IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 28, 2018, accepted December 13, 2018, date of publication December 25, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2889815

Jointly Optimized Energy-Minimal Resource Allocation in Cache-Enhanced Mobile Edge Computing Systems

PENG LIU[1], GAOCHAO XU[1], (Member, IEEE), KUN YANG[2], (Senior Member, IEEE), KEZHI WANG[3], (Member, IEEE), AND XIANGYU MENG[1]

[1]College of Computer Science and Technology, Jilin University, Changchun 130012, China
[2]School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, U.K.
[3]Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8ST, U.K.

Corresponding author: Xiangyu Meng (pliu16@maisl.jlu.edu.cn)

This work was supported in part by the Jilin Industrial Innovation Special Fund Project under Grant 2017C028-4.

ABSTRACT Mobile edge computing (MEC) has attracted extensive studies recently due to its ability to augment the computational capabilities of mobile devices. This paper considers a cache-enhanced multiuser MEC system where the task can be cached in the MEC servers to avoid the transmission of duplicate data. To further improve the energy efficiency and satisfy the users' requirement on delay, we jointly optimize caching, computation, and communication resources in this system. The formulated problem is a mixed integer non-convex optimization problem that is very challenging to solve. We thus propose an efficient iterative algorithm by jointly applying the block coordinate descent and convex optimization techniques, which is guaranteed to converge at least a suboptimal solution. Specifically, the formulated joint optimization problem is decomposed into two subproblems to optimize caching policy and resource allocation, respectively, which are alternately optimized by convex optimization in each iteration. To further speed up the algorithm convergence, an efficient initialization scheme based on the linear weighted method is proposed for caching policy. The extensive simulation results are provided to demonstrate that the proposed jointly optimizing caching, computation, and communication method can improve the energy efficiency with lower time cost compared with other benchmark methods.

INDEX TERMS Mobile edge computing, edge caching, joint optimization, convex optimization, block coordinate descent.

I. INTRODUCTION

With the rapid advancement of intelligent mobile device and internet of things, the new type of mobile applications, such as artificial intelligence,augmented reality and virtual reality, are constantly emerging, which put forward higher requirement on the computing power, data access of mobile devices. However, mobile devices generally have limited resources of computation, communication, storage and energy, which makes it face great challenge for finishing such computation-insensitive and latency-sensitive applications [1]. To address this challenge, several offloading based computing models were proposed, typically including mobile cloud computing (MCC) [2], [3], mobile edge computing (MEC) [4], [5] ect. These computing models strengthen capability by migrating part or all of the application from resource-hungry mobile devices to powerful cloud servers. Especially, MEC that deploys cloud servers at the edge of radio access networks to avoid the large delay has received growing attentions from both industry and academia.

However, despite being in close proximity to the servers, mobile devices still have to consume additional delay and energy on offloading. In addition, the massive task data transmission will bring another challenges on the access network. For further improvement, some researches proposed edge caching [6], [7], which caches the popular contents or tasks at the edge to reduce the delay and energy consumption by avoiding unnecessary duplicate transmissions.

Unfortunately, on the one hand, current concerns of edge caching are mainly on content caching, exploring the optimal caching policy to increase the hit rate [8], [9]. But ignore the fact that task caching actually has more extensive and universal application scene. On the other hand, task caching is

a complicated problem, not only depending on task popularity but on task size and task complexity etc, which is hard to find out a universal caching method. Besides, caching policy will affect the allocation of computation and communication resources and vice versa, a joint caching and resources allocation policy is essential to improve the energy efficiency of cache-enhanced MEC system.

In this paper, we propose a cache-enhanced MEC system that caches the tasks on the edge servers. Different from previous researches that concern on caching task results, we define the task caching as caching the application program and task data on the edge servers. This makes it possible to deal with the same data according to the different user requests, which enhances the system flexibility and increases the cache hit rate in a sense. Due to the small size and few types, it is realistic to assume that all the programs are cached on the edge servers. Besides, we neglect the transmission of configuration and request data because of its small size. Therefore, we can consider that the cached task can be executed on the edge servers without any data transmission. Once cached, the tasks will surely executed on the edge servers, or it will be executed on the mobile devices and edge servers parallel to reduce the time and energy consumption. To further improve the energy efficiency of proposed task caching-enhanced MEC system, we formulate a problem that jointly optimize the caching policy and resource allocation of computation and communication to minimize the energy consumption of mobile devices while meet the users' requirements on task delay. Finally, we propose an efficient iterative algorithm by jointly applying the block coordinate descent and convex optimization techniques to solve the formulated mixed integer optimization problem. The main contributions to this work are summarized as follows.

In terms of system model. We propose a cache-enhanced MEC system that integrates task caching into MEC to reduce the duplicate transmission. Different from researches that caching task result, task caching in the proposed system refers to caching the application program and related task data. Due to the small size and few types of program, we assume that the programs are all cached in the edge servers, and what the caching policy decide is whether to caching the task data. Once cached, the task will be executed on the edge servers, or it will be executed on the edge servers and local device parallel. Proposed model is able to reduce the time and energy consumption by jointly using the resource of caching, computation and communication.

In terms of formulated problem. Caching in general is a longer process (e.g., in minutes or hours) and reflect the statistic of the system, but resource allocation is a much shorter process (e.g. in seconds or millisecond) and utilize the instantaneous system state information as much as possible. The contradiction make it impossible to optimize the caching policy and resource allocation at the same time. To avoid the contradiction, we consider that a caching policy can be used in several following time slots, but the computation and communication resources are reallocated for users in each time slot. Thereby, we formulate a joint optimization problem of caching policy and communication, computation resources to minimize the energy consumption of mobile devices while meet t he users' requirement on delay.

In terms of solution. The formulated problem is a mixed integer non-convex optimization problem which is hard to solve and no well-studied optimization techniques can be used directly. In this paper, we propose an efficient iterative algorithm by jointly applying the block coordinate descent and convex optimization techniques. Specifically, we decompose the original problem into two subproblems and alternately optimize them in each iteration. Besides, we give a linear weighted method based initialization method for caching policy to overcome the excessive number of iterations and large time consumption.

The rest of this paper is organized as follows: Section II gives some related work. In section III, system model and problem formulation are introduced. Next, section IV gives the detail problem solution. In section V, the simulation results and analysis are shown. Finally, we conclude this paper in section VI.

II. RELATED WORK

As a key solution for insufficient physical resources of mobile devices, MEC has attracted extensive attentions in recent years. However, it still faces some challenges to be more widely and efficiently applied.

In MEC system, the efficiency of computation offloading largely depends on the quality of communication because data transmission is necessary. This calls for incorporating the characteristics of communication and computation, jointly optimizing communication and computation resources. In [10], a method jointly optimizing mobile-transmission power and CPU cycles assigned to each application is proposed to minimize the power consumption at the mobile side, under an average latency constraint. By Karush-Kuhn-Tucher (KKT) condition, they get the one-to-one relationship between the transmit power and the percentage of CPU cycles assigned to each user. In [11], Wang *et al.* considered a single-user scenario, they focus on partial computation offloading by jointly optimizing the CPU cycle frequency, transmission power and offloading ratio. In [12], the radio resource and computational resource allocation were jointly optimized to minimize the weighted sum energy consumption in a MIMO system. In [13], Zhao *et al.* jointly optimized the offloading selection, radio resource allocation and computational resource allocation coordinately to minimize the energy consumption of mobile device. Reference [14] developed an online joint radio and computational resource management algorithm for multi-user MEC systems, they aimed at minimizing the long-term average weighted sum power consumption of the mobile device and MEC server, under the task buffer stability constraint. Reference [15] proposed a game theoretic approach to get the computation offloading decision among multiple mobile device users.

By jointly optimizing the computation and communication resource, the resources utilization and energy efficiency of MEC system has been improved significantly. However, offloading operation and content delivery during the execution will still consume additional time and energy. To further reduce task latency and improve energy efficiency, some researches has proposed edge caching, caching the popular contents on the edge of network to avoid duplicate transmission.

Gu *et al.* [16] investigate the storage allocation problem in MBS caching and propose a heuristic method to solve the NP-hard problem. Bai *et al.* [17] proposed a caching based D2D communication scheme that consider the social relations among users and their interests. Traverso *et al.* [18] proposed a dynamic popularity based caching model, using a pulse with two parameters to model each content. Ahlehagh and Dey [19] defined the user preference profile (UPP) as the probability that a user requests video of specific video category and propose a caching policy based on UPP. Sengupta *et al.* [20] proposed a learning based caching policy that solving the problem of distributed caching in SBSs from a reinforcement learning view. They adopted coded caching to reduce caching problem to a linear program that considering the network connectivity, getting better performance than the uncoded scheme.

These researches mainly focus on content caching, including Where to Cache? What to cache? How to cache? However, some researches has introduced edge caching into MEC and proposed task caching which will be extensive used with the size of task data and results increasing. Reference [21] formulated an optimization problem that jointly considering the offloading decision, physical spectrum resource, computation resource and content caching strategy. And developed an alternating direction method of multipliers (ADMM) based algorithm to solve the optimization problem to maximize the revenue. Reference [22] proposed an MEC enhanced adaptive bitrate (ABR) video delivery scheme that combines content caching and ABR streaming technology. The joint cache and radio resource allocation (JCRA) problem is tackled into a matching problem and be solved to make the cooperation between caching and radio resources. Reference [23] proposed a joint caching and offloading mechanism that involves task uploading and executing for tasks with uncached computation results as well as computation result downloading for all tasks at the BS. Then they formulate the problem that optimally allocate the storage resources at BS for caching computation results as well as the uploading and downloading time durations to minimize average total energy minimization problem. Reference [24] studied the energy-efficient resource allocation in software-defined mobile networks with mobile edge computing and caching. They formulate a novel optimization problem by jointly considering bandwidth provisioning and content source selection and solve the problem by decoupling the problem. Reference [25] introduced the concept of task caching and investigated the problem of joint optimization of task caching and offloading on edge cloud with computing and storage resource constraint. They formulated the optimization problem and proposed an alternating iterative based algorithm to solve it. Reference [26] studied the conditions under which the area power consumption is minimized with respect to BS transmit power, while ensuring a certain quality of service (QoS) in terms of coverage probability. Furthermore, they provide the optimal BS transmission power that maximizes the area spectral efficiency per unit total power spent.

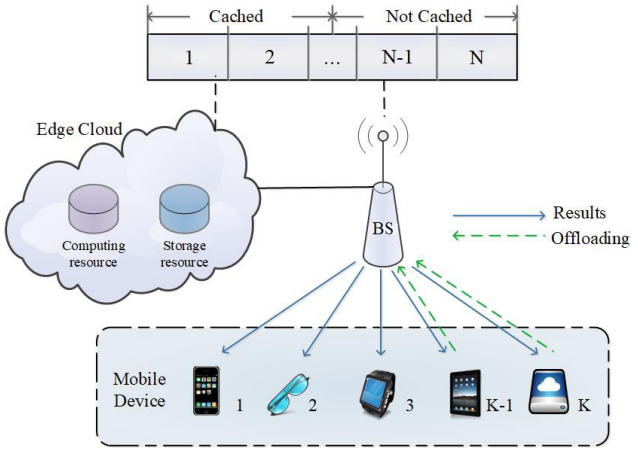


FIGURE 1. System model. One BS, K users and N tasks.

III. SYSTEM MODEL AND PROBLEM FORMULATION

As illustrated in Fig.1, we consider a multi-user cache-enhanced MEC system with one BS and K mobile devices under its coverage, denoted by $\mathcal{K} \triangleq \{1, 2, 3, .., K\}$. MEC servers directly connected with the BS, providing computation and caching capability. Supposing that the time is divided into discrete time slots with duration T and each mobile device has a computation-insensitive and latency-sensitive task at the beginning of each slot. MEC servers are able to cache tasks selectively with limited storage space to reduce the execution time. Once cached, the task will be executed on the MEC server. Otherwise, it will be parallel executed on the servers and local devices. Since caching and allocation scheme of computation and communication resources are interactive, a joint allocation scheme is necessary for minimizing the energy consumption of mobile devices.

Taking augmented reality(AR) as a typical example of proposed system model. Some mobile devices require to run augmented reality application and the application program is cached on the MEC server. For the mobile devices, they may need to deal with the same task data according to different requirements within a certain scope of time and space. If the task data is cached, the task can be executed on the edge server without any data transmission under the premise of ignoring the configuration and request data. Therefore, the cached task will be executed on the edge servers, and the others will be executed on the edge server and local device parallel.

A. TASK MODEL

Supposing that a task set with N computation-incentive and latency-sensitive tasks, denoted by $\mathcal{N} \triangleq \{1, 2, 3, \ldots, N\}$. Each task $n \in \mathcal{N}$ is characterized by two parameters, i.e the size of task input d_n (KB) and the computational complexity c_n (Megacycle). Besides, each task must be completed within one time slot T. Note that we neglect the size of computation result, because it is much smaller than the input data.

At the beginning of each time slot, mobile devices request a task randomly from set $\mathcal{N}$, where one task may be requested by multiple users simultaneously. Without loss of generality, each mobile user has different preference on the tasks and hence we model the task popularity. Specifically, the mobile k need to execute the task $N_k \in \mathcal{N}$. Defined that $P_{N_k}(n) = Pr[N_k = n]$ denotes the probability that the mobile k execute the task n, where $n \in \mathcal{N}$ and $\sum_{n \in \mathcal{N}} P_{N_k}(n) = 1, \forall k \in \mathcal{K}$. Note that the task variables $N_k, k \in K$ are independently distributed, and the probability distribution function $P_{N_k}(n)$ may be different. For one time slot, the tasks executed by all the mobile devices construct the task state, defined by $Y \triangleq (N_1, N_2, \ldots, N_K)$. Supposing that we consider M time slots in total, $m \in [1, 2, 3, \ldots, M]$ denotes the sequence number of time slot and the corresponding task state denoted by Y_m. Besides, $S_n(m) \subseteq \mathcal{K}$ denotes the set of mobile devices that execute the task n in the time slot m and $|S_n(m)|$ denotes size of the set.

B. COMPUTATION MODEL

Consider that tasks could be partitioned into two parts of any size, executing on the mobile devices and edge servers parallel. Therefore, the computation model including two parts, i.e. local execution and edge execution. Note that the next descriptions all take the mobile device k execute task n as an example.

1) LOCAL EXECUTION MODEL

For local execution, the CPU computing power of mobile device k is denoted by f_k^{loc}. Thus, the time consumption for local execution can be given by:

$$T_k^{loc} = \frac{c_n}{f_k^{loc}}. \tag{1}$$

The energy consumption for one computing cycle can be given by $\kappa f_k^{loc\,2}$ where κ is the effective switched capacitance that depends on the chip architecture, and thus the total energy consumption for local execution are as follows:

$$E_k^{loc} = \kappa f_k^{loc\,2} c_n, \tag{2}$$

where we set $\kappa = 10^{-26}$ in this paper.

2) MEC SERVER EXECUTION MODEL

For edge execution, the computation resources are reallocated at the beginning of each time slot. The CPU computing power that edge servers allocate to mobile device k in the m-th time slot is denoted by $f_{k,m}^{edge}$. The time consumption of mobile device k in the m-th time slot can be expressed as follows:

$$T_{k,m}^{edge} = \frac{c_n}{f_{k,m}^{edge}}. \tag{3}$$

Previous researches usually consider that mobile device does not consume any energy when edge execution, but the energy of edge servers. However, they neglect the fact that the energy consumption and computation time is directly proportional, longer waiting time brings more energy consumption. In this paper, the energy consumed by the mobile device k when edge execution is considered and can be given by:

$$E_{k,m}^{edge} = P_k^{wait} T_{k,m}^{edge}, \tag{4}$$

where P_k^{wait} represents the power in waiting state. Note that the computation resource of edge server can not be infinite, it will be reallocated at each time slot. By F denotes the total computation resource, and hence have the following constraints in each time slot as:

$$\sum_{n \in \mathcal{N}} \sum_{k \in S_n(m)} f_{k,m}^{edge} \leq F, \quad m \in \{1, 2, \ldots, M\}. \tag{5}$$

C. COMMUNICATION MODEL

For the edge execution, the tasks need to be offloaded to the edge server with the uplink. And thus the communication resources are also allocated at the beginning of each time slot. Note that we assumes that each mobile device will be assigned one or more subchannels for offloading and the subchannels assigned to each mobile device are homogeneous (i.e., the channel power gain and transmission power of different subchannels are the same for a mobile device, but different for different mobile device [27]). Let p_k denotes the transmission power of mobile device k, and $h_{k,m}$ denotes the channel power gain from mobile device k to the BS in the m-th time slot, $\theta_{k,m}$ denotes the number of subchannels assigned to mobile device k in the m-th time slot. Based on the definition, the data rate between mobile device k and the BS in the m-th time slot can be given as follows:

$$r_{k,m} = \theta_{k,m} B \log_2\left(1 + \frac{p_k h_{k,m}}{\sigma^2}\right), \tag{6}$$

where B denotes the channel bandwidth, and σ^2 denotes the noise power. For simplicity, we consider both B and σ^2 are constant. The transmission power p_k is constant for each mobile device, but may be different from each other. Note that the channel state is usually changing in different moments, thus we define $\mathcal{H} = \{1, 2, 3, \ldots, H\}$ as the channel state set, from which the channel power gain of mobile device is selected and keep unchanged in one time slot. The channel power gain of mobile device k denoted by H_k and $p_{H_k}(h) = Pr[H_k = h]$ denotes the probability that H_k takes the value $h \in \mathcal{H}$. Supposing that each mobile device has the independently distributed p_{H_k}, and $\sum_{h \in \mathcal{H}} p_{H_k}(h) = 1 \quad \forall k \in \mathcal{K}$.

The offloading time for mobile device k to execute task n in the time slot m can be given by:

$$T_{k,m}^{tran} = \frac{d_n}{r_{k,m}}. \tag{7}$$

The energy consumption can be given by:

$$E_{k,m}^{tran} = p_k T_{k,m}^{tran}, \tag{8}$$

by L denotes the total number of subchannels, we have the following constraints on communication model:

$$\sum_{n\in\mathcal{N}} \sum_{k\in S_n(m)} \theta_{k,m} \le L, \quad m \in \{1, 2, \ldots, M\}. \tag{9}$$

D. CACHE MODEL

Based on previous definition of task caching, the application programs are all cached in the MEC servers and caching policy is used to decide whether to cache the task data. There is no doubt that task caching can reduce task latency and energy consumption because of no data transmission under the premise of ignoring the configuration and request data. However, how to assign the limited caching capacity is a complicated problem, which depends on the task popularity, size and computation complexity etc. Assume that the size of caching capacity is D, by a_n denotes the caching decision of task n, we define the cache model. Specifically, $a_n = 1$ denotes that the task n is cached in the MEC server and $a_n = 0$ otherwise. Note that if task n is cached, owing to there is no task data transmission and we ignore the result transmission latency, the task latency only depends on edge execution time. Meanwhile, edge execution is usually much faster and consume less energy of mobile device than local execution. Therefore, we consider that if the task executed by mobile device k is cached, the whole task is executed by the edge servers. Otherwise, it will be executed on edge server and local device parallel to further reduce the latency and energy consumption. For the uncached tasks, let $\alpha_{k,m} \in [0, 1]$ denotes the offloading ratio of mobile device k in the m-th time slot, $\alpha_{k,m} = 1$ means the completed task is executed on the edge server and $\alpha_{k,m} = 0$ means the completed task is executed on the local device, otherwise, the task is partitioned into two parts, $\alpha_{k,m}$ for edge execution and $(1 - \alpha_{k,m})$ for local execution. Based on above model, if the mobile device k executes the cached task n, the time consumption $T_{k,m}^{cache}$ is equal to the time consumption of edge execution (i.e.$T_{k,m}^{cache} = T_{k,m}^{edge}$), and the energy consumption $E_{k,m}^{cache}$ is equal to the consumption of edge execution. (i.e.$E_{k,m}^{cache} = E_{k,m}^{edge}$). While the executed task n is not cached, the task will be executed on the edge server and local device at the same time and thus the time consumption can be given by:

$$T_{k,m}^{uncache} = max\big((1-\alpha_{k,m})T_k^{loc}, \alpha_{k,m}(T_{k,m}^{tran} + T_{k,m}^{edge})\big). \tag{10}$$

Besides, the energy consumption can be given by:

$$E_{k,m}^{uncache} = (1 - \alpha_{k,m})E_k^{loc} + \alpha_{k,m}(E_{k,m}^{edge} + E_{k,m}^{tran}). \tag{11}$$

Then, the total energy consumption of mobile devices in the m-th ($m \in \{1, 2, 3, \ldots, M\}$) time slot can be given by:

$$E_m = \sum_{n\in\mathcal{N}} \sum_{k\in S_n(m)} (1 - a_n)E_{k,m}^{uncache} + a_n E_{k,m}^{cache}. \tag{12}$$

E. PROBLEM FORMULATION

Note that caching is in general a long process and reflect the statistic of system, but resource allocation is usually an instant process and utilizes the real-time states. Therefore, we consider the scenery that the task states in the continuous multiple time slots share the same caching policy, i.e. the resource allocation changes in different time slots but the caching policy remains unchanged. The goal of this paper is to minimize the total energy consumption of continuous M time slots by jointly optimizing caching policy, computation, communication resources (J3C), formulating the J3C problem as:

$$\begin{aligned}
P1: \quad & \min_{a,f,\theta,\alpha} \sum_{m=1}^{M} E_m \\
& subject\ to: \ C1: a_n T_{k,m}^{cache} + (1 - a_n)T_{k,m}^{uncache} \le T \\
& \qquad \forall n \in \mathcal{N},\ m \in \{1, 2, \ldots, M\},\ k \in S_n(m) \\
& C2: \sum_{n\in\mathcal{N}} a_n d_n \le D \\
& C3: \sum_{n\in\mathcal{N}} \sum_{k\in S_n(m)} \alpha_{k,m} c_n \le F \cdot T \quad \forall m \in \{1, 2, \ldots, M\} \\
& C4: \sum_{n\in\mathcal{N}} \sum_{k\in S_n(m)} f_{k,m}^{edge} \le F \quad \forall m \in \{1, 2, \ldots, M\} \\
& C5: \sum_{n\in\mathcal{N}} \sum_{k\in S_n(m)} \theta_{k,m} \le L \quad \forall m \in \{1, 2, \ldots, M\} \\
& C6: a_n \in \{0, 1\} \quad \forall n \in \mathcal{N} \\
& C7: \alpha_{k,m} \in [0, 1] \quad \forall k \in \mathcal{K}
\end{aligned} \tag{13}$$

where $\mathbf{a}, \mathbf{f}, \boldsymbol{\alpha}, \boldsymbol{\theta}$ denotes the optimal solution of P1. $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ is a n dimensions vector, denotes the caching policy consist of caching decision of each task. $\mathbf{f}$ is $k \times m$ solution matrix of computation resources allocation, representing the allocated computation power of mobile device k in the m-th time slot.Similarly $\boldsymbol{\theta}, \boldsymbol{\alpha}$ denotes the solution matrix of communication resources and offloading ratio. The objective function denotes the minimal energy consumption of mobile devices in M time slots. Constraint C1 requires that the mobile device must finish its task in one time slot, no matter the executed task is cached or not. Constraint C2 requires that the total size of cached tasks can't exceed the caching capacity D. Constraints C3 denotes that the total size for edge execution is limited, because of the MEC server has limited computing power. Constraints C4 denotes the computing power allocated to the mobile devices can't exceed the computing power of MEC servers. Constraints C5 denotes the sum of allocated subchannels for mobile devices can't exceed the total number L. C6 denotes the caching policy is

binary variable, and C7 denotes the offloading ratio is a continuous variable between 0 and 1. Note that the resource is reallocated at the begging of each time slot and thus the corresponding constraints should be meet in any time slot.

It can be seen that P1 is a mixed discrete-continuous optimization problem with two types of variables, i.e. the discrete variable (caching policy) and continuous variable (CPU frequency, the number of subchannels and offloading ratio), which is NP-hard and very challenging for solving.

Remark: It is well known that caching policy is in general a long process, and depends on the task popularity, task size and computing complexity. Thus, we assume that a caching policy is used in several successive time slots, but the resource is allocated at each time slot. Besides, P1 shows that the task with greater popularity, larger size and higher complexity will be cached to minimize the objective function, which coincide with the real scene.

IV. PROBLEM SOLUTION

Since P1 is a mixed discrete-continuous problem and evidently non-convex due to the discrete caching policy a and the multiplicative terms of two variables in objective function and constraints. In this section, we propose a block coordinate descent and convex techniques based iterative optimization method. Specifically, we decompose P1 into two subproblems, one for the optimization of computation and communication resource by supposing the caching policy is given and the other for the optimization of caching policy by supposing the resource allocation scheme is given. These two optimization problems are solved alternately in each iteration until convergence condition is achieved.

A. OPTIMIZATION OF COMPUTATION AND COMMUNICATION RESOURCE

In this section, by supposing the caching policy is given, we formulate the first subproblem to optimize the computation and communication resources. Specifically, this subproblem jointly optimize CPU frequency, the number of subchannels and offloading ratio for each mobile device to get the optimal resource allocation scheme.

Given the initial caching policy $a = a^{(0)}$, the energy consumption in the m-th time slot can be obtained by the following function:

$$\begin{aligned} h(f,\theta,\alpha) &= \sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}(1-a_n^{(0)})E_{k,m} + a_n^{(0)}E_{k,m}^{edge} \\ &= \sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}(1-a_n^{(0)})((1-\alpha_{k,m})\kappa {f_k^{loc}}^2 c_n \\ &\quad + \alpha_{k,m}(P_k^{wait}\frac{c_n}{f_{k,m}^{edge}} + p_k\frac{d_n}{r_{k,m}})) \\ &\quad + a_n^{(0)}P_k^{wait}\frac{c_n}{f_{k,m}^{edge}} \end{aligned} \tag{14}$$

Next, by substituting $a_n^{(0)}$ into constraint C1 and then replacing it with two equal constraints C8 and C9. The problem of optimization of computation and communication resource can be described as follows:

$$\begin{aligned} P2:\ &\min_{f,\theta,\alpha}\sum_{m=1}^{M}h(f,\theta,\alpha) \\ &subject\ to: \\ &C8: a_n^{(0)}\frac{c_n}{f_{k,m}^{edge}} + (1-a_n^{(0)})(1-\alpha_{k,m}\frac{c_n}{f_k^{loc}}) \le T \\ &C9: a_n^{(0)}\frac{c_n}{f_{k,m}^{edge}} + (1-a_n^{(0)})\alpha_{k,m}(\frac{d_n}{r_{k,m}} + \frac{c_n}{f_{k,m}^{edge}}) \le T \\ &C3, C4, C5, C7 \end{aligned} \tag{15}$$

To avoid divided-by-zero exception, we first introduce two constant variable ε_1 and ε_2 and then define two auxiliary variables that $\beta_{k,m} = (\theta_{k,m}+\varepsilon_1)^{-1}$ and $\gamma_{k,m} = (f_{k,m}^{edge}+\varepsilon_2)^{-1}$. As a result, $h(\alpha, \beta, \gamma)$ can be converted to $h'(\alpha, \beta, \gamma)$ as:

$$\begin{aligned} h'(\alpha,\beta,\gamma) = &\sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}(1-a_n^{(0)})((1-\alpha_{k,m})\kappa {f_k^{loc}}^2 c_n \\ &+ \alpha_{k,m}(P_k^{wait}c_n\gamma_{k,m} + \frac{p_k d_n\beta_{k,m}}{B\log_2(1+\frac{p_k h_{k,m}}{\sigma^2})})) \\ &+ a_n^{(0)}P_k^{wait}c_n\gamma_{k,m} \end{aligned} \tag{16}$$

Furthermore, by substituting $\beta_{k,m}$,$\gamma_{k,m}$ into corresponding constraints in P2, we get the converted problem P3:

$$\begin{aligned} P3:\ &\min_{\alpha,\beta,\gamma}\sum_{m=1}^{M}h'(\alpha,\beta,\gamma) \\ &subject\ to: \\ &C10: a_n^{(0)}c_n\gamma_{k,m} + (1-a_n^{(0)})(1-\alpha_{k,m})\frac{c_n}{f_k^{loc}} \le T \\ &C11: a_n^{(0)}c_n\gamma_{k,m} + (1-a_n^{(0)})\alpha_{k,m}(\frac{d_n\beta_{k,m}}{B\log_2(1+\frac{p_k h_{k,m}}{\sigma^2})} \\ &\qquad + c_n\gamma_{k,m}) \le T \\ &C12: \sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}\frac{1}{\gamma_{k,m}} - \varepsilon_1 \le F \\ &C13: \sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}\frac{1}{\beta_{k,m}} - \varepsilon_2 \le L \quad \forall m\in\{1,2,\ldots,M\} \\ &C3, C7 \end{aligned} \tag{17}$$

Obviously, P3 is still a non-convex problem because of the second order terms of $\alpha_{k,m}\cdot\beta_{k,m}$ and $\alpha_{k,m}\cdot\gamma_{k,m}$. Thus, we adopt Reformulation-Linearization Technique (RLT) to linearize the second order terms [28], [29] that included in the objective function and corresponding constraints. Specifically, to eliminate the second order term $\alpha_{k,m}\cdot\beta_{k,m}$, we introduce an auxiliary variable $\mu_{k,m} = \alpha_{k,m}\cdot\beta_{k,m}$ where $0 \le \alpha_{k,m} \le 1$ and $0 \le \beta_{k,m} \le \frac{1}{L+\varepsilon_1}$. We can obtain the

RLT bound-factor product constraints for $\mu_{k,m}$ as:

$$\begin{cases} \{[\alpha_{k,m}-0]\cdot[\beta_{k,m}-\frac{1}{L+\varepsilon_1}]\}_{LS}, \\ \{[1-\alpha_{k,m}]\cdot[\beta_{k,m}-\frac{1}{L+\varepsilon_1}]\}_{LS}, \\ \{[\alpha_{k,m}-0]\cdot[\frac{1}{\varepsilon_1}\beta_{k,m}]\}_{LS}, \\ \{[1-\alpha_{k,m}]\cdot[\frac{1}{\varepsilon_1}\beta_{k,m}]\}_{LS} \end{cases} \tag{18}$$

where $\{\cdot\}_{LS}$ denotes the linearization setp under $\mu_{k,m}=\alpha_{m,k}\cdot\beta_{k,m}$. By substituting $\mu_{m,k}$ into (18), we can get:

$$\begin{cases} \mu_{k,m}-\frac{1}{L+\varepsilon_1}\geq 0, \\ \beta_{m,k}-\frac{1}{L+\varepsilon_1}-\mu_{k,m}-\frac{1}{L+\varepsilon_1}\alpha_{k,m}\geq 0, \\ \frac{1}{L+\varepsilon_1}\alpha_{k,m}-\mu_{k,m}\geq 0, \\ \frac{1}{L+\varepsilon_1}-\beta_{k,m}-\frac{1}{L+\varepsilon_1}\alpha_{k,m}+\mu_{k,m}\geq 0 \end{cases} \tag{19}$$

Similarly, for the second order term $\alpha_{m,k}\cdot\gamma_{m,k}$, we defined $\omega_{m,k}=\alpha_{m,k}\cdot\gamma_{m,k}$, where $0\leq\alpha_{k,m}\leq 1$ and $0\leq\gamma\leq\frac{1}{F+\varepsilon_2}$, the RLT bond-factor product constraints for $\omega_{m,k}$ are:

$$\begin{cases} \omega_{k,m}-\frac{1}{F+\varepsilon_2}\geq 0, \\ \gamma_{m,k}-\frac{1}{F+\varepsilon_2}-\omega_{k,m}-\frac{1}{F+\varepsilon_2}\alpha_{k,m}\geq 0, \\ \frac{1}{L+\varepsilon_2}\alpha_{k,m}-\omega_{k,m}\geq 0, \\ \frac{1}{L+\varepsilon_2}-\gamma_{k,m}-\frac{1}{L+\varepsilon_1}\alpha_{k,m}+\omega_{k,m}\geq 0 \end{cases} \tag{20}$$

Substituting $\mu_{k,m}$ and $\omega_{k,m}$ into the $h'(\alpha,\theta,\gamma)$, we can get $h''(\alpha,\beta,\gamma,\mu,\omega)$ defined as follows:

$$\begin{aligned} &h''(\alpha,\beta,\gamma,\mu,\omega) \\ &=\sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}(1-a_n^{(0)})((1-\alpha_{k,m})\kappa f_k^{loc^2}c_n \\ &\quad+P_k^{wait}c_n\omega_{m,k}+\frac{p_k d_n\mu_{k,m}}{B\log_2(1+\frac{p_k h_{k,m}}{\sigma^2})}) \\ &\quad+a_n^{(0)}P_k^{wait}c_n\gamma_{k,m} \end{aligned} \tag{21}$$

Furthermore, substitute $\mu_{k,m}$ and $\omega_{k,m}$ into constraint C11, we can get the problem P4 as:

$$\begin{aligned} P4:\ &\min_{\alpha,\beta,\gamma,\omega,\mu}\sum_{m=1}^{M}h''(\alpha,\beta,\gamma,\omega,\mu) \\ &subject\ to: \\ &C14: a_n^{(0)}c_n\gamma_{k,m}+(1-a_n^{(0)})(\frac{d_n\mu_{k,m}}{B\log_2(1+\frac{p_k h_{k,m}}{\sigma^2})} \\ &\qquad+c_n\omega_{k,m})\leq T \\ &C3, C7, C10, C12, C13 \\ &C15: (19) \\ &C16: (20) \end{aligned} \tag{22}$$

Obviously, P4 is a convex optimization problem and thus can be solved by the well-studied optimization techniques such as interior point method and Lagrange method. etc. After solving P4, we can get five $k\times m$ solution matrix $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\mu},\boldsymbol{\omega}$ by which we can get an suboptimal solution of P2.

B. OPTIMIZATION OF CACHING POLICY

With the obtained resource allocation scheme $(f^{(0)},\alpha^{(0)},\theta^{(0)})$ by initial $a^{(0)}$, we can reformulate problem P1 to a caching policy optimization problem, the total energy consumption in time slot m can be obtained by the following function with variable a:

$$g(a)=\sum_{n\in\mathcal{N}}\sum_{k\in S_n(m)}(1-a_n)E_{k,m}^{uncache}+a_nE_{k,m}^{cache}. \tag{23}$$

Considering the caching constraint in problem P1, the optimal caching policy for the given resource allocation scheme can be given by problem P5 as:

$$\begin{aligned} P5:\ &\min_{a}\sum_{m=1}^{M}g(a) \\ &subject\ to: \\ &C17: a_n\frac{c_n}{f_{k,m}^{edge(0)}}+(1-a_n)(1-\alpha_{k,m}^{(0)}\frac{c_n}{f_k^{loc}})\leq T \\ &C18: a_n\frac{c_n}{f_{k,m}^{edge(0)}}+(1-a_n)\alpha_{k,m}^{(0)} \\ &\qquad\times(\frac{d_n}{\theta_{k,m}^{(0)}B\log_2(1+\frac{p_k h_{k,m}}{\sigma^2})}c_n f_{k,m}^{edge(0)})\leq T \\ &C2, C6 \end{aligned} \tag{24}$$

P5 is a 0-1 integer programming, a straightforward method is to enumerate all the 2^N possible caching policy and find the optimal one that has the minimum objective value. However, the time complexity of the exhaustion method is $O(n!)$, it can be acceptable when N is smaller, but quickly becomes impossible to compute as N increase. It can be mainly used as a benchmark to evaluate the performance of actually used low-complexity algorithm. Consider the subsequent overall optimization, we adopt convex optimization based method to solve problem P5 in this section.

However, P5 is not a convex optimization problem because of the discrete variable a. Therefore, we first relax it by $0\leq a\leq 1$, then P5 is transformed to the problem P6 as:

$$\begin{aligned} P6:\ &\min_{a}\sum_{m=1}^{M}g(a) \\ &subject\ to: \\ &C19: a_n\in[0,1] \\ &C2, C17, C18 \end{aligned} \tag{25}$$

Obviously, P6 is a convex optimization problem, and thus can be solved by several well-studied method.

Algorithm 1 Block Coordinate Descent Based Overall Algorithm

1: Initialize the caching policy $\boldsymbol{a}^{(0)}$. Let $r = 0$.
2: **repeat**
3: Solve problem P4 for given $\{\boldsymbol{a}^{(r)}\}$, and denote the optimal solution as $\{\boldsymbol{\alpha}^{(r+1)}, \boldsymbol{\beta}^{(r+1)}, \boldsymbol{\gamma}^{(r+1)}, \boldsymbol{\mu}^{(r+1)}, \boldsymbol{\omega}^{(r+1)}\}$.
4: Solve problem P6 for given $\{\boldsymbol{\alpha}^{(r+1)}, \boldsymbol{\beta}^{(r+1)}, \boldsymbol{\gamma}^{(r+1)}, \boldsymbol{\mu}^{(r+1)}, \boldsymbol{\omega}^{(r+1)}, \boldsymbol{a}^{(r)}\}$, and denote the optimal solution as $\{\boldsymbol{a}^{(r+1)}\}$.
5: Update $r = r + 1$.
6: **until** The fractional increase of the objective value is below a threshold ϵ.

C. OVERALL ALGORITHM DESIGN

Based on the previous two sections, optimal caching policy and allocation scheme both can be obtained by optimizing one block of optimization variables while keeping the other variables fixed, what's more, two subproblems are both able to converted to the convex optimization problem. Therefore, we propose a block coordinate descent based alternately algorithm to solve the original problem P1.

Specifically, we first give an initial caching policy $a^{(0)}$. By given $a^{(0)}$, problem P1 is reformulated to the optimization problem of computation and communication resource and can be solved by solving converted problem P4 instead. Furthermore, the optimal resource allocation scheme obtained by previous step is used as input to get next optimal caching policy by solving P6. The iterative process will go on, alternately optimizing caching policy and allocation scheme in each iteration until the fractional increase of the objective values is below a threshold $\epsilon \geq 0$. By r denotes the iteration number, the details of the overall algorithm are summarized in Algorithm 1. Note that P2 and P6 are both convex optimization problem, and thus P2 and P6 construct a multi-convex problem, which has been proved to be convergence [30]. The solving process and transformation relationship from problem P1 to P6 can be described as Figure 2.

Since we relaxed the discrete variable a, we get a continuous value in the range of [0, 1]. But caching policy is a binary variable, representing caching or not (1 or 0). Therefore, we propose a simple and effective method to transform the obtained value to binary variable. Specifically, let $a_n = 1$ when the continuous variable is greater than 0.7 and $a_n = 0$ when it is less than 0.3. Besides, we add the value between 0.3 and 0.7, and then divide the sum value by 1 to get the task number that can be continuous cached. Following, cache the tasks with smallest size until reaching the obtained number. 0.3 and 0.7 in this method is determined by experiments. In fact, how to reconstruct the binary variable is still an open issue, some researches proposed their solutions according to their specific problems. We had also tried some existed solutions to reconstruct the binary variable in caching policy decision, but there still exists certain errors. Although proposed method also has errors, its effect is similar to the better methods we have tried and it almost has less time cost.

Note that although the solution obtained by proposed block coordinate descent iterative method is suboptimal, we proofed it effective by comparing with exhaustive method in the following section.

D. CACHING POLICY INITIALIZATION

The block coordinate descent based algorithm usually has the shortage that the iteration times and time cost largely depends on the initial value, a good initial method can certainly reduce iteration times and time cost. In this section, we propose a linear-weighted based initial method by considering the real characteristics of caching decision. By analyzing the objective function of P1 and combining the real caching scene, we see that the task popularity, task size and complexity are all the key factors that influence the caching decision. Therefore, we define the caching income function $u(n)$ as follows:

$$u(n) = w_1 \Phi(\sum_{i=1}^{M} |S_n(m)|) + w_2 \Phi(d_n) + w_3 \Phi(c_n), \quad (26)$$

where w_1, w_2, w_3 denotes the weight of task popularity, task size and complexity, respectively, and $w_1 + w_2 + w_3 = 1$. The weights are given by the principal component analysis (PCA) [33]. Due to the variables have different units and larger difference from each other, we normalize the variable. $\Phi(.)$ denotes the max-min normalization function defined as:

$$\Phi(x) = \frac{x - min(X)}{max(X) - min(X)}, \quad (27)$$

where x denotes the current value to be normalized and X represents the set of variables to be normalized, $min(X)$ and $max(X)$ denotes the minimum and maximum value of set X, respectively. For the normalization of task popularity, task popularity of task n is denoted by the total execution times of task n in M time slots (i.e.$\sum_{i=1}^{M} |S_n(m)|$), and the set to be normalized is $X = \{\sum_{i=1}^{M} |S_1(m)|, \sum_{i=1}^{M} |S_2(m)|, \ldots, \sum_{i=1}^{M} |S_N(m)|\}$, thus we get the normalized value $\Phi(\sum_{i=1}^{M} |S_n(m)|)$. Similarly, the normalized value of task size and complexity for each task can be obtained by this function. Then, we calculate the variance of each characters and determine the weight by variance ratio.

By this definition, we can get the income value $\phi_n = u(n)$ of each task n. Note that the income function is used to get the income when caching each task n, and thus larger function value ϕ_n means it has higher chance to be cached. Next, we obtain the income value of each task, constructing the set $S = \{\phi_1, \phi_2, \ldots, \phi_N\}$, and sort the value of the set in descent order (i.e. $\phi_1 \geq \phi_2 \geq \ldots \geq \phi_N$), the sorted set is denoted by S_{sort}. From the caching policy $a^{(0)} = \{0, 0, \ldots, 0\}$ on, we fetch the task in turn according to the order of S_{sort} and caching it (i.e. set the corresponding task decision $a_n = 1$) until the caching constraints C2 is violated. As a result, the $a^{(0)}$ is the initial caching policy.

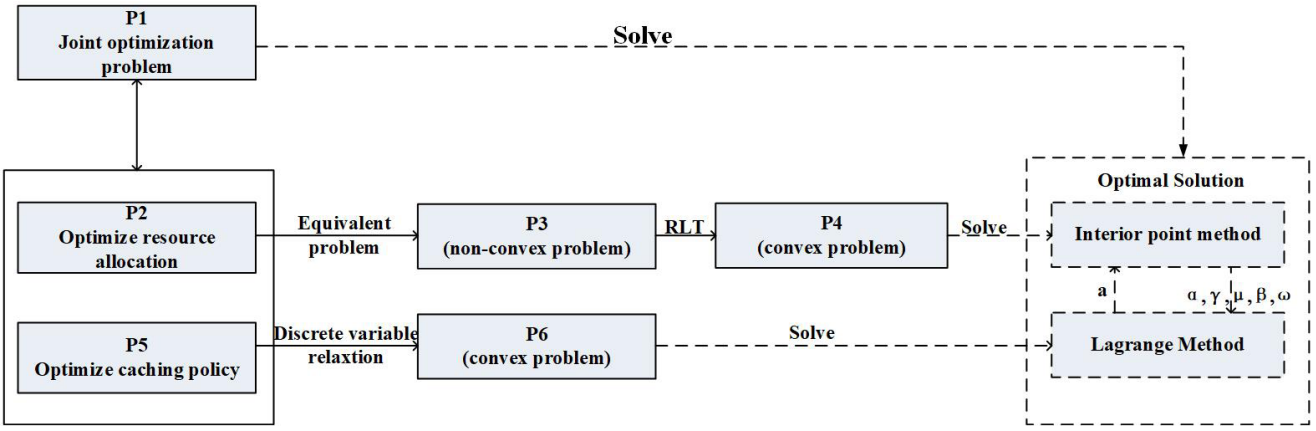


FIGURE 2. Process of problem transformation and solving.

V. SIMULATIONS

In this section, we design a simulation environment of proposed caching-enhanced MEC system to confirm the effectiveness of proposed joint optimization algorithm. We assume that the MEC system has $K = 20$ mobile devices and task list have $N = 10$ tasks. For computation model, we set the computation frequency of edge servers $F = 50GHz$, computation frequency for each mobile device f_k^{loc} takes random value in the range of $[0.5, 1]GHz$ and have the average value $0.7GHz$. For communication model, the bandwidth $B = 2MHz$, the channel number $L = 200$, the noise power $\sigma^2 = 10^{-8}$ and transmission power p_k is the random value in the range of $[0.6, 1.2]W$, the idle power $P_k^{wait} = 0.01W$, besides, we neglect the influence of distance on the channel power gain, and give $H = 10$ different channel states instead, i.e.$\mathcal{H} = \{1 * 10^{-7}, 2 * 10^{-7}, \ldots, 10 * 10^{-7}\}$. Note that the following simulations are conducted for 20 times and average the value to void the occasionality. The simulations are conducted on a common PC with $4 * 2.88GHz$ CPU, $8GB$ memory and coded with Python and its tool package for convex optimization CVXPY [31], [32].

A. EFFECT OF TASK CACHING

In this simulation, we aim at verifying the effect of introducing caching into MEC and caching policy obtained by proposed jointly optimizing algorithm of caching policy and resource allocation (J3C). For this purpose, we give the following three methods for comparison.

No Caching (NC): This method supposes that no caching enhanced MEC system is used, i.e. $a = \{0, 0, \ldots, 0\}$ in problem P2. This method just jointly optimizing the computation, communication resources and offloading ratio.

Random Caching (RC): This method firstly gives a caching policy randomly, then solves the corresponding resource allocation scheme by solving problem P2 with the given random caching policy.

Popularity Based Caching (PBC): This method firstly gives a caching policy by the order of task popularity, that is caching the task with high popularity until the caching capacity is insufficient. And then the resource allocation is computed with the obtained caching policy.

In this simulation, we observe the changes of energy consumption with the task size and task complexity increasing, where the average value of task size varies from 700 to 1600(KB) and the average task complexity varies from 300 to 1200 (Megacycles). From Figure. 3, we can see that the energy consumption increases with the task size and task complexity increasing and proposed J3C method still has lowest energy consumption than other caching methods in general. Obviously, NC method consumes much more energy than other methods which demonstrates that caching enhanced MEC system can reduce the energy consumption significantly. Besides, RC and PBC methods have little difference with J3C method and RC is inferior to PBC, because RC doesn't take any task character into account but PBC consider the task popularity. What's more, proposed J3C method not only considers several characters that influence the caching effect, but also jointly optimizes the caching policy and computation, communication resource, finding out the best allocation combination. From Figure.3 (a), we can see that the energy consumption increases quickly with the task size increasing, because larger tasks consume much energy on transmission and larger tasks lead to less task can be cached when caching capacity is fixed. By comparison, Figure.3(b) shows that the energy consumption increases more slowly, because the task number can be cached is fixed and only the computation complexity influences the energy consumption.

B. EFFECT OF RESOURCE ALLOCATION

In this simulation, we verify the effect of joint resource allocation, and thus we give contrast method as follows:

Caching and Local Execution (CLE): This method assumes that all the tasks that aren't cached is executed by local device, i.e. offloading ratio $\alpha = 0$. Since the task is executed by the local device, the subchannels are not used, i.e. $\theta = 0$. Besides, all the computing resources of edge servers are average allocated to mobile devices.

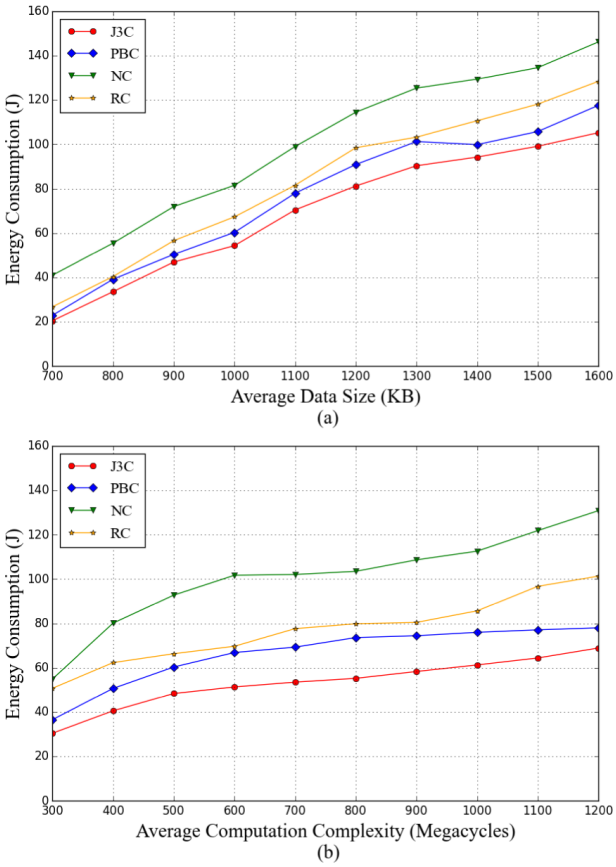


FIGURE 3. Comparison of energy consumption on different caching effect with the increasing of (a) task size (b) task complexity.

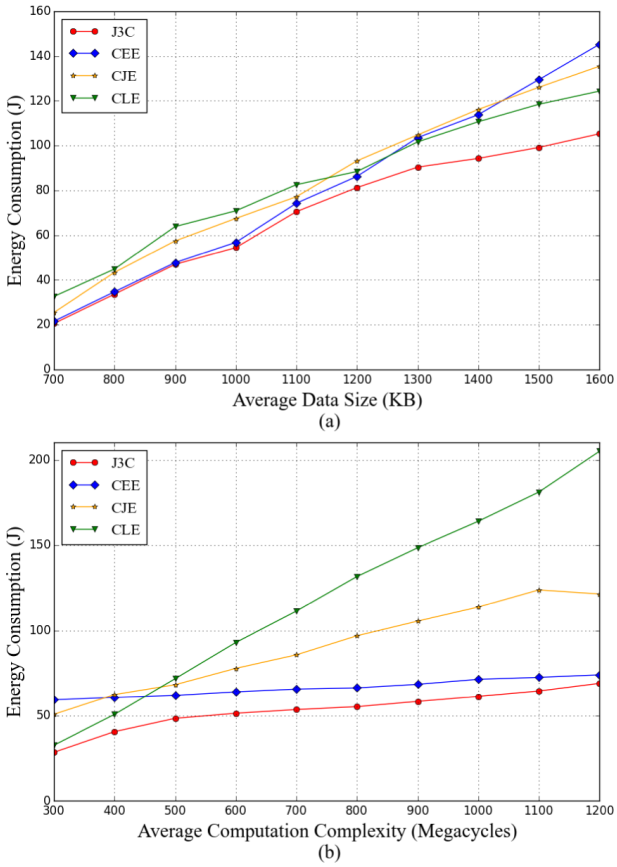


FIGURE 4. Comparison of energy consumption on different resource allocation scheme with the increasing of (a) task size (b) task complexity.

Caching and Edge Execution (CEE): This method assumes that all the tasks that aren't cached offloading to the edge server, i.e. offloading ratio $\alpha = 1$. Since the task is executed by the edge server, the subchannels are average allocated to the devices. Besides, all the computing resources of edge servers are average allocated to the mobile devices.

Caching and Joint Execution (CJE): This method assumes that half of the tasks that aren't cached offloading to the edge server, but the other executed on local devices, i.e. offloading ratio $\alpha = 0.5$. Since the task is executed by the edge server, the subchannels are average allocated to all the devices. Besides, all the computing resources average allocated to mobile devices in each time slot.

We can observe from the Figure.4 that the energy consumption increases with the task size and task complexity increasing in general, and proposed J3C method still has the smallest value. From Figure.4(a), we can observe that CEE method almost has the same optimal value with J3C method when the task size is smaller, but it is inferior to CLE method when the task size becomes larger and the gap between J3C and CLE tends to become larger. Besides, the CJE method almost has the optimal value located between CEE and CLE. This is due to caching is able to cache more tasks and offloading task consume less energy when the task size is smaller. Conversely, the cached tasks become less and offloading consume much energy when task size is larger. Besides, there exists some waste of computation and communication resources in CEE method. Therefore, CLE method is better than CEE when exceeding an critical value, about $1250KB$ in this simulation. From Figure.4(b), we can observe that although CEE method consumes much more energy than other methods at first, it increases quite slowly with the complexity increasing. Conversely, CLE method consumes less energy when the task complexity is lower, but has the highest increasing rate, exceeding the CEE method soon. This is because that the energy consumption for CLE method depends only on the computation complexity and is directly proportional to the square of complexity, having higher increasing rate. But the energy consumption of CEE method mainly depends on transmission, and thus computation complexity has less influence.

C. EFFECT OF PROPOSED ALGORITHM

In this simulation, we verify the effect of proposed algorithm to solve J3C problem. To this end,we give the other two algorithms for comparison on the time cost of algorithm and the effect of optimal value.

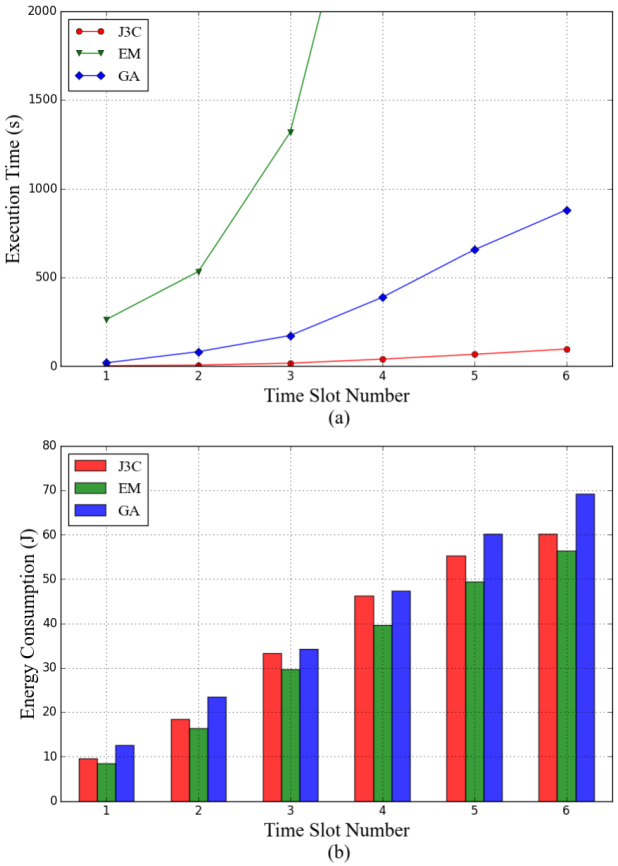


FIGURE 5. Algorithm performances and energy consumption versus the time slots number.

Exhaustive Method (EM): we discrete the continuous variable and try all the possible solution to find the optimal solution. This method is usually able to get the optimal solution but with high time complexity, and thus used as a benchmark.

Greedy Algorithm (GA): this method constantly update the caching policy according to the current optimal value until the termination condition is reached.

Figure 5(a) and 5(b) shows the execution time and energy consumption of three methods with the time slots increasing, respectively. From Figure 5, we can see that proposed method is able to get ideal effect with acceptable time consumption in general. With the number of time slot increasing, although still has the lowest energy consumption, EM method will soon unacceptable since the high time cost. GA algorithm has relative lower time complexity, but the solution has larger gap with EM. Look back forward proposed J3C method, the time consumption is smaller than other two methods and has the slowest increasing rate. Besides, the effect is better than GA and close to EM in general. This is because our initialization method reduces the iteration times and the convex problem is more efficient than local optimal method GA.

Although the decomposing based method is usually considered time-consuming, it is used quite widely because of its ability to solve the complexity problem. What's more, simulations show that we can usually get the sub-optimal solution with less iteration times in our problem.

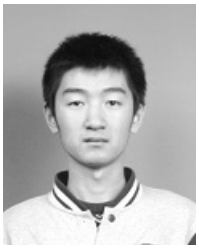
VI. CONCLUSION

In this paper, we redefined the task caching as caching the application program and related task data, where assumed the program had been cached on the MEC server and what caching policy need to decide is whether to caching the task data. Once the task data is cached, the task will be executed on the edge server without any data transmission, or it will be executed on the edge server and mobile device parallel. Then we proposed a caching enhanced MEC system that integrates task caching into MEC system to reduce the duplicate data transmission. Proposed system provides a good solution to solve the problem that offloading task with large size consume much time and energy. To further reduce energy consumption, we formulated a problem that jointly optimizes caching, computation and communication resources (J3C). Finally, we propose a block coordinate based iterative method to solve the formulated mixed integer non-convex optimization problem. The simulation results show that our proposed joint optimization method is not only superior to other single caching policies and resource allocation methods but be able to get an acceptable combination solution with lower time cost, proving proposed method has wide application prospect to further reduce the energy consumption of mobile devices in the mobile edge computing systems.

REFERENCES

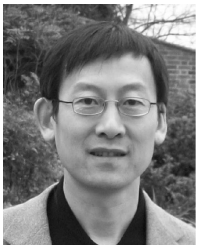
[1] F. Liu *et al.*, "Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14–22, Jun. 2013.
[2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013.
[3] S. Barbarossa, S. Sardellitti, and P. D. Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 45–55, Nov. 2014.
[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.
[5] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, Mar. 2017.
[6] A. Sengupta, R. Tandon, and O. Simeone, "Cloud RAN and edge caching: Fundamental performance trade-offs," in *Proc. IEEE 17th Int. Workshop*, Jul. 2016, pp. 1–5.
[7] F. Gabry, V. Bioglio, and I. Land, "On energy-efficient edge caching in heterogeneous networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3288–3298, Dec. 2016.
[8] M. Tao, E. Chen, H. Zhou, and W. Yu, "Content-centric sparse multicast beamforming for cache-enabled cloud RAN," *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 6118–6131, Sep. 2016.
[9] B. Zhou, Y. Cui, and M. Tao, "Stochastic content-centric multicast scheduling for cache-enabled heterogeneous cellular networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 6284–6297, Sep. 2016.
[10] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Joint allocation of computation and communication resources in multiuser mobile cloud computing," in *Proc. IEEE 14th Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2013, vol. 395, no. 6, pp. 26–30.
[11] 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.

[12] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.
[13] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11255–11268, 2017.
[14] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.
[15] 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.
[16] J. Gu, W. Wang, A. Huang, and H. Shan, "Proactive storage at caching-enable base stations in cellular networks," in *Proc. IEEE Pers. Indoor Mobile Radio Commun.*, Sep. 2013, pp. 1543–1547.
[17] B. Bai, L. Wang, Z. Han, W. Chen, and T. Svensson, "Caching based socially-aware D2D communications in wireless content delivery networks: A hypergraph framework," *IEEE Wireless Commun.*, vol. 23, no. 4, pp. 74–81, Aug. 2016.
[18] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini, "Temporal locality in today's content caching: Why it matters and how to model it," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 5, pp. 5–12, 2013.
[19] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1444–1462, Oct. 2014.
[20] A. Sengupta, S. D. Amuru, R. Tandon, and R. M. Buehrer, "Learning distributed caching strategies in small cell networks," in *Proc. Int. Symp. Wireless Commun. Syst.*, Aug. 2014, pp. 917–921.
[21] C. Wang, C. Liang, F. R. Yuv, Q. Chen, and L. Tang, "Joint computation offloading, resource allocation and content caching in cellular networks with mobile edge computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[22] X. Xu, J. Liu, and X. Tao, "Mobile edge computing enhanced adaptive bitrate video delivery with joint cache and radio resource allocation," *IEEE Access*, vol. 5, pp. 16406–16415, 2017.
[23] Y. Cui, W. He, C. Ni, C. Guo, and Z. Liu, "Energy-efficient resource allocation for cache-assisted mobile edge computing," in *Proc. IEEE 42nd Conf. Local Comput. Netw.*, Oct. 2017, pp. 640–648.
[24] C. Liang, Y. He, F. R. Yu, and N. Zhao, "Energy-efficient resource allocation in software-defined mobile networks with mobile edge computing and caching," in *Proc. INFOCOM WKSHPS*, May 2017, pp. 121–126.
[25] Y. Hao, M. Chen, L. Hu, M. S. Hossain, and A. Ghoniem, "Energy efficient task caching and offloading for mobile edge computing," *IEEE Access*, vol. 6, pp. 11365–11373, 2018.
[26] B. Perabathini, E. Bastug, M. Kountouris, M. Debbah, and A. Conte, "Caching at the edge: A green perspective for 5G networks," in *Proc. IEEE Int. Conf. Commun. Workshop*, Jun. 2015, pp. 2830–2835.
[27] 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.
[28] H. D. Sherali and W. P. Adams, *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, vol. 31, no. 8. Berlin, Germany: Springer, 1999, p. 790.
[29] 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.
[30] Y. Xu and W. Yin, "A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion," *SIAM J. Imag. Sci.*, vol. 6, no. 3, pp. 1758–1789, 2013.
[31] S. Diamond and S. Boyd, "CVXPY: A python-embedded modeling language for convex optimization," *J. Mach. Learn. Res.*, vol. 17, no. 83, pp. 1–5, 2016.
[32] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *J. Control Decis.*, vol. 5, no. 1, pp. 42–60, 2018.
[33] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433–459, 2010.



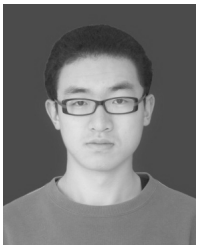
PENG LIU received the M.S. degree from the College of Computer Science and Technology, Jilin University, China, in 2016, where he is currently pursuing the Ph.D. degree. His main research interests include mobile cloud computing, mobile edge computing, edge caching, and the Internet of Things.

GAOCHAO XU received the B.S., M.S., and Ph.D. degrees from the College of Computer Science and Technology, Jilin University, China, in 1988, 1991, and 1995, respectively. He is currently a Professor and a Ph.D. Supervisor with the College of Computer Science and Technology, Jilin University. His main research interests include distributed system, grid computing, cloud computing, the Internet of Things, information security, software testing, and software reliability assessment. As a person-in-charge or a principal participant, he has finished more than 10 national, provincial, and ministerial-level research projects of China.



KUN YANG received the B.Sc. and M.Sc. degrees from the Computer Science Department, Jilin University, China, and the Ph.D. degree from the Department of Electronic and Electrical Engineering, University College London, U.K. He was with UCL on several European Union (EU) research projects for several years. In 2003, he joined the University of Essex, U.K., where he is currently a Chair Professor with the School of Computer Science and Electronic Engineering and is also leading the Network Convergence Laboratory. He manages research projects funded by various sources, such as UK EPSRC, EU FP7/H2020, and industries. He has authored over 80 journal papers. His main research interests include wireless networks, future Internet technologies, and mobile cloud computing. He is a Fellow of the IET. He serves on the editorial boards of the IEEE and the non-IEEE journals.

KEZHI WANG received the B.E. and M.E. degrees from the College of Automation, Chongqing University, China, in 2008 and 2011, respectively, and the Ph.D. degree from The University of Warwick, U.K., in 2015. He was a Senior Research Officer with the University of Essex, U.K. He is currently a Lecturer with the Department of Computer and Information Sciences, Northumbria University. His research interests include wireless communication, signal processing, and mobile cloud computing.



XIANGYU MENG received the Ph.D. degree from the College of Computer Science and Technology, Jilin University, Changchun, China, in 2017. He is currently a Postdoctoral Researcher with Jilin University. His research interests include data mining, network security, cloud computing, and mobile edge computing.