into $\mathcal{S}_l$, the others are classified into $\mathcal{S}_o$. Considering that SMDs in $\mathcal{S}_l$ must execute their application locally, we consider $\mathcal{S}_o$ as $\mathcal{N}$ in (9) and do not deal with $\mathcal{S}_l$, thereby reducing the problem scale of (9).

2) Gini COEFFICIENT CALCULATION

We define the *Income Function* as

Definition 2 (Income Function):

$$\Phi(i) = \eta_i \left[\kappa \left(f_i^l\right)^2 C_i - \left(\frac{P_i^T D_i}{r_i L} + \frac{P_i^I C_i}{F} \right) \right]^+, \quad i \in \mathcal{S}_o \tag{22}$$

where $[x]^+ = max\{x, 0\}$. *η_i is the weight factor of SMD i's energy saving and is calculated as*

$$\eta_i = \left(\frac{T_i^{th}}{\frac{D_i}{r_i L} + \frac{C_i}{F}} \right) \left(\frac{\sum_{j \in \mathcal{S}_o} \theta_j^n}{|\mathcal{S}_o| \theta_i^n} + \frac{\sum_{j \in \mathcal{S}_o} f_j^n}{|\mathcal{S}_o| f_i^n} \right), \quad i \in \mathcal{S}_o \tag{23}$$

where $\theta_i^n = \frac{D_i}{T_i^{th} - \frac{C_i}{F}}, i \in \mathcal{S}_o$ *and* $f_i^n = \frac{C_i}{T_i^{th} - \frac{D_i}{r_i L}}$, $i \in \mathcal{S}_o$. $\left[\kappa \left(f_i^l\right)^2 C_i - \left(\frac{P_i^T D_i}{r_i L} + \frac{P_i^I C_i}{F} \right) \right]^+$ denote the offloading energy saving when all the radio resource and computational resource are allocated to SMD i. In order to take the required radio resource, required computational resource and latency of a application into consideration, we introduce the weight factor. In the weight factor, we use the dimensionless $\frac{\sum_{j \in \mathcal{S}_o} \theta_j^n}{|\mathcal{S}_o| \theta_i^n}$ express the radio resource efficiency in energy saving of SMD i. Similarly, we use the dimensionless $\frac{\sum_{j \in \mathcal{S}_o} f_j^n}{|\mathcal{S}_o| f_i^n}$ express the computational resource efficiency in energy saving of SMD i. Considering the latency related to the radio resource and computational resource, we introduce the latency ratio $\frac{T_i^{th}}{\frac{D_i}{r_i L} + \frac{C_i}{F}}$ as a multiplier in the weight factor. The higher η_i is, the higher offloading income SMD i has. And vice versa. Therefore, we design the weight factor of energy saving as (23).

We calculate all the *income* of SMDs in $\mathcal{S}_o$ according to the Income Function. We sort all the SMDs belonging to $\mathcal{S}_o$ in ascending order of the income Φ_s: $\Phi_1 \leq \Phi_2 \leq \cdots \leq \Phi_{|\mathcal{S}_o|}$. The sorted $\mathcal{S}_o$ is defined as $\mathcal{S}_o^s$. We define $W = \sum_{i=1}^{|\mathcal{S}_o|} \Phi_i$ as the sum income, and define $w_i = \frac{1}{W} \sum_{j=1}^{i} \Phi_j$, $i = 1, 2, \cdots, |\mathcal{S}_o^s|$ as the cumulative income ratio. Therefore, we calculate the Gini Coefficient G as

$$G = 1 - \frac{1}{|\mathcal{S}_o^s|} \left(1 + 2 \sum_{i=1}^{|\mathcal{S}_o^s|-1} w_i \right) \tag{24}$$

and define the partition index I as

$$I = \min \left\{ \left\lceil \frac{1}{G} \right\rceil + \left\lceil \frac{K}{|\mathcal{S}_o^s|} (|\mathcal{S}_o^s| - \left\lceil \frac{1}{G} \right\rceil) \right\rceil, |\mathcal{S}_o^s| \right\} \tag{25}$$

where

$$\begin{aligned} K &= \min\{\left\lfloor \frac{L}{\theta^n_{max}} \right\rfloor, \left\lfloor \frac{F}{f^n_{max}} \right\rfloor, |\mathcal{S}^s_o|\} \\ \theta^n_{max} &= \max\{\theta^n_i, \forall i \in \mathcal{S}^s_o\} \\ f^n_{max} &= \max\{f^n_i, \forall i \in \mathcal{S}^s_o\} \end{aligned} \tag{26}$$

$\left\lceil \frac{1}{G} \right\rceil$ can denote the number of SMDs contributing to the majority of total income. $\left\lceil \frac{K}{|\mathcal{S}^s_o|}\left(|\mathcal{S}^s_o| - \left\lceil \frac{1}{G} \right\rceil\right)\right\rceil$ is a correction term of $\left\lceil \frac{1}{G} \right\rceil$. K denotes the load capacity of resources. $\frac{K}{|\mathcal{S}^s_o|}$ is the weight factor of the gap $\left(|\mathcal{S}^s_o| - \left\lceil \frac{1}{G} \right\rceil\right)$. Then we further reduce the problem scale of (9) and obtain the searching space $\mathcal{S}^{s1}_o$, whose SMDs are sorted by offloading income, from $\mathcal{S}^s_o$ as

$$\mathcal{S}^{s1}_o \leftarrow \{S^s_{o|\mathcal{S}^s_o|}, S^s_{o(|\mathcal{S}^s_o|-1)}, \cdots, S^s_{o(|\mathcal{S}^s_o|+1-I)}\} \tag{27}$$

3) RESOURCE ALLOCATION

In this stage, we adopt a *greedy* algorithm into $\mathcal{S}^{s1}_o$ to obtain a suboptimal solution. Firstly, we initialize

$$\begin{aligned} E^s &= \sum_{i=1}^{N} \kappa\left(f^l_i\right)^2 C_i, \mathcal{N}^s_1 = \emptyset, \mathcal{N}^s_0 = \mathcal{N}, \boldsymbol{\theta}^s \\ &= \underbrace{[0, 0, \cdots, 0]}_{N} \text{ and } \boldsymbol{f}^s = \underbrace{[0, 0, \cdots, 0]}_{N}. \end{aligned}$$

