Energy-Efficient Resource Allocation for Application Including Dependent Tasks in Mobile Edge Computing

Yang Li[1], Gaochao Xu[1], Jiaqi Ge[1], Peng Liu[1] and Xiaodong Fu[1*]
[1] College of Computer Science and Technology, Jilin University
Changchun, Jilin 130012, P.R. China
[e-mail: e-mail: liyang17@mails.jlu.edu.cn]
*Corresponding author: Xiaodong Fu

Received March 10, 2019; revised February 24, 2020; revised March 25, 2020; accepted April 5, 2020; published June 30, 2020

Abstract

This paper studies a single-user Mobile Edge Computing (MEC) system where mobile device (MD) includes an application consisting of multiple computation components or tasks with dependencies. MD can offload part of each computation-intensive latency-sensitive task to the AP integrated with MEC server. In order to accomplish the application faultlessly, we calculate out the optimal task offloading strategy in a time-division manner for a predetermined execution order under the constraints of limited computation and communication resources. The problem is formulated as an optimization problem that can minimize the energy consumption of mobile device while satisfying the constraints of computation tasks and mobile device resources. The optimization problem is equivalently transformed into solving a nonlinear equation with a linear inequality constraint by leveraging the Lagrange Multiplier method. And the proposed dual Bi-Section Search algorithm Bi-JOTD can efficiently solve the nonlinear equation. In the outer Bi-Section Search, the proposed algorithm searches for the optimal Lagrangian multiplier variable between the lower and upper boundaries. The inner Bi-Section Search achieves the Lagrangian multiplier vector corresponding to a given variable receiving from the outer layer. Numerical results demonstrate that the proposed algorithm has significant performance improvement than other baselines. The novel scheme not only reduces the difficulty of problem solving, but also obtains less energy consumption and better performance.

Keywords: Mobile edge computing, computation offloading, task dependency, optimization problem, convex optimization

This work was supported in part by National Key R&D Plan of China under Grant No. 2017YFA0604500, in part by the Jilin Industrial Innovation Special Fund Project under Grant No. 2017C028-4, and in part by the Jilin Province Science and Technology Development Plan Project under Grant No. 20190201180JC.

1. Introduction

With generations of mobile devices launched by mobile device providers, users focus on implementing more complex application on their MDs. More and more emerging applications (i.e. online game, augmented reality, video optimization/acceleration, real-time monitoring, face recognition) are being integrated to form more powerful mobile application[1,2]. Those powerful mobile applications consist of multiple computation tasks with dependencies. In order to accomplish every computation task, the computation and communication resources of MD will be heavily occupied. However, MD's finite battery life and limited computation capacity pose significant challenges on accommodating the resource demand of those applications[3]. Fortunately, offloading computation tasks partly to Mobile Cloud Computing (MCC)[4] provides a promising technique to the elastic scaling of the capability of MDs. However, driven by the vision of 5G communications, the inherent limitation of MCC, i.e. the long transmission distance from MD to MCC, will lead to exceedingly long latency for mobile applications[5]. Moreover, there still exists many urgent issues to solve, such as increasing demand for high bandwidth, decreasing the energy consumption, and high quality of experience. Mobile Edge Computing (MEC)[6] can address those limitation by offloading computation tasks to the near-user MEC server instead of the remote cloud center.[7] indicated that MEC would play an significant role in 5G, the next generation mobile network. MEC[8,9] is regraded as a brilliant solution to overcome these difficulties.

By combining with partial computation offloading, the computation task can be divided into two parts, which are executed locally and on the MEC server. [10] deeply studied the computation offloading scheme with the two protocols TDMA and OFDMA in a multiuser Mobile Edge Computing Offloading system by joint optimization the offloading ration, computation and communication resources, and formulated the two computation offloading problem as a convex optimization problem and a mixed-integer problem, respectively. Chen et al.[11] studied a multiuser computational offloading problem in a MEC system under the constraints of communication and computation resources, and designed a distributed computation algorithm which could obtain the Nash equilibrium by using the game theoretic. Chen et al.[12] proposed a decentralized computation offloading algorithm which could achieve a Nash equilibrium in a multiple mobile devices MCC system. [13] proposed a computation offloading framework in a multiple wireless access points MEC system by optimizing the chosen edge server, the CPU frequency and computation and communication resources to trade off the energy consumption and execution time of mobile device including multiple independent tasks. [14,15] all carried out reaserch for the purpose of maximizing the computation rate, but the backgrounds of the proposed problem and the strategies and measures show were different. [14] jointly optimized the transmission power at the access point, the task computation mode(local computing or computation offloading) and time allocation to tasks, and derived the optimal solution by using the convex optimization techniques. In order to prolong the standy time of MD, wireless powered transfer(WPT) and MEC had been integrated together. [15] jointly optimized the computation mode selection(local computing or edge computing), the time allocated to wireless power transfer(WPT) and task offloading in a WPT-assisted multi-MD MEC system. And a joint optimization algorithm based on ADMM decomposition technique was proposed to tackle this problem. In [16], the proposed dynamic computation offloading algorithm achieved the multi-objective optimization by jointly optimizing the computation offloading ratio, the CPU frequency allocated to local execution, and the transmission power for computation offloading under the MEC system with an energy harvesting technique. In the following paper[17], MDs not only could offload their jobs to the MEC server, but also could request computation resource from others MD in the same MEC system. Meanwhile, all the MDs obtained energy from the AP by using WPT technique. [17] aimed at minimizing the whole AP's energy transmitted under the constraints of communication and computation resources, the deadline and the power transmission. [18,19] also gived a study on computation offloading. Although these papers are all about independent task with MD as shown previously, there is no dependency relationship among tasks. As the application becomes increasingly complex, multiple tasks, functions or applications are incorporated into an integrated application. Thus, an in-depth study of offloading computation tasks with dependencies is worthwhile.

Coarse-granularity based task offloading policy is used to provide a optimal solution for tasks with dependencies. [20] proposed a collaborative task computation algorithm to prolong the standby time of mobile devices for the application model in the linear topology. [21] provided a comprehensive computation offloading algorithm for computationally intense applications with multiple subtasks to determine which subtasks should be computed in the MD or cloud. [22] aimed to trade-off the energy consumption of MD and latency for application by finding the optimal assignments of tasks executed on local or remote devices. [23] achieved the trade-offs between the energy cost and latency by optimizing the power allocation and the offloading decisions for an application modeled as directed acyclic graph(DAG). [24,25] also could obtain an overall solution by designing a joint scheduling offloading policy for the application with sequential task dependencies. [26] divided the application into a non-offloadable task and multi-offloadable tasks, and a low complexity sub-optimal algorithm was proposed to decide which offloadable task should be transmitted to MEC server. For an application including multiple subtasks depended on each other, [27] proposed a heuristic algorithm based on particle swarm optimizer (PSO) to solve the 0-1 programming problem, where 0 represented local computing and 1 represented edge computing.

Though these studies improve the performance of mobile device by using computation offloading, most of those researches focus on the optimal scheduling order. Meanwhile, the transmission power and the CPU frequency allocated to tasks are fixed. Therefore, we make a study on the computation offloading in the mobile application which includes multiple computational intensive dependent tasks for a single MD MEC system, and propose an algorithm that minimizes the energy cost and improves performance by jointly optimizing the offloading ratio, CPU frequency, the transmission power and the transmission time. [28] compared the performance of offloading the dependent tasks from MD to a remote cloud center or an edge server and made the conclusion that offloading to an edge server is a promising technique in providing better performance than a remote cloud server. [29] took use of offloading strategy which migrated tasks in the task graph to the nearby wearable or mobile device through the available wireless communication interface such as Bluetooth or Wi-Fi. Experimental conclusions of [29] demonstrated that the scheduling policy could achieve the two objectives of extending battery lifetime and enhancing performance. At present, there are not many works related to task graph on MEC system, most of them are sub-task granularity and these works aim at different objective. As far as we know, there is no related work with the goal of minimizing energy consumption of MD by dividing the input data of all the subtasks on task graph bit by bit, which is able to guarantee the higher resource utilization and less energy consumption. Accordingly, this paper aims to minimize the energy consumption of task graph by jointly optimizing the resource allocation strategy.

In this paper, we study an single MD MEC system where MD executes a mobile application made up of multiple computation tasks. The objective of the paper is to minimize the energy consumption of MD with energy-saving resources allocation. In this paper, we model the complex mobile application consisting of multiple interdependent tasks as a DAG task-graph, and develop an energy-efficient computation offloading algorithm by joint optimization the communication and computation resources. Our main contributions are generalized as follows.

Firstly, in term of system model, most the computation offloading algorithms do not take into account the computation results feedback, and partial computation offloading for tasks with dependencies is task granularity rather than dividing the input data bit by bit. Therefore, we propose a system model of partial computation offloading for a application consisting of multiple tasks with dependencies, which divides the computation data by bitwise and integrates results feedback.

Secondly, in term of problem formulation, for tasks with dependencies, most studies are aimed at achieving the objective of saving energy or reducing latency by optimizing scheduling strategy with fixed transmission power and local computation capacity. However, this paper uses the DVFS technique optimize the CPU frequency, and achieves the optimal transmission power. The corresponding time slots for the local computing, uploading the offloading bits and computation results feedback are allocated with a time division mechanism. Therefore, the problem is formulated as an optimization problem that minimizes the MD's energy consumption with the constraints of the delay, the computation and communication capacity.

Finally, in term of the optimal solution, convex optimization techniques and the Lagrange Multiplier method are used to simplify the problem, and the original non-convex optimization problem is transformed into a univariate nonlinear equation, which accelerates the problem solving. Simulation results demonstrate that the proposed algorithm not only saves energy consumption, but also prolongs the standby time of MD.

The rest of this paper is organized as follows. In Section II, we expounds the system model. The problem is formulated and the algorithm for the problem is proposed in Section III and its performance evaluation and simulation results are analysed and shown in Section IV. The conclusion is given in Section V.

2. System Model

As shown in **Fig. 1**, we consider a basic two-node MEC system that consists of single MD and one AP node integrated with a MEC server. In order to reduce the impact of multiple wireless channel creation, frequent requests for the MEC server resources, etc. on task offloading, we conduct the study in the single MEC server scenatio. Both of the two nodes are equipped with one single antenna. This system provides the simplex mode. That is to say, the two processes, offloading input data or receiving computation results, can not be executed through the wireless channel at the same time. The MEC server not only provides the same execution environment but has more sufficient resources than the MD. Therefore, the MEC server can accomplish the computation tasks of application more efficient than the MD. The distance is defined as d between MD and AP, and the bandwidth is denoted as B.

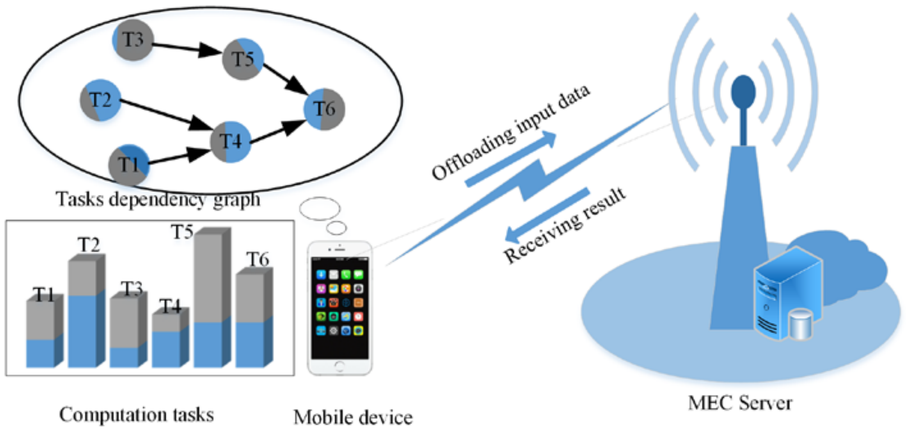

Fig. 1. A mobile-edge computing system with a mobile device and an AP

2.1 Computation Application Model

We assume that the MD has an application that is composed of N computation-intensive tasks (See **Fig. 1**, with an example where $N = 6$). Each task can be offloaded to MEC server by using partial computation offloading technique. This application model can be described by using Directed Acyclic Graph(DAG) $G = (\mathbb{V}, \mathbb{E})$ where the node v_j in the node set $\mathbb{V}$ denotes a task in the application, the edge (v_i, v_j) represents the dependency from task v_i to v_j and the edge set $\mathbb{E}$ of the DAG denotes the set of dependencies. Set_i represents the in-degree set of task i. Task i can not start executing until tasks belonging to Set_i are completed.To describe the parametric context of each task, a two-tuple (I_i, η_i) is defined, where $i = 1, ..., N$. Accordingly, I_i is the size of task i input data which includes the existing data l_i and the computation results transmitted from tasks in Set_i. According to [10,17,19,20], we learn that there is a linear relationship between the computation result and the input data for the computation task. η_i is the ratio of the output data to the input data for task i. Task i may need the computation results from related tasks if Set_i is not null.As such, the total input data size I_i is $l_i + \sum_{k \in Set_i} \eta_i l_i$ while Set_i is non-null. If Set_i is an empty set, I_i is equal to l_i. We use D, a N*N matrix, represent the task graph.

In this paper, we focus on the scenarios where the MD has finite computing capability. In order to accomplish the application, offloading parts of the input data to the MEC server is necessary. We assume that all tasks of the application are executed in a predetermined scheduling policy which results from Breadth-First Traversal(BFT). For example, we can get the scheduling policy $(1,2,3,4,5,6)$ in **Fig. 1**. And then, we realize our objective which minimizes the energy consumption of the MD. Meanwhile, with the given scheduling strategy, we assume that both local computing and computation offloading of the current task begin simultaneously, which ensures that its dependent tasks have completed correctly. λ_i is the ratio of the offloading data to the input data for task i, so we can get the constraint of λ_i:

$$0 \le \lambda_i \le 1, \forall i \in \{1, ..., N\}. \tag{C1}$$

where $\boldsymbol{\lambda} = \{\lambda_1, ..., \lambda_N\}$ denotes the offloading policy of the tasks graph.

2.2 Communication Model

For each computational intensive task $i \in \mathbb{V}$, its input bits I_i is split into two parts $\lambda_i I_i \geq 0$ and $(1-\lambda_i)I_i \geq 0$ bits, which denote the number of bits offloaded to the MEC server through AP and computed locally, respectively. MD receives the offloading computing results from the MEC server. We introduce the TDMA protocol into the system to avoid any interference among tasks. Because of MD and AP equipped with one antenna, the wireless channel can execute only one process during each time slot. As shown in Fig. 2, the protocol divides the whole time T into $2N$ time slots, and each task contains two time slots (t_i^u, t_i^d).

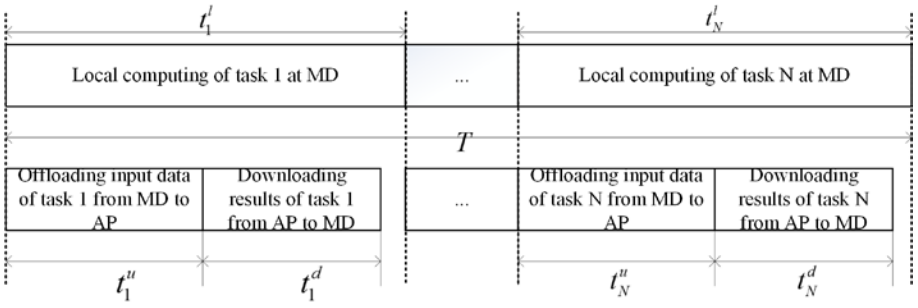
t_1^l
t_N^l
Local computing of task 1 at MD
...
Local computing of task N at MD
T
Offloading input data of task 1 from MD to AP
Downloading results of task 1 from AP to MD
...
Offloading input data of task N from MD to AP
Downloading results of task N from AP to MD
t_1^u
t_1^d
t_N^u
t_N^d

Fig. 2. The TDMA protocol for mutli-task computation offloading in an application

1) Computation Offloading from MD to the AP

In the t_i^u time slot, task i offloads $\lambda_i I_i$ input bits to the AP with the transmission power $P_i^t \geq 0$. We define h^2 as the channel power gain between MD and AP. Next, the achievable data rate(bits/sec) for uploading offloading input data from the MD to the AP is defined as

$$r_i^u = B\log_2(1+\frac{P_i^t h^2}{N_0}) \tag{1}$$

where N_0 means the additive white Gaussian noise at the receiver of the AP. For simplifying the computation, we define a function $g(x) = \frac{N_0}{h^2}(2^{\frac{x}{B}}-1), x>0$. According to (1), we can get the expression of P_i^t with the variable r_i^u. That is to say, $P_i^t = \frac{N_0}{h^2}(2^{\frac{r_i^u}{B}}-1)$. After we get the offloading bits $\lambda_i I_i$ and the uploading time t_i^u, r_i^u can be represented as $r_i^u = \frac{\lambda_i I_i}{t_i^u}$. So P_i^t can be re-defined as followed:

$$P_i^t = \frac{N_0}{h^2}(2^{\frac{I_i\lambda_i}{Bt_i^u}}-1) = g(\frac{I_i\lambda_i}{t_i^u}) \tag{2}$$

Based on the actual observation offered in [30], and the equation recommended by the EARTH project[31], we know that the uploading power P_i^u includes the transmission power P_i^t and an extra constant circuit power P_t^c caused by converting from digital signal to analog signal, packaging, and so on. Moreover, P_i^u is linear with P_i^t. Therefore, we give the following definition:

$$P_i^u = P_t^c + \kappa_t P_i^t \tag{3}$$

where κ_t denotes the linear growth and is a constant coefficient without unit. According to (3),the energy consumption of uploading data at MD in time slot t_i^u is expressed as

$$E_i^u = P_i^u t_i^u = (g(\frac{I_i \lambda_i}{t_i^u}) + \kappa_t P_i^t) t_i^u \tag{4}$$

2) Downloading the Computation Results from the AP to the MD

Considering that the MD needs to receive the computation results from AP, so the delay cannot be ignored. The computation results of task i are $\eta_i I_i$. And then, the achievable data rate for receiving computation results from the AP to the MD is expressed by

$$r_d = B\log_2(1+\frac{P_F h^2}{N_0}) \tag{5}$$

where P_F denotes the transmission power of the AP, which is a constant. Therefore, r_d is a constant. Meanwhile, we have known the amount of the computation results $\eta_i \lambda_i I_i$. Ultimately, combining (5), we can get the representation as follows:

$$t_i^d = \frac{\eta_i \lambda_i I_i}{r_d} \tag{6}$$

According to [32,33], the MD power consumption P_i^d increases linearly with the download data rate r_d. So we give the formula of P_i^d as follows.

$$P_i^d = P_d^c + \kappa_d r_d \tag{7}$$

where P_d^c is similar to the previous definition P_t^c. However, P_d^c is affected by converting from analog signal to digital signal, unpackaging, and so on. κ_d represents the linearly increasing coefficient. And then, we get the equation of calculating the energy consumed by receiving the computation results from AP in the time slot t_i^d. By merging (6) and (7), E_i^d is expressed as the following equation.

$$E_i^d = P_i^d t_i^d = (P_d^c + \kappa_d r_d)\frac{\eta_i \lambda_i I_i}{r_d} = (\frac{P_d^c}{r_d} + \kappa_d)\eta_i I_i \lambda_i \tag{8}$$

2.3 Computing Model

1) Local Computing:

$(1-\lambda_i)I_i$ input data of task i is executed on the mobile device(MD) within a duration t_i^l. C denotes the number of CPU cycles required for computing 1-bit of input data at MD, and f_i is the CPU frequency allocated to task i, i.e. $f_i = \frac{(1-\lambda_i)I_i C}{t_i^l}$. In practical application, the CPU frequency of MD has a maximum value, denoted by f_{max}. So f_i is capped by f_{max}, i.e.

$$\frac{(1-\lambda_i)I_i C}{t_i^l} \le f_{max}, \forall i \tag{C2}$$

Also, the computation latency constraint t_i^l for local computing of task i must be satisfied as follows:

$$t_i^l \geq 0. \tag{9}$$

Then, the whole local computation tasks must be accomplished before the deadline, i.e.

$$\sum_{i=1}^{N} t_i^l \leq T \tag{C3}$$

The energy consumption of task i for local computing is

$$E_i^l = \kappa_l f_i^3 t_i^l = \kappa_l \frac{(1-\lambda_i)^3 I_i^3 C^3}{(t_i^l)^2}, \forall i \tag{10}$$

where κ_l means the effective capacitance coefficient which has relation with the chip structure of MD.

According to **Fig. 2**, for task i, the time that MD starts to transmit input data through wireless channel is simultaneous with starting local execution. Besides, according to (10), the bigger t_i^l is, the smaller E_i^l is. So we can infer that CPU is not in idle state between the end of the previous task and the beginning of the next task, which means that local execution lasts the whole time period t_i^l. Consequently, the wireless channel occupation time and local computation time should meet the following time constraint for task i.

$$t_i^u + t_i^d \leq t_i^l, \forall i. \tag{11}$$

Replacing t_i^d with (6), we can rewrite the former time constraint as below.

$$t_i^u + \frac{\eta_i \lambda_i I_i}{r_d} \leq t_i^l, \forall i. \tag{C4}$$

2) Edge-Computation: We assume that the network and computational resources of the MEC server are infinite. Owing to the sufficient computation capability of the MEC server, the delay of receiving offloading data, the time spending on accomplishing the offloaded computation task, and the delay of sending computation results are relatively small and negligible at the MEC server. Therefore,we assume that MD immediately receives computation result from the AP when the offloading data transmission is completed. Since the objective is to minimize the energy consumption of mobile device, the energy consumption of MEC server is negligible.

3. Problem Formulation and Optimal Solution

In this section, we propose a feasible computation offloading scheme to the application modeled as task graph. In order to minimize the energy consumption of MD, the CPU frequency and the transmission power are optimized for local computing and computation offloading, respectively. The solution process of the optimal computation offloading scheme is shown below.

3.1 Problem Formulation

Under the system model above mentioned, we give a trade-off model between the energy consumption of local computing and the energy cost of communication among the AP and the MD to obtain the total minimum energy consumption of the MD. Meanwhile, the trade-off model must satisfy the delay constraint, the limited computation and communication resources. From (4), (8) and (10), we can come to the conclusion that all the energy consumption of MD is impacted by local computing, computation offloading and receiving computation results. Learning from the former section, the local energy consumption is influenced by the locally executed bits $(1-\lambda_i)I_i$ and its executed time t_i^l, and the energy consumed by communication between the AP and the MD is impacted by the time spending on uploading the offloading bits t_i^u and the offloading bits $\lambda_i I_i$. Then, Let $\boldsymbol{\lambda} = \{\lambda_1,...,\lambda_N\}$, $\boldsymbol{t}^u = \{t_1^u,...,t_N^u\}$, $\boldsymbol{t}^l = \{t_1^l,...,t_N^l\}$, and these three vectors determine how much energy the application will consume. Mathematically, the delay-constrained energy minimization optimization problem is formulated as

$$\begin{aligned}\mathcal{P}:\ &\min_{(\boldsymbol{\lambda},\boldsymbol{t}^u,\boldsymbol{t}^l)} \sum_{i=1}^{N} E_i^u + E_i^d + E_i^l \\ &s.t.\, C1, C2, C3, C4 \\ &C5: t_i^u, t_i^l > 0 \qquad \forall i\end{aligned}$$

Now we substitute all related equations into problem $\mathcal{P}$, and $\mathcal{P}$ is represented as follows.

$$\begin{aligned}\mathcal{P}_1:\ &\min_{(\boldsymbol{\lambda},\boldsymbol{t}^u,\boldsymbol{t}^l)} \sum_{i=1}^{N} t_i^u\Big(P_t^c + \kappa_t \frac{N_0}{h^2}\big(e^{\frac{I_i \ln 2\, \lambda_i}{B t_i^u}} - 1\big)\Big) + \Big(\frac{P_d^c}{r_d} + \kappa_d\Big)\eta I_i \lambda_i + \kappa_l \frac{((1-\lambda_i)I_i C)^3}{(t_i^l)^2} \\ &s.t.\, C1: 0 \le \lambda_i \le 1 \qquad \forall i \\ &C2: \frac{(1-\lambda_i)I_i C}{t_i^l} \le f_{max} \quad \forall i \\ &C3: \sum_{i=1}^{N} t_i^l \le T \\ &C4: t_i^u + \frac{\eta I_i \lambda_i}{r_d} \le t_i^l \qquad \forall i \\ &C5: t_i^u, t_i^l > 0 \qquad \forall i\end{aligned} \tag{12}$$

In Problem $\mathcal{P}_1$, C1 denotes the constraint of the offloading ratio, C2 is the maximum CPU frequency constraint for each task of the application, and C3 is the deadline constraint. C4 guarantees that the time cost by local computing must be greater than or equal to the time taken by the data transmission and receiving computation results for task i. Note that $\mathcal{P}_1$ is non-convex in general because of the coupling of λ_i and t_i^l in C2. Therefore, we need convert $\mathcal{P}_1$ into a convex problem[34] with some measures, and then give the optimal solution.

3.2 Optimal Solution To $\mathcal{P}_1$

This subsection gives the optimal solution of $\mathcal{P}_1$. To achieve this goal, firstly, we transform C2 into a convex constraint by multiplying both sides of inequality C2 with t_i^l. The constraint C2 is converted to $C2'$.

$$(1-\lambda_i)I_i C - f_{max} t_i^l \le 0, \forall i. \tag{$C2'$}$$

Then, $\mathcal{P}_1$ is reformulated as follows.

$$\begin{aligned}\mathcal{P}_2: &\min_{(\lambda,t^u,t^l)} \sum_{i=1}^{N} t_i^u \Big(P_t^c + \kappa_t \frac{N_0}{h^2}\big(e^{\frac{I_i \ln 2\, \lambda_i}{B\, t_i^u}} - 1\big)\Big) + \Big(\frac{P_d^c}{r_d} + \kappa_d\Big)\eta I_i \lambda_i + \kappa_l \frac{((1-\lambda_i) I_i C)^3}{(t_i^l)^2} \\ &s.t.\ C1, C3, C4, C5 \qquad (13) \\ &C2': (1-\lambda_i) I_i C - f_{max} t_i^l \le 0\ \forall i\end{aligned}$$

Meanwhile, the function $\frac{(1-\lambda_i)^3}{(t_i^l)^2}$ is convex with respect to $0<\lambda_i<1$ and $t_i^l>0$ [35], and hence the equation $\frac{\kappa_l((1-\lambda_i)I_i C)^3}{(t_i^l)^2}$ in the objective function is jointly convex with respect to $0<\lambda_i<1$ and $0<t_i^l<T$. Moreover, the following lemma demonstrates the evidence that Problem $\mathcal{P}_2$ is a convex problem.

Lemma 1 Problem $\mathcal{P}_2$ is a convex optimization problem.

Proof: Because $g(x)$ is an exponential function and a convex function, its perspective function[13], $t_i^u g(\frac{I_i\lambda_i}{t_i^u})$ is still convex. Also, $\frac{\kappa_l((1-\lambda_i)I_i C)^3}{(t_i^l)^2}$ is jointly convex function[35]. Thus, the objective function which is the summation of a set of convex functions, holds the convexity. All constraints are the linear convex function. Consequently, we get the result of Lemma 1.

Assume that Problem $\mathcal{P}_2$ is feasible, we use the Lagrange Multiplier method to simply the problem solving. We define $\alpha_i \ge 0, \beta_i \ge 0, \gamma_i \ge 0$ as the Lagrangian multipliers for C1, $C2'$, C4, respectively. μ is the Lagrange multiplier for C3. Therefore, the partial Lagrangian function of $\mathcal{P}_2$ is given by

$$\begin{aligned}\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{t}^u, \boldsymbol{t}^l, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \mu) = &\sum_{i=1}^{N} t_i^u \Big(P_t^c + \frac{\kappa_t N_0}{h^2}\big(e^{\frac{I_i \ln 2\, \lambda_i}{B\, t_i^u}} - 1\big)\Big) + \Big(\frac{P_d^c}{r_d} + \kappa_d\Big)\eta I_i \lambda_i + \kappa_l \frac{((1-\lambda_i) I_i C)^3}{(t_i^l)^2} \\ &+ \sum_{i=1}^{N} \alpha_i(\lambda_i - 1) + \sum_{i=1}^{N} \beta_i((1-\lambda_i)I_i C - f_{max} t_i^l) + \sum_{i=1}^{N} \gamma_i\Big(t_i^u + \frac{\eta_i I_i \lambda_i}{r_d} - t_i^l\Big) \qquad (14) \\ &+ \mu\Big(\sum_{i=1}^{N} t_i^l - T\Big)\end{aligned}$$

Let $\{\lambda_i^*, t_i^{u*}, t_i^{l*}\}$ denote the optimal solution of $\mathcal{P}_2$. The optimal solution always satisfies all constraints and has the minimum energy consumption of MD. By applying KKT conditions, the following necessary and sufficient conditions must be true.

$$\frac{\partial \mathcal{L}}{\partial \lambda_i^*} = \frac{\kappa_t N_0 I_i \ln 2}{B h^2} e^{\frac{I_i \ln 2 \lambda_i^*}{B t_i^{u*}}} + \frac{(P_d^c + \kappa_d r_d)\eta_i I_i}{r_d} - \frac{3\kappa_l (1-\lambda_i^*)^2 I_i^3 C^3}{(t_i^{l*})^2} + \alpha_i^* - \beta_i^* I_i C + \gamma_i^* \frac{\eta_i I_i}{r_d} = 0 \qquad (a)$$

$$\frac{\partial \mathcal{L}}{\partial t_i^{u*}} = P_t^c + \kappa_t \frac{N_0}{h^2}\big(e^{\frac{I_i \ln 2\cdot \lambda_i^*}{B\cdot t_i^{u*}}} - 1\big) - \kappa_t \frac{n_0}{h^2} \frac{I_i \ln 2 \cdot \lambda_i^*}{B \cdot t_i^{u*}} e^{\frac{I_i \ln 2\cdot \lambda_i^*}{B\cdot t_i^{u*}}} + \gamma_i^* = 0 \qquad (b)$$

$$\frac{\partial \mathcal{L}}{\partial t_i^{l*}} = -\frac{2\kappa_l (1-\lambda_i^*)^3 I_i^3 C^3}{(t_i^{l*})^3} - \beta_i^* f_{max} - \gamma_i^* + \mu^* = 0 \qquad (c)$$

$$\alpha_i^*(\lambda_i^* - 1) = 0, \forall i \qquad (d)$$

$$\beta_i^*((1-\lambda_i^*)I_iC - f_{max}t_i^{l*}) = 0, \forall i \tag{e}$$

$$\gamma_i^*(t_i^{u*} + \frac{\eta_i I_i \lambda_i^*}{r_d} - t_i^{l*}) = 0, \forall i \tag{f}$$

$$\mu^*(\sum_{i=1}^{N} t_i^{l*} - T) = 0 \tag{g}$$

According to (b), we can get Lemma 2 which describes the relations among $\{\lambda_i^*, t_i^{u*}, \gamma_i^*\}$.

Lemma 2 The optimal $\{\lambda_i^*, t_i^{u*}, \gamma_i^*\}$ satisfies

$$\frac{\lambda_i^*}{t_i^{u*}} = \frac{B}{I_i \ln 2}(1 + W(\frac{-1}{e} + \frac{(P_t^c + \gamma_i)h^2}{\kappa_t e N_0}))\ \forall i, \tag{15}$$

where $W(x)$ defines the Lambert-W function, i.e. $x = W(x)e^{W(x)}$.

Proof: With the help of (b), we have $(\frac{I_i \ln 2 \cdot \lambda_i^*}{B \cdot t_i^{u*}} - 1)e^{(\frac{I_i \ln 2 \cdot \lambda_i^*}{B \cdot t_i^{u*}} - 1)} = \frac{-1}{e} + \frac{(P_t^c + \gamma_i^*)h^2}{\kappa_t e N_0}$. According to the Lambert-W function, i.e. $x = W(x)e^{W(x)}$, which is the inverse function of $f(x) = xe^x$ .We can straightforwardly infer that $(\frac{I_i \ln 2 \cdot \lambda_i^*}{B \cdot t_i^{u*}} - 1) = W(\frac{-1}{e} + \frac{(P_t^c + \gamma_i^*)h^2}{\kappa_t e N_0})$, which leads to the result in Lemma 2 with some simple operations.

With some operations on (c), we can get the following expression. $\mu^* = \frac{2\kappa_l(1-\lambda_i^*)^3 I_i^3 C^3}{(t_i^{l*})^3} + \beta_i^* f_{max} + \gamma_i^*,\ \ \forall i$. Therefore, we can get the result $\mu^* > 0$. Combining with (g), the summation of t_i^{l*} is identically equal to T, i.e.

$$\sum_{i=1}^{N} t_i^{l*} = T \tag{16}$$

which conforms to the monotonically decreasing property of the objective function respect to t_i^l. According to (a), (c), (d), we can get the following Lemma 3. In order to reduce the complexity of interpretation, we firstly define

$$\varphi_i(\gamma_i) = 1 + W(\frac{-1}{e} + \frac{(P_t^c + \gamma_i)h^2}{\kappa_t e N_0}). \tag{17}$$

Lemma 3 The optimal $\{\lambda_i^*, t_i^{l*}, \gamma_i^*\}$ satisfies

$$\frac{1-\lambda_i^*}{t_i^{l*}} = (\frac{\frac{\kappa_t N_0 \ln 2}{Bh^2}e^{\varphi_i(\gamma_i^*)} + \frac{\eta_i \gamma_i^*}{r_d} + (P_d^c + \kappa_d r_d)\frac{\eta_i}{r_d}}{3\kappa_l I_i^2 C^3})^{\frac{1}{2}} \tag{18}$$

The ratio is influenced by the i-th task parameters of $\{\eta_i, I_i\}$ and the lagrangian multiplier γ_i.

Proof: At first, according to (d), because not all the input data is transmitted to the MEC server, we obtain $\alpha_i^* = 0$. The result makes sure that the local energy cost is not the maximum. And then, to ensure (e) true, since the CPU frequency f_i allocated to task i always less than f_{max}, we obtain $\beta_i^* = 0$. Plugging $\beta_i^* = 0$ into (c), and making some simple manipulations, we obtain the following result.

$$\mu^* = \frac{2\kappa_l (1-\lambda_i^*)^3 I_i^3 C^3}{(t_i^{l*})^3} + \gamma_i^*, \forall i. \tag{19}$$

At last, we substitute (17), $\alpha_i^* = 0, \beta_i^* = 0$ into (a). The Lemma 3 is inferred.

We know $f_i = \frac{(1-\lambda_i) I_i C}{t_i^l} \le f_{max}$. Meanwhile, plugging (18) into f_i, we obtain an inequality about γ_i^*. We introduce the function $F_i(\gamma_i^*)$ for replacing the ratio $\frac{1-\lambda_i^*}{t_i^{l*}}$, i.e. $F_i(\gamma_i^*) = \frac{1-\lambda_i^*}{t_i^{l*}}$. Correspondingly, the range of γ_i^* can be described as Proposition 1.

Proposition 1 $F_i(\gamma_i^*)$ is a monotonically increasing function when $\gamma_i^* > 0$. The one and only $\gamma_i > 0$ satisfies $F_i(\gamma_i^*) = f_{max}$ existed. The value is defined as $\gamma_i^{(0)}$, and $\gamma_i^* \le \gamma_i^{(0)}$.

Proof: $e^{\varphi_i(\gamma_i)}$ is an increasing function with $\gamma_i \ge 0$ because $\varphi_i(\gamma_i)$ is an increasing function with $\gamma_i \ge 0$. Meanwhile, $\frac{\eta_i}{r_d}\gamma_i$ is a linear function. Because $F_i(\gamma_i)$ is the square root of the sum of an increasing function, we infer that $F_i(\gamma_i)$ is an increasing function with $\gamma_i \ge 0$. $\gamma_i \to +\infty$ and $F_i(\gamma_i) \to +\infty$. So we can get the unique $\gamma_i^{(0)}$ by using the Bi-Section Search method. The Proposition 1 is proved.

$$\mu^* = \gamma_i^* + 2\kappa_l (F_i(\gamma_i^*))^3. \tag{20}$$

Now, substituting (18) into (19), we get (20). Learning from (18), we acknowledge that μ is associated with γ_i^*, η_i, and the performance parameters of SD. Although the ratio of (18) changes over η_i, I_i, γ_i^*, the lagrangian multiplier μ is fixed. Meanwhile, μ is a scalar rather than a vector. Therefore, the lagrangian multiplier γ_i^* should satisfy $\gamma_i^* > 0$**.** In order to meet the condition (f), we get the result $t_i^{u*} + \frac{\eta_i I_i \lambda_i^*}{r_d} = t_i^{l*}$ because of $\gamma_i^* > 0$. Accordingly, we obtain Lemma 4 by using the above derivation results, which denotes that the local execution time t_i^{l*} can be expressed as a function with a single variable lagrangian multiplier μ^*.

Lemma 4 The optimal t_i^{l*} can be expressed as

$$t_i^{l*} = Q_i(\mu^*) = \frac{1}{\frac{r_d}{I_i \eta_i} - \frac{r_d^2 \ln 2}{\eta_i (I_i r_d \ln 2 + B I_i \eta_i \varphi_i(\psi_i^{-1}(\mu^*)))} + \frac{(\mu^* - \psi_i^{-1}(\mu^*))^{\frac{1}{3}}}{I_i C (2\kappa_l)^{\frac{1}{3}}}}, \tag{21}$$

$Q_i(\mu)$ is monotonically decreasing with $\mu^l \le \mu^* \le \mu^r$. If $\sum Q_i(\mu^r) > T$ or $\sum Q_i(\mu^l) < T$, $\boldsymbol{\lambda}$, $\boldsymbol{t}^l$, and $\boldsymbol{t}^u$ do not exist. If $T < \sum Q_i(\mu^l)$ and $\sum Q_i(\mu^r) < T$, $\boldsymbol{\lambda}$, $\boldsymbol{t}^l$, and $\boldsymbol{t}^u$ are unique.

Proof: Firstly, $\mu = \psi_i(\gamma_i)$ is introduced to simplify the derivation and computation, i.e. $\psi_i(\gamma_i) = \gamma_i + 2\kappa_l F_i^3(\gamma_i)$. $\psi_i'(\gamma_i) = 1 + 6\kappa_l F_i^2(\gamma_i) F_i'(\gamma_i)$ denotes its derivation. Learning from Proposition 1, we can get $\psi_i'(\gamma_i) > 1$ because $F_i(\gamma_i)$ is a monotonically increasing function $F_i'(\gamma_i) > 0$. Consequently, $\psi_i(\gamma_i)$ is a monotonically increasing function with $\gamma_i > 0$. Given the value of μ, we can get a unique γ_i, and vice versa. The inverse function of $\psi_i(\gamma_i)$ is denoted as $\gamma_i = \psi_i^{-1}(\mu)$, and $F_i(\gamma_i)$ can be rewritten as $F_i(\gamma_i) = (\frac{\psi_i(\gamma_i) - \gamma_i}{2\kappa_l})^{\frac{1}{3}}$. As we all know, the derivative of the original function is equal to the inverse of the derivative of its inverse function in calculus. Hence, $0 < (\psi_i^{-1})'(\mu) = \frac{1}{\psi_i'(\gamma_i)} < 1$, where $\mu = \psi_i(\gamma_i)$. Learning from Lemma 2 and Lemma 3, we can get the expression of t_i^{l*} by solving the simultaneous equation (22). We use $\psi_i^{-1}(\mu)$ replace the Lagrangian multiplier γ_i in (23), and we can use a function $Q_i(\mu^*)$ only associated with μ^* represent t_i^{l*}, which is (21).

$$\begin{cases} t_i^{u*} + \frac{\eta_i I_i \lambda_i^*}{r_d} = t_i^{l*} \\ (15), (18) \end{cases} \tag{22}$$

$$t_i^{l*} = \frac{1}{\frac{r_d}{I_i \eta_i} - \frac{r_d^2 \ln 2}{I_i \eta_i (r_d \ln 2 + B\eta_i \varphi_i(\gamma_i^*))} + \frac{F_i(\gamma_i^*)}{I_i C}} \tag{23}$$

Secondly, based on the property of Lambert-W function $\varphi_i' > 0$, and $0 < (\psi_i^{-1})' < 1$, the derived function $Q_i'(\mu) < 0$, and $Q_i(\mu)$ is μ's rigid monotony decrease by degrees function.

$$Q_i'(\mu) = \frac{\frac{BI_i r_d^2 \ln 2 \varphi_i'(\psi_i^{-1}(\mu))(\psi_i^{-1}(\mu))'}{(I_i r \ln 2 + BI_i \eta_i \varphi(\psi_i^{-1}(\mu)))^2} + \frac{1}{3I_i C(2\kappa_l)^{\frac{1}{3}}} \cdot \frac{1 - (\psi_i^{-1}(\mu))'}{(\mu - \psi_i^{-1}(\mu))^{\frac{2}{3}}}}{-(\frac{r}{I_i \eta_i} - \frac{r_d^2 \ln 2}{\eta_i (I_i r_d \ln 2 + BI_i \eta_i \varphi(\psi_i^{-1}(\mu)))} + \frac{(\mu - \psi_i^{-1}(\mu))^{\frac{1}{3}}}{I_i C(2\kappa_l)^{\frac{1}{3}}})^2} < 0 \tag{24}$$

Thirdly, we get the interval range of γ_i^* as $0 \le \gamma_i^* \le \gamma_i^{(0)}$ according to Proposition 1. And, (20) indicates that μ is a monotonically increasing function with γ_i. We define $\mu_l^{(0)} = \max\{\psi_i(0)\}$ and $\mu_r^{(0)} = \min\{\psi_i(\gamma_i^{(0)})\}$, and the numeric zone of μ^* is displayed as follows.

$$\mu_l^{(0)} \le \mu^* \le \mu_r^{(0)}. \tag{25}$$

Meanwhile, according to (21), t_i^{l*} is monotonically decreasing with $\mu^* \ge 0$. What's more, $0 \le t_i^{l*} \le T$. By solving $t_i^{l*} = T$, we can get a unique value of μ, defined as $\mu_l^{(1)}$. Let $\mu^l = max\left(\mu_l^{(0)}, \mu_l^{(1)}\right)$ and $\mu^r = \mu_r^{(0)}$. The restriction of μ^* is rewritten as

$$\mu^l \le \mu^* \le \mu^r. \tag{26}$$

By using the well studied methods, i.e. Bi-Section Search, Newton Method, solve $\mu = \psi_i(\gamma_i), \forall i$, the numeric zone of $\gamma_i, \forall i$ can be achieved with the given lower and upper limits of μ.

Ultimately, by judging the value of $\sum Q_i(\mu)$, i.e. $\sum t_i^l$, we can determine whether problem $\mathcal{P}_2$ exists the optimal solution $\boldsymbol{\lambda}^*$, $\boldsymbol{t}^{l*}$, and $\boldsymbol{t}^{u*}$ or not. If $\sum Q_i(\mu^r) > T$ or $\sum Q_i(\mu^l) < T$, it does not exist the optimal solution. If $T < \sum Q_i(\mu^l)$ and $\sum Q_i(\mu^r) < T$, the unique optimal solution can be achieved by solving (16). Lemma 4 is proved now.

Accordingly, problem $\mathcal{P}_2$, a multiple variables non-linear optimization problem with multi-constraint, is equivalently transferred to a non-linear equation with a linear inequality constraint. The nonlinear equation is expressed as follows:

$$\begin{aligned} &\mathcal{P}_3 : \sum_{i=1}^{N} Q_i(\mu) = T \\ &s.t.\ \mu^l \le \mu \le \mu^r \end{aligned} \tag{27}$$

Since t_i^l is a monotonic decreasing function respect to μ, $\sum t_i^l$ inherits the descending characteristic. Therefore, there exists a unique μ to $\mathcal{P}_3$. Then, we propose our algorithm Bi-JOTD to calculate the optimal solution to $\mathcal{P}_3$. The solution is the optimal offloading strategy for $\mathcal{P}_1$, too. The proposed optimization method not only successfully avoids solving a nonlinear convex optimization problem with high dimensions and high orders, but also overwhelmingly decreases on the computational complexity and the computation time to acquire the optimal strategy. The following pseudo-code gives the accurate processes to solve the non-linear equation with a linear inequality constraint.

Algorithm 1 Bi-Inner: Bi-section algorithm for achieving the approximate quantity γ_i satisfied $\mu = \psi_i(\gamma_i)$ with μ known

```
Input: the value of lagrange multiplier $\mu, \delta_0 \leftarrow 1e-8$
$LB \leftarrow \gamma_i^l, UB \leftarrow \gamma_i^r$;
repeat
    $\bar{\gamma_i} \leftarrow \frac{UB+LB}{2}$;
    if $(\psi_i(\bar{\gamma_i}) - \mu > 0)$ then
        $UB \leftarrow \bar{\gamma_i}$;
    else
        $LB \leftarrow \bar{\gamma_i}$;
    end if
until $(|UB - LB| \le \delta_0)$;
$\gamma_i \leftarrow \bar{\gamma_i}$;
return $\gamma_i$;
```

Algorithm 1 calculates the approximate multiplier γ_i which satisfies (19) with given μ. Bi-JOTD computes the multiplier μ among the linear constraint. After Lagrangian multipliers γ_i^* and μ^* are achieved, we can use (15), (18), (21) obtain the final optimal resource allocation strategy $\boldsymbol{\lambda}^*$, $\boldsymbol{t}^{l*}$, and $\boldsymbol{t}^{u*}$.

Algorithm 2 Bi-JOTD: achieving the optimal lagrange multiplier μ^* satisfied $\boldsymbol{\mathcal{P}_3}$

```
Input: $N, \boldsymbol{\eta}, D, \boldsymbol{l}, B, T, P_t^c, P_d^c, P_F, d, N_0, \kappa_t, \kappa_d, \kappa_l,$
    $\delta_1 \leftarrow 1e-8$
$\boldsymbol{I} \leftarrow [I_1, ..., I_i, ..., I_N]$ {the whole input data of every task in the task graph}
$I_i = I_i + \sum_{j=1}^{N} I_j \eta_j D_{ij}$;
$LB_\mu \leftarrow \mu^l, UB_\mu \leftarrow \mu^r$;
repeat
    $\bar{\mu} \leftarrow \frac{UB_\mu + LB_\mu}{2}$;
    for each $i \in [1, N]$ do
        $\gamma_i \leftarrow \boldsymbol{Bi-Inner}(\bar{\mu})$;
    end for
    if $(\sum t_i^l(\gamma_i) - T > 0)$ then
        $LB_\mu \leftarrow \bar{\mu}$;
    else
        $UB_\mu \leftarrow \bar{\mu}$;
    end if
until $(|UB_\mu - LB_\mu| \le \delta_1)$;
$\mu^* \leftarrow \bar{\mu}$;
for each $i \in [1, N]$ do
    $\gamma_i^* \leftarrow \boldsymbol{Bi-Inner}(\bar{\mu})$;
    Calculate $t_i^{l*}$ using $\mu^*$, $\gamma_i^*$ and (21);
    Calculate $\lambda_i^*$ using $t_i^{l*}$, $\gamma_i^*$ and (18);
    Calculate $t_i^{u*}$ using $\lambda_i^*$, $\gamma_i^*$ and (15);
end for
return $\mu^*, \boldsymbol{t^{l*}}, \boldsymbol{t^{u*}}, \boldsymbol{\lambda^*}, \boldsymbol{\gamma^*}$;
```

Through analyzing the proposed algorithm Bi-JOTD, the complexity of Bi-JOTD is associated with the iteration number of inner and outer Bi-Section Search algorithms. We define $\mathcal{C}_{inner}$ and $\mathcal{C}_{outer}$ as the complexity of inner and outer algorithms, respectively. Therefore, the complexity of Bi-JOTD algorithm is $\mathcal{O}(N\mathcal{C}_{inner}\mathcal{C}_{outer})$. Compared with the original Problem $\mathcal{P}$, our proposed algorithm can greatly reduce the complexity.

4. Simulation Results

In this section, the simulation results of the proposed computation offloading scheme through joint optimization time allocation and dependency tasks is presented. The scheme is named as Bi-JOTD in the simulation results. Based on [16,23,24], the simulation parameters are set as follows. We assume that the channel reciprocity holds for the downlink and uplink, and thus $h_u^2 = h_d^2 = h^2$. The channel power gain is modeled as $h^2 = 10^{-3} d^{-\zeta} \Phi$ [24], where Φ stands for the short-term fading which obeys the Rayleigh fading. For distance d in meters with the same path-loss exponent ζ, a 30dB average signal power attenuation is assumed for all the channels at reference of 1m. Moreover, the input data size and the ratio of the computation results follow the uniform distribution with $I_i \in [10,100]KB$ and $\eta_i \in [0.01, 0.1]$, separately. The other detailed parameters are listed in **Table 1**.

Table 1. Simulation Parameters

Parameter	**Value**
Bandwidth B	5 $Mbps$
Deadline T	$1s$
Circuit powers P_t^c and P_d^c	$5.0*10^{-4}W$
Distance between the MD and the MEC server d	100m
Path-loss exponent ζ	1.7
White Gaussian channel noise N_0	$10^{-9}W$
MD's maximum CPU frequency f_{max}	$1.0GHz$
The linear growth coefficient κ_t	1.0
The effective capacitance coefficient κ_l	10^{-27}
The power consumption of receiving per bit κ_d	$2.86*10^{-7}\ W/bit$
The required number of CPU cycles per bit C	$800\ cycles/bit$

Meanwhile, in order to comparison, the following five baselines[5,16,17,23,24,25,26], are displayed.

1) $\lambda = 0.4$: The computation input data are divided into two parts for each task. The ratio $\lambda_i = 0.4$ of input data is accomplished in the MEC server, and the rest of input data is completed in MD.

2) Task: In the task-graph representing the mobile application, some tasks are offloaded to MEC server, and the rest of tasks execute on the mobile device.

3) $\lambda = 0.8$: The offloading ratio of computation data transmitted to MEC server is set as $\lambda_i = 0.8$.

4) Ave-JOTD: The offloading ratio λ_i is set as the average of suboptimal solution λ_i^*, i.e. $\lambda_i = \overline{\sum \lambda_i^*}$.

5) PSO-JOTD: The suboptimal task offloading scheme is achieved by using particle swarm optimizer (PSO).

4.1 The Validation of the Optimal Solution

In this subsection, we verify the correctness and superiority of the proposed algorithm. The energy consumption of the approximate solution is close to the cost of the exact solution in a tolerable accuracy range, which compares the simulation results of different block input data $I(KB)$ and the ratio η at the same deadline $T = 1.0s$ in **Fig. 3** and **Fig. 4**, respectively.

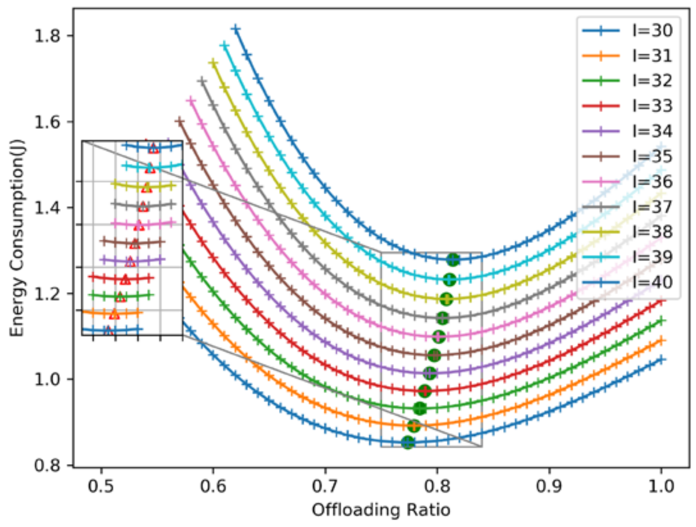

Fig. 3. The energy consumption with different I versus T

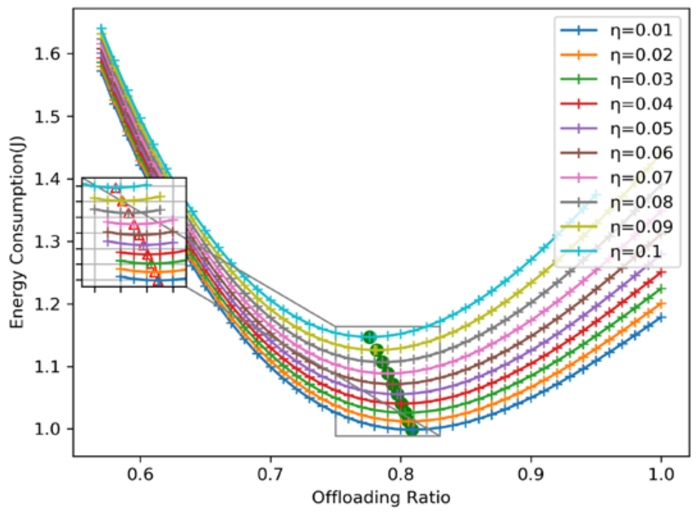

Fig. 4. The energy consumption with different η versus T

By comparing with the energy consumption between the global search with the default step size of 0.001 among the fixed range of offloading ratio λ and the proposed optimization algorithm, the energy consumption of the proposed approximate solution is minimum. The simulation results of **Fig. 3** and **Fig. 4** provide the evidence that problem $\mathcal{P}_2$ is a convex optimization problem. Meanwhile, the process of solving $\mathcal{P}_2$ is correct.

4.2 The Simulation Results for Task Graph

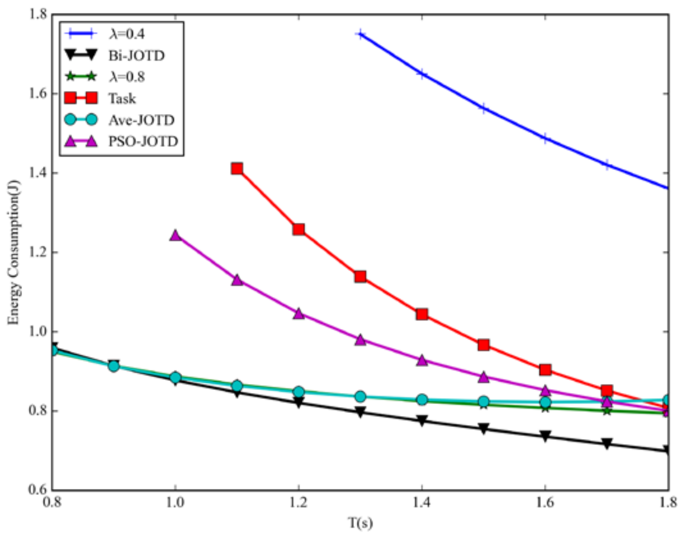

Fig. 5. The energy consumption with fixed I and random η versus T

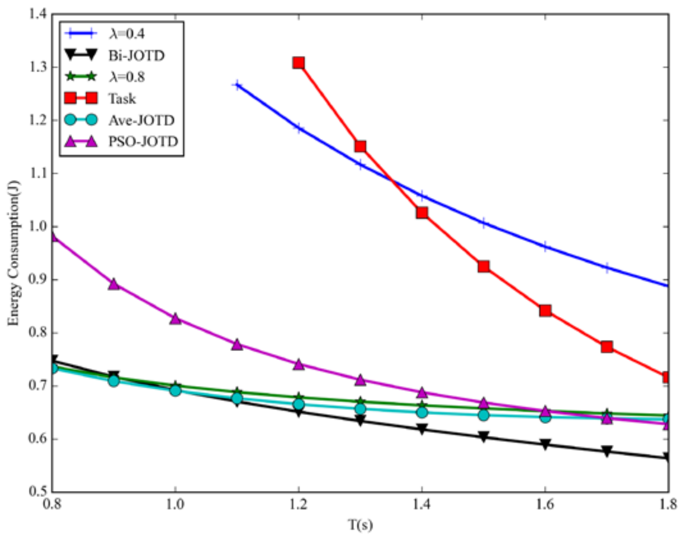

Fig. 6. The energy consumption with fixed η and random I versus T

This subsection evaluates the energy consumption of the proposed Bi-JOTD algorithm and other five baselines, which spends on computing and communicating versus the deadline T .

Fig. 5 makes the comparison simulation of energy consumption with a vector (I_i, η_i), where I_i is fixed and η_i is random, and **Fig. 6** gives a simulation with random I_i and fixed η_i. The proposed Bi-JOTD algorithm is obviously superior to the other baselines. Specifically, the energy consumption of the proposed algorithm is always less than other baselines, which further displays the advantages of offloading the computation tasks to the MEC server. The minimum energy cost has decreased trend with the increase of T. Therefore, Bi-JOTD algorithm has some advantages in solving multi-task application. Then, making a comparison test between Bi-JOTD and Ave-JOTD, we can see that the energy cost of Bi-JOTD is less than Ave-JOTD with the increase of T. It can be concluded that task's result feedback has a great influence on the ratio of offloading input data. Hence, we should comprehensively take task's feedback into account in the task graph to solve the objective of the formulated problem.

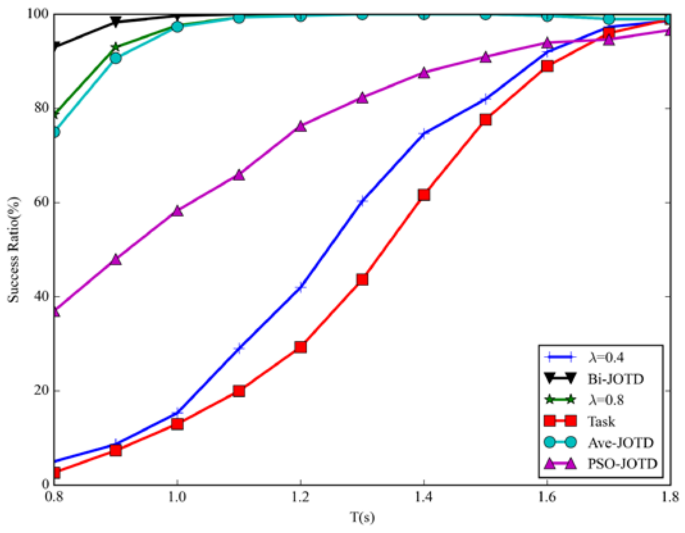

Fig. 7. The succeeding assignment ratios with random η and I versus T

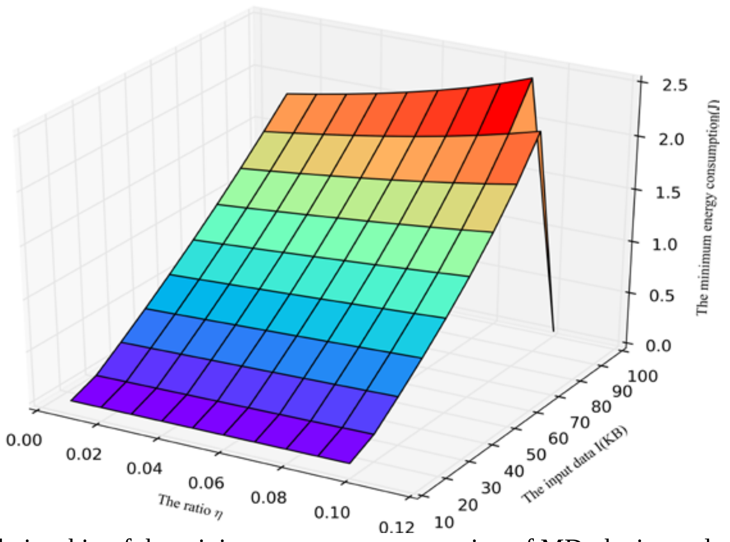

Fig. 8. shows the relationship of the minimum energy consumption of MD, the input data I and the ratio of the computation results η

The figure simulated in **Fig. 7** is based on 300 times simulated testing, in which I_i, η_i are randomly selected on the basis of the above assumptions in every implementation, modeling the actual computation offloading scenario. We simulate the resource allocation, and the statistics of success ratios are brought in **Fig. 7**. The success ratios of all the schemes increase with T, as we excepted. Besides, the success ratio of Bi-JOTD grows faster than others. Bi-JOTD can achieve a high success ratio even with low latency T. Meanwhile, as time goes by, the success ratio of Bi-JOTD reaches 100% and remains stable. Compared with other baselines, our proposed algorithm has more advantages in a high success ratio. **Fig. 8** shows the relationship of the minimum energy consumption of MD, the input data and the ratio of computation results. The energy consumption is achieved by