Secondly, we check each SMD in $\mathcal{S}^{s1}_o$ whether to be offloaded based on problem (16). For the first SMD of $\mathcal{S}^{s1}_o$, if adding it into $\mathcal{N}^s_1$ ($\mathcal{N}^s_0 = \mathcal{N}\backslash\mathcal{N}^s_1$) has energy saving (i.e., the corresponding optimal value $\overline{E}$ of problem (16) is smaller than E^s), we add the SMD into $\mathcal{N}^s_1$, $E^s = \overline{E}$, $\boldsymbol{\theta}^s = \overline{\boldsymbol{\theta}}$ and $\boldsymbol{f}^s = \overline{\boldsymbol{f}}$ ($\{\overline{\boldsymbol{\theta}}, \overline{\boldsymbol{f}}\}$ and $\overline{E}$ are the optimal solution and optimal value of (16) based on the $\mathcal{N}^s_1$ and $\mathcal{N}^s_0$,respectively); otherwise, $\mathcal{N}^s_1$, E^s, $\boldsymbol{\theta}^s$ and $\boldsymbol{f}^s$ keep unaltered. The remained SMDs of of $\mathcal{S}^{s1}_o$ are checked in the same way as the first SMD. When all the SMDs are checked, the checking procedure is terminated. Finally, we obtain a suboptimal value of problem (9), E^s, and the corresponding suboptimal solution, $\boldsymbol{\theta}^s$, $\boldsymbol{f}^s$ and

$$\boldsymbol{\alpha}^s = [\alpha^s_i \mid \alpha^s_i = \begin{cases} 1, & i \in \mathcal{N}^s_1 \\ 0, & i \in \mathcal{N}^s_0 \end{cases} \quad \forall i \in \mathcal{N}] \tag{28}$$

The detail of the proposed GCGH scheme is illustrated in Algorithm 2

Theorem 3: The computation of GCGH scheme has polynomial complexity.

Proof: See Appendix F. ■

Algorithm 2 GCGH Algorithm

Stage 1: SMD classification
Initialize $\mathcal{S}_l = \mathcal{S}_o = \emptyset$
for $i = 1 : N$ **do**
  if $\frac{D_i}{r_i L} + \frac{C_i}{F} \le T^{th}_i$ & $\frac{P^T_i D_i}{r_i L} + \frac{P^I_i C_i}{F} < \kappa (f^l_i)^2 C_i$ **then**
    $\mathcal{S}_o = \{\mathcal{S}_o, i\}$
  else
    $\mathcal{S}_l = \{\mathcal{S}_l, i\}$
  end if
end for
Stage 2: Gini Coefficient calculation
Calculate $\Phi(i)_{|i\in\mathcal{S}_o}$ by the Income Function
Sort Φ_s in ascending order: $\Phi_1 \le \Phi_2 \le \cdots \le \Phi_{|\mathcal{S}_o|}$, $\mathcal{S}^s_o \leftarrow$ the sorted $\mathcal{S}_o$
Calculate the cumulative income rate

$$W = \sum_{i=1}^{|\mathcal{S}^s_o|} \Phi_i, \quad w_{i|i=1,2,\cdots,|\mathcal{S}^s_o|} = \frac{1}{W} \sum_{j=1}^{i} \Phi_j$$

Calculate the Gini Coefficient G and the partition index I by formulation (24)-(26)
Obtain $\mathcal{S}^{s1}_o$ from $\mathcal{S}^s_o$ based on I and formulation (27):
Stage 3: Resource allocation
Initialize $E^s = \sum_{i=1}^{N} V_i C_i$, $\mathcal{N}^s_1 = \emptyset$, $\mathcal{N}^s_0 = \mathcal{N}$, $\boldsymbol{\theta}^s = \underbrace{[0, 0, \cdots, 0]}_{N}$ and $\boldsymbol{f}^s = \underbrace{[0, 0, \cdots, 0]}_{N}$
for $i = 1 : |\mathcal{S}^{s1}_o|$ **do**
  $\mathcal{N}_1 \leftarrow \{\mathcal{N}^s_1, \mathcal{S}^s_{oi}\}$, $\mathcal{N}_0 \leftarrow \mathcal{N}\backslash\mathcal{N}_1$
  Solve problem (16) based on $\mathcal{N}_1$ and $\mathcal{N}_0$ to obtain the optimal result $\overline{E}, \overline{\boldsymbol{\theta}}, \overline{\boldsymbol{f}}$
  if $\overline{E} < E^s$ **then**
    $\mathcal{N}^s_1 = \mathcal{N}_1, \mathcal{N}^s_0 = \mathcal{N}\backslash\mathcal{N}^s_1, E^s = \overline{E}, \boldsymbol{\theta}^s = \overline{\boldsymbol{\theta}}, \boldsymbol{f}^s = \overline{\boldsymbol{f}}$
  end if
end for
Calculate the execution indicator by formulation (28)
Output:
The suboptimal result E^s, $\boldsymbol{\alpha}^s$, $\boldsymbol{\theta}^s$, $\boldsymbol{f}^s$

V. SIMULATION RESULTS

There is an orthohexagonal region, which is covered by an eNB located at the center, with 500m in diameter. SMDs are randomly scattered over the region. There is an MEC server located in the eNB, whose computation capability is $F = 5GHz/sec$. The SMD's idle power and transmission power are set to be $P^I_i = 10mWatts$ and $P^T_i = 100mWatts$ [17], respectively. And the computational capability of each SMD is $F^l_i = 0.5GHz$. We simulate the face recognition application [38]. The data size[8] of the computation offloading and total number of CPU cycles[9] are Gaussian distributions, $D_i \sim N(400, 100)$ and $C_i \sim N(1000, 100)$.

The proposed algorithms are compared with three methods. The optimal results are obtained by brute-force search. In this case, when the offloading selection is given, we use the convex optimal method to calculate the radio and computational resource allocations. "All-Local" stands for that all SMDs execute their applications locally. "Greedy Heuristic" is a greedy heuristic which loses sight of Gini Coefficient Calculation (i.e., **Stage 2**) compared with GCGH. "RLTBB-ε = 0.6" stands for the RLTBB with solving accuracy $\varepsilon = 0.6$.

[8]The data size is measured by KB
[9]The number of CPU cycles is measured by Megacycles

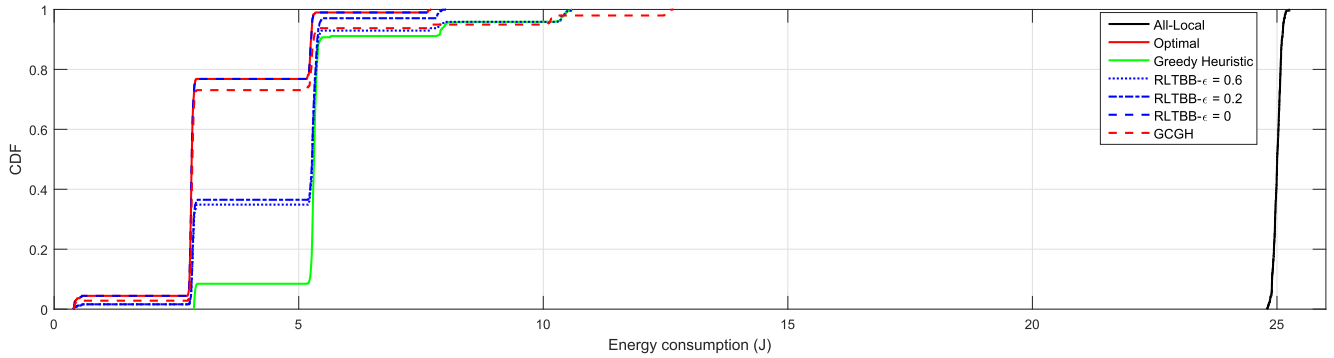


FIGURE 2. CDF of energy consumption, $|\mathcal{N}| = 10, L = 500, F = 5GHz, T_i^{th} = \frac{C_i}{F_i}, \forall i \in \mathcal{N}$.

Similarly, "RLTBB-$\varepsilon = 0.2$" and "RLTBB-$\varepsilon = 0$" stand for the RLTBBs with solving accuracies $\varepsilon = 0.2$ and $\varepsilon = 0$, respectively.

A. CDF OF THE ENERGY CONSUMPTION

CDFs of energy consumption under different algorithms are shown in Fig. 2. From the curves, RLTBB-$\varepsilon = 0$ can obtain the optimal result. RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and GCGH can not achieve the optimal result, but their results save much energy compared with All-Local. The energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic achieve 97.97%, 94.69%, 93.54% and 90.08% of the optimal energy saving, respectively. The result of GCGH is smaller than that of Greedy Heuristic. Greedy Heuristic does not have the **Stage 2** compared with GCGH. Thus, Greedy Heuristic is short of the selecting pertinence of offloading SMDs. SMDs with processed priority are superior to SMDs with unprocessed priority on energy saving. The result of RLTBB-$\varepsilon = 0.2$ is superior to the result of RLTBB-$\varepsilon = 0.6$ on account of that the solving accuracy of RLTBB-$\varepsilon = 0.2$ is smaller. In addition, we see that RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$ achieve the same result usually. The reason is that the gap between the upper bound and lower bound in RLTBB decreases leapingly (i.e., the gap may decrease from "$> 0.6 \cdot (E_{all}^{local} - L_b)$" to "$\leq 0.2 \cdot (E_{all}^{local} - L_b)$" once).

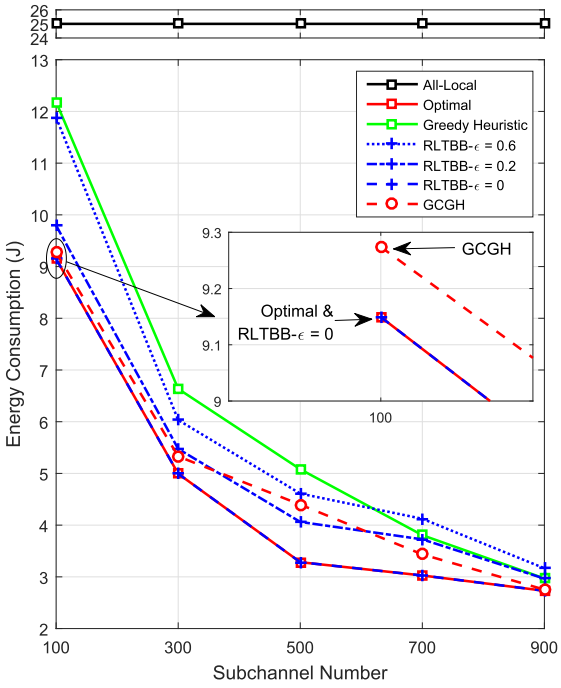


FIGURE 3. Energy consumption w.r.t. subchannel number, $|\mathcal{N}| = 10, F = 5GHz, T_i^{th} = \frac{C_i}{F_i}, \forall i \in \mathcal{N}$.

B. SUBCHANNEL NUMBER

We investigate the impact of subchannel number in Fig. 3. From Fig. 3, we observe that RLTBB-$\varepsilon = 0$ can achieve the optimal result. GCGH and RLTBB-$\varepsilon = 0.2$ can achieve near-optimal results. The energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic achieve 98.09%, 97.14%, 92.90% and 91.96% of the optimal energy saving, respectively. In Fig. 3, the All-Local curve is almost invariant, because the *local execution* needs not to be transmitted wirelessly and the energy consumption of *local execution* does not change with the subchannel number. The other curves decrease with the subchannel number increasing. The reason is that the more subchannels there are, the shorter the transmitting time is Thus, there will be more *edge execution* SMDs and *edge execution* SMDs can economize the transmitting energy. And the total energy consumption on SMDs decrease. Furthermore, we see that GCGH curve is usually under RLTBB-$\varepsilon = 0.2$ curve, however, GCGH curve is above RLTBB-$\varepsilon = 0.2$ curve occasionally. The reason is that GCGH can not grantee the solving accuracy. But, with luck, the overall performance of GCGH is always very well. In Fig. 3, the optimal energy saving increases slowly with subchannel increasingwhen $L \geq 500$, because the number of offloaded SMDs is mainly restrained by the computational resource at this time.

C. CPU-CYCLE FREQUENCY OF MEC SERVER

Fig. 4 shows the impact of the MEC server's CPU-cycle frequency. From Fig. 4, we see that GCGH can achieve the optimal result. GCGH and RLTBB-$\varepsilon = 0.2$ can

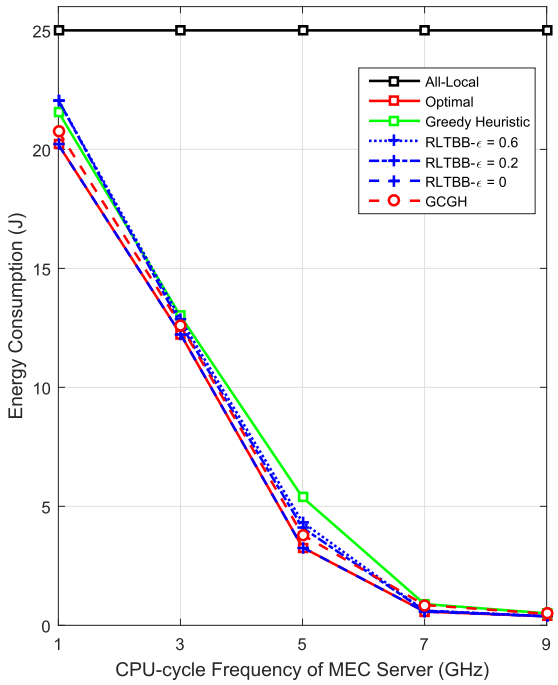


FIGURE 4. Energy consumption w.r.t. CPU-cycle frequency of MEC server, $|\mathcal{N}| = 10, L = 500, T_i^{th} = \frac{C_i}{F_i^l}, \forall i \in \mathcal{N}$.

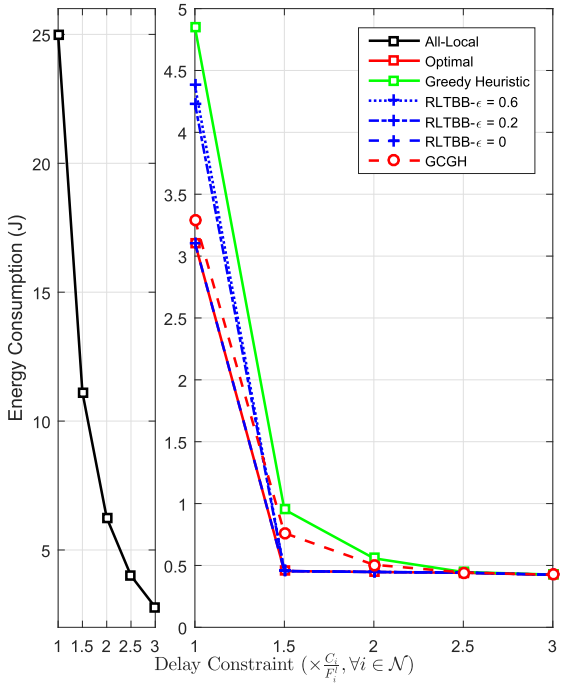


FIGURE 5. Energy consumption w.r.t. delay constraint, $|\mathcal{N}| = 10, L = 500, F = 5GHz$.

achieve near-optimal results. The energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic achieve 96.27%, 90.76%, 90.23% and 90.80% of the optimal energy saving, respectively. In Fig. 4, the All-Local curve is almost invariant, because the *local execution* does not employ the MEC servers computational resource and the energy consumption of *local execution* does not change with the MEC servers CPU-cycle frequency. The other curves decrease with the MEC servers CPU-cycle frequency increasing. The reason is that the higher the CPU-cycle frequency of MEC server is, the shorter the calculating time is. There may be more *edge execution* SMDs to decrease the energy consumption, and *edge execution* SMDs economize the waiting energy. In Fig. 4, the optimal energy saving increases slowly with the MEC server's CPU-cycle frequency increasing when $F \geq 5GHz$, because the number of offloaded SMDs is mainly restrained by the radio resource at this time.

D. DELAY CONSTRAINT

We investigate the impact of delay constraint in Fig. 5. From Fig. 5, we see that RLTBB-$\varepsilon = 0$ can achieve the optimal result. GCGH, RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$ can achieve near-optimal results. The energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic achieve 99.04%, 98.96%, 98.83% and 97.05% of the optimal energy saving, respectively. Furthermore, We observe that the looser the delay constraint is (i.e., $T_i^{th}, \forall i \in \mathcal{N}$ is bigger), the less energy consumption is. The reason is that *local execution* SMDs consume less energy compared with the scenario under stricter delay constraint. And there may be more SMDs which can execute their applications by *edge execution* to save energy. We also observe that the looser delay constraints are, the less energy savings are. The reason is that the looser delay constraints are, the less the energy consumption of All-Local is, that is to say there is less energy to be saved. When "$T_i^{th} \geq 1.5\frac{C_i}{F_i^l}, \forall i \in \mathcal{N}$", curves of Optimal, RLTBB-$\varepsilon = 0$, RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$ are almost invariant, because all the SMDs are offloaded in the results of Optimal, RLTBB-$\varepsilon = 0$, RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$. With delay constraints becoming looser, GCGH and Greedy Heuristic achieve the optimal result. The reason is that when the looser are the delay constraints, the more offloaded SMDs of GCGH and Greedy Heuristic are. With offloaded SMDs increasing to $|\mathcal{N}|$, the difference between GCGH and Greedy Heuristic decreases to 0.

E. SMD NUMBER

We investigate the impact of the SMD number in Fig. 6. On account of the brute-force search taking exponential complexity, we do not simulate the the brute-force search (i.e., Optimal) when there are more than 11 SMDs. We do not simulate RLTBB-$\varepsilon = 0$, RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$ when there are more than 13 SMDs, because RLTBB can not guarantee the complexity. From Fig. 6, we see that RLTBB-$\varepsilon = 0$ can achieve the optimal result. GCGH and RLTBB-$\varepsilon = 0.2$ can achieve near-optimal results. The energy savings (i.e., gaps between the All-Local curve and the other curves) of GCGH, RLTBB-$\varepsilon = 0.2$,

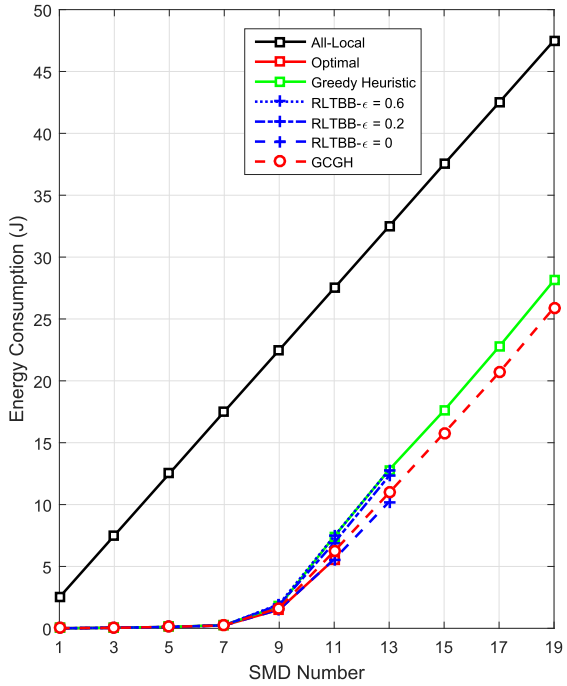


FIGURE 6. Energy consumption w.r.t. SMD number, $L = 500, F = 5GHz, T_i^{th} = \frac{C_i}{F_i}, \forall i \in \mathcal{N}$.

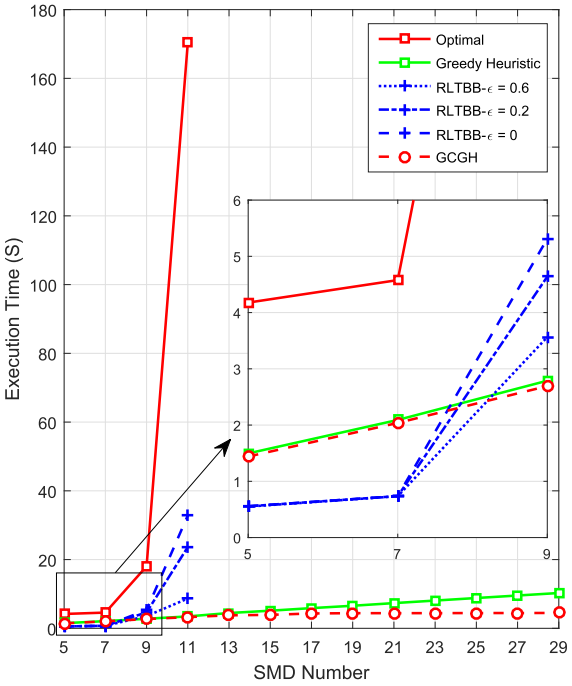


FIGURE 7. Execution time w.r.t. SMD number, $L = 500, F = 5GHz, T_i^{th} = \frac{C_i}{F_i}, \forall i \in \mathcal{N}$.

RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic achieve 99.02%, 97.52%, 96.82% and 96.93% of the optimal energy saving, respectively. In addition, when the SMD number is smaller (i.e., $|\mathcal{N}| \leq 5$), GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic can achieve the optimal result. Energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic increase with the SMD number increasing when $|\mathcal{N}| \leq 9$, because the radio and computational resources are enough to bear all the computation offloading at this time. With offloaded SMDs increasing, the energy saving increases. However, when $|\mathcal{N}| > 9$, energy savings of GCGH, RLTBB-$\varepsilon = 0.2$, RLTBB-$\varepsilon = 0.6$ and Greedy Heuristic are almost invariant. The reason is that the radio and computation resources are limited when $|\mathcal{N}| > 9$. With SMDs increasing, the number of *edge execution* SMDs is almost invariant. Thus, the energy savings are almost invariant.

F. EXECUTION TIME

We investigate the impact of the SMD number in Fig. 7. From Fig. 7, we observe that the execution time of each algorithm increases with the SMD number increasing. For the growth rate, the brute-force search is the fastest, followed by RLTBB-$\varepsilon = 0$ followed by RLTBB-$\varepsilon = 0.2$ followed by RLTBB-$\varepsilon = 0.6$ followed by Greedy Heuristic and finally GCGH. The brute-force search, RLTBB-$\varepsilon = 0$, RLTBB-$\varepsilon = 0.2$ and RLTBB-$\varepsilon = 0.6$ increase exponentially, while, Greedy Heuristic increases linearly, GCGH firstly increases linearly and then converges. The reason is that RLTBB can not guarantee the computation complexity, and the bigger the required tolerance is, the quicker RLTBB terminates. With the increasement of SMD number, the search space of Greedy Heuristic increases accordingly, so its execution time increases linearly. While, due to the indicative function of Gini Coefficient, GCGH can determine a suitable searching space. When the SMD scale is small, this searching space increases with the SMD scale. When the SMD scale is large, this searching space is nearly constant. So the execution time of GCGH firstly increases linearly and then converges, and is acceptable for practical scenario. We also observe that when there are less than or equal to 7 SMDs, RLTBB is faster than GCGH. The reason is that the less SMDs there are, the more efficient the pruning of RLTBB is. Above all, RLTBB is more suitable for the small scale scenario and GCGH is more suitable for the large scale scenario.

VI. CONCLUSION

In this paper, we investigated the MEC computation offloading in a multi-users system. In order to minimize the energy consumption on SMDs, we jointly optimized the offloading selection, radio resource and computational resource allocations. We formulated an energy consumption minimization problem under specific application latencies. To solve the MINLP problem, we proposed the RLTBB method which can not only obtain the optimal result but also calculate a specific suboptimal result with the adjustable solving accuracy. Furthermore, we designed the GCGH scheme to solve

the MINLP problem in polynomial complexity. We also conducted numerous simulations, which validate the energy saving enhancement in our proposed RLTBB and GCGH.

APPENDIX

A. PROOF OF LEMMA 1

When $T_i^{th} \geq \frac{C_i}{F_i^l}, \forall i \in \mathcal{N}$, the strategy where all mobile applications are executed locally satisfies all constraints $C1-C6$. Therefore, the feasible domain $\mathcal{X}$ of problem (9) is nonempty, i.e., there are feasible solutions for problem (9). Because the variables are all bounded, $\mathcal{X}$ is a closed set. So there exists an optimal solution for problem (9). Specifically, $\boldsymbol{\alpha}$ has M feasible values (M is a finite number because $\alpha_i, \forall i \in N$ is a Boolean variable). For a feasible value of $\boldsymbol{\alpha}$, problem (9) can be degraded into a specific problem (16). So problem (9) can be degraded into M specific problem (16) with respect to M feasible values of $\boldsymbol{\alpha}$. In addition, problem (16) is a convex optimization problem and has an optimal solution. Accordingly, there are M optimal solutions for M degraded problem (16). In these M optimal solution, there must be a solution $\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\}$ with the minimal objective value for problem (9). And $\{\boldsymbol{\alpha}^*, \boldsymbol{\theta}^*, \boldsymbol{f}^*\}$ is an optimal solution of problem (9).

B. PROOF OF LEMMA 2

We define the feasible of problem (9) and (10) as $\mathcal{X}$ and $\mathcal{X}1$, respectively. For any feasible solution $\boldsymbol{S}^{\forall} = \{\boldsymbol{\alpha}^{\forall}, \boldsymbol{\theta}^{\forall}, \boldsymbol{f}^{\forall}\}$ belonging to $\mathcal{X}$, it also belongs to $\mathcal{X}1$. The reason is that problem (9) and (10) have the same constraints except $C5$ and $C7$. And $C5$ is stricter than $C5$.

$$\begin{aligned}
&\alpha_i\left(\frac{D_i}{R_i}+\frac{C_i}{f_i}\right)+(1-\alpha_i)\frac{C_i}{f_i^l}\\
&\quad=\alpha_i\left(\frac{D_i}{r_i\theta_i}+\frac{C_i}{f_i}\right)+(1-\alpha_i)\frac{C_i}{f_i^l}\\
&\quad\geq\alpha_i\left[\frac{D_i}{r_i(\varepsilon_1+\theta_i)}+\frac{C_i}{\varepsilon_2+f_i}\right]+(1-\alpha_i)\frac{C_i}{f_i^l} \qquad (29)
\end{aligned}$$

Therefore, $\mathcal{X} \subset \mathcal{X}1$. In addition, the objective function value of problem (9) is larger than or equal to the objective function value of problem (10)

$$\begin{aligned}
&\sum_{i=1}^{N}\left[\alpha_i\left(\frac{P_i^T D_i}{r_i}\beta_i+P_i^I C_i\gamma_i\right)+(1-\alpha_i)\kappa(f_i^l)^2C_i\right]|_{\boldsymbol{S}^{\forall}}\\
&\geq\sum_{i=1}^{N}\left\{\alpha_i\left[\frac{P_i^T D_i}{r_i(\varepsilon_1+\theta_i)}+\frac{P_i^I C_i}{\varepsilon_2+f_i}\right]+(1-\alpha_i)\kappa(f_i^l)^2C_i\right\}|_{\boldsymbol{S}^{\forall}}
\end{aligned} \qquad (30)$$

Thus, the optimal result of problem (10) is small than or equal to the optimal result of problem (9), i.e., Problem (10) is sensible to ε_1 and ε_1 for obtaining a lower bound of problem (9).

C. PROOF OF LEMMA 3

An inverse function with the form $f(x) = \frac{1}{x}$ is a convex function. Since nonnegative weighted sums and composition with an affine mapping are the operations that preserve convexity of functions, all the constraints of problem (15) and (16) can be equally translated the form where some convex functions is less than or equal to a constant. Obviously, the feasible domains of problem (15) and (16) are convex sets, the objective functions of problem (15) and (16) are convex functions. Therefore, problem (15) and (16) are convex optimization problems.

D. PROOF OF THEOREM 1

Firstly, we prove that RLTBB can converge. For the upper bound U_b of problem (9), U_b is nonincreasing with b, because the obtained feasible solution will not be less with b increasing and U_b is the smallest of these optimal values. For the lower bound L_b of problem (9), L_b is nondecreasing with b, because the feasible is smaller. So $U_b - L_b$ is nonincreasing with b. When $\mathcal{N}_d$ has N elements, problem (20) and (21) will degraded into problem (16). At this time, $\overline{\boldsymbol{\alpha}}_{2b} = \underline{E}_{2b}$, $\overline{\boldsymbol{\alpha}}_{2b+1} = \underline{E}_{2b+1}$, $\overline{E}_{2b} = \underline{E}_{2b}$ and $\overline{E}_{2b+1} = \underline{E}_{2b+1}$. So there must exist the case where $U_b = L_b$ with b increasing (The most obvious case is that the depthes of all feasible nodes are N).

Secondly, we prove that the computation complexity of RLTBB is exponential. In the worst case, BB requires effort that grows exponentially with problem size, but in some cases BB converges with much less effort [37]. Because problem (15)(16)(20) and (21) are convex optimization problems,respectively, we define the computation complexities are $O(\mathcal{C}1)$, $O(\mathcal{C}2)$, $O(\mathcal{C}3)$ and $O(\mathcal{C}4)$ which are all polynomial. For line 3-4, the complexity is $O(\mathcal{C}1+\mathcal{C}2)$. The cycle number of "while loop" can not be guaranteed. But, in the worst case, the "while loop" has 2^N iterations. For line 6-9, the complexity is $O(\mathcal{C}3+\mathcal{C}4+2\mathcal{C}2)$. For line 10-15, the complexity is $O(4|\mathcal{L}_{b+1}^{up}|)$, where $|\mathcal{L}_{b+1}^{up}| = b+1$ in the worst case. Above all, the computation complexity of RLTBB can be given

$$\begin{aligned}
&O(\mathcal{C}1+\mathcal{C}2+2^N(O(\mathcal{C}3+\mathcal{C}4+2\mathcal{C}2)+O(1+2^N)))\\
&\quad=O(\mathcal{C}1+2^N\max\{\mathcal{C}2,\mathcal{C}3,\mathcal{C}4\}+2^{2N})\\
&\quad=O(2^{2N}) \qquad (31)
\end{aligned}$$

This is the worst case, but with luck, this case hardly happens.

E. PROOF OF THEOREM 2

If $\frac{D_i}{r_iL}+\frac{C_i}{F} > T_i^{th}$, SMD i can not satisfy its specific delay anyway, and must execute its task locally. If $\frac{P_i^T D_i}{r_iL}+\frac{P_i^I C_i}{F} \geq \kappa(f_i^l)^2C_i$, *edge execution* has no energy saving compared with *local execution* for SMD i, and SMD i has no motivations to offload its task. Above all, SMDs, which do not satisfy the Basic Offloading Condition, must execute their tasks locally.

F. PROOF OF THEOREM 3

Firstly, the Stage 1 of Algorithm 2 has N iterations to classify each SMD in SMDs set $\mathcal{N}$. Secondly, for the Stage 2, line 11 has $|\mathcal{S}_o|$ iterations to calculate $\Phi(i)$. The sorting of line 12 has $O(|\mathcal{S}_o|^2)$ complexity. The calculation of lines 13-15 has

$|\mathcal{S}_o^s|$ iterations. Thirdly, the "*for loop*" in Stage 3 has $|\mathcal{S}_o^{s1}|$ iterations. Problem (16) can be solved in polynomial complexity because problem (16) is a convex optimization problem. We define the polynomial complexity of problem (16) as $O(\mathcal{C})$. Above all, the computational complexity of GCGH can be given

$$O(N + |\mathcal{S}_o| + O(|\mathcal{S}_o|^2) + |\mathcal{S}_o^s| + |\mathcal{S}_o^{s1}|O(\mathcal{C})) = O(N + |\mathcal{S}_o|^2 + |\mathcal{S}_o^s| + |\mathcal{S}_o^{s1}|\mathcal{C}) \quad (32)$$

Due to $|\mathcal{S}_o^{s1}| \leq |\mathcal{S}_o^s| \leq |\mathcal{S}_o| \leq N$, the computational complexity of GCGH can be further expressed as $O(\max\{N^2, N\mathcal{C}\})$. Since $O(\mathcal{C})$ is polynomial, $O(\max\{N^2, N\mathcal{C}\})$ is polynomial.

REFERENCES

[1] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 9, pp. 2546–2559, Sep. 2016.

[2] B. Fan, S. Leng, and K. Yang, "A dynamic bandwidth allocation algorithm in mobile networks with big data of users and networks," *IEEE Netw.*, vol. 30, no. 1, pp. 6–10, Jan. 2016.

[3] "Forecast and methodology, 2012–2017," Cisco Visual Networking Index, San Jose, CA, USA, White Paper, May 2013.

[4] E. Ahmed, A. Gani, M. Sookhak, S. H. Ab Hamid, and F. Xia, "Application optimization in mobile cloud computing: Motivation, taxonomies, and open challenges," *J. Netw. Comput. Appl.*, vol. 52, pp. 52–68, Jun. 2015.

[5] L. Ericsson, "More than 50 billion connected devices," White Paper, 2011.

[6] 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.

[7] J. Shen, H. Tan, J. Wang, J. Wang, and S. Lee, "A novel routing protocol providing good transmission reliability in underwater sensor networks," *J. Internet Technol.*, vol. 16, no. 1, pp. 171–178, 2015.

[8] Z. Pan, Y. Zhang, and S. Kwong, "Efficient motion and disparity estimation optimization for low complexity multiview video coding," *IEEE Trans. Broadcast.*, vol. 61, no. 2, pp. 166–176, Jun. 2015.

[9] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.

[10] J. Liu, E. Ahmed, M. Shiraz, A. Gani, R. Buyya, and A. Qureshi, "Application partitioning algorithms in mobile cloud computing: Taxonomy, review and future directions," *J. Netw. Comput. Appl.*, vol. 48, pp. 99–117, Feb. 2015.

[11] B. Gu and V. S. Sheng, "A robust regularization path algorithm for ν-support vector classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1241–1248, May 2017.

[12] M. T. Beck, S. Feld, C. Linnhoff-Popien, and U. Pützschler, "Mobile edge computing," *Inf. Spektrum*, vol. 39, no. 2, pp. 108–114, 2016.

[13] Y. Yu, J. Zhang, and K. B. Letaief. (2016). "Joint subcarrier and CPU time allocation for mobile edge computing." [Online]. Available: https://arxiv.org/abs/1608.06128

[14] E. Cuervo *et al.*, "MAUI: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 49–62.

[15] 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*, Mar. 2012, pp. 945–953.

[16] F. Xia, F. Ding, J. Li, X. Kong, L. T. Yang, and J. Ma, "Phone2cloud: Exploiting computation offloading for energy saving on smartphones in mobile cloud computing," *Inf. Syst. Frontiers*, vol. 16, no. 1, pp. 95–111, 2014.

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

[18] Y. Cao, T. Jiang, and C. Wang, "Optimal radio resource allocation for mobile task offloading in cellular networks," *IEEE Netw.*, vol. 28, no. 5, pp. 68–73, Sep. 2014.

[19] ETSI. *ETSI Announces First Meeting of New Standardization Group on Mobile-Edge Computing*, accessed on Oct. 30, 2014. [Online]. Available: http://www.etsi.org/news-events/news/838-2014-10-news-etsi-announces-first-meeting-of-new-standardization-group-on-mobile-edge-computing

[20] Y. D. Lin, E. T. H. Chu, Y. C. Lai, and T. J. Huang, "Time-and-energy-aware computation offloading in handheld devices to coprocessors and clouds," *IEEE Syst. J.*, vol. 9, no. 2, pp. 393–405, Jun. 2015.

[21] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

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

[23] W. Labidi, M. Sarkiss, and M. Kamoun, "Energy-optimal resource scheduling and computation offloading in small cell networks," in *Proc. 22nd Int. Conf. Telecommun. (ICT)*, Apr. 2015, pp. 313–318.

[24] K. Zhang *et al.*, "Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[25] 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.

[26] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing May help to save energy in cloud computing," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1728–1739, May 2016.

[27] 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, Sep. 2013.

[28] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," *HotCloud*, vol. 10, p. 4, Jun. 2010.

[29] K. Zhu and E. Hossain, "Virtualization of 5G cellular networks as a hierarchical combinatorial auction," *IEEE Trans. Mobile Comput.*, vol. 15, no. 10, pp. 2640–2654, Oct. 2016.

[30] V. Chandrasekhar and J. G. Andrews, "Spectrum allocation in tiered cellular networks," *IEEE Trans. Commun.*, vol. 57, no. 10, pp. 3059–3068, Oct. 2009.

[31] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 9, no. 11, pp. 3590–3600, Nov. 2010.

[32] Y. Li *et al.*, "Energy-efficient transmission in heterogeneous wireless networks: A delay-aware approach," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7488–7500, Sep. 2016.

[33] H. D. Sherali and W. P. Adams, *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Norwell, MA, USA: Kluwer, 1999.

[34] Y. Niu, C. Gao, Y. Li, D. Jin, L. Su, and D. Wu, "Boosting spatial reuse via multiple-path multihop scheduling for directional mmWave WPANs," *IEEE Trans. Veh. Technol.*, vol. 65, no. 8, pp. 6614–6627, Aug. 2016.

[35] J. Clausen, "Branch and bound algorithms-principles and examples," Ph.D. dissertation, Dept. Comput. Sci., Univ. Copenhagen, København, Denmark, 1999, pp. 1–30.

[36] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, Jul. 1960.

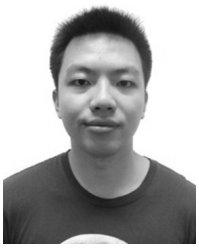
[37] S. Boyd and J. Mattingley, "Branch and bound methods," Stanford Univ., Stanford, CA, USA, Course notes EE364b, Mar. 2007.

[38] 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. 17th IEEE Symp. Comput. Commun. (ISCC)*, Cappadocia, Turkey, Jul. 2012, pp. 59–66.

PENGTAO ZHAO received the B.S. degree in communications engineering from Beijing Jiaotong University, Beijing, China, in 2014. He is currently pursuing the Ph.D. degree in information and communications engineering with the Beijing University of Posts and Telecommunications, Beijing. His research interests include mobile edge computing, network slicing, and convex optimizations.

HUI TIAN (M'03) received the M.S. degree in microelectronics in 1992, and the Ph.D. degree in circuits and systems from the Beijing University of Posts and Telecommunications (BUPT), in 2003. She is currently a Professor with BUPT and the Director of the State Key Laboratory of Networking and Switching Technology. Her research interests include LTE and 5G system design, MAC protocols, resource scheduling, cross-layer design, cooperative relaying in cellular systems, and ad hoc and sensor networks. She is a Committee Member of the Beijing Key Laboratory of wireless communication testing technology, a Core Member of the Innovation Group of the National Natural Science Foundation of China, a member of the China Institute of Communications, and an Expert with the Unified Tolling and Electronic Toll Collection Working Group, China National Technical Committee on ITS Standardization.

She was a co-recipient of the National Award for Technological Invention, Science and Technology Award of China Communications and the Ten major scientific and technological progresses Award of China's colleges and Universities for her contribution in wireless communication. She was a Lead Guest Editor of the *EURASIP Journal on Wireless Communications and Networking*. She has been a TPC Member of IEEE conferences, including GlobalCom, WCNC, WPMC, PIMRC, VTC and ICCC,and the Reviewer for the IEEE TVT, the IEEE CL, the *IET Communications*, the *Transactions on Emerging Telecommunications Technologies*, the *EURASIP Journal on Wireless Communications and Networking*, the *Journal of Networks*, the *Majlesi Journal of Electrical Engineering*, the *Journal of China University of Posts and Telecommunications*, the *Chinese Journal of Electronics*, the *Journal of Electronics and Information Technology*, and the *Chinese Journal of Aeronautics*.



CHENG QIN (S'14) received the B.S. degrees in communications engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011, where he is currently pursuing the Ph.D. degree in information and communications engineering. His research interests include cooperative systems, MIMO systems, energy harvesting, and convex optimizations.

GAOFENG NIE received the B.S. and Ph.D. degrees from the Beijing University of Posts and Telecommunications (BUPT) in 2010 and 2016, respectively. He is currently a Lecturer with BUPT. His research interests are radio resource management in ultra dense networks and key technologies in 5G wireless networks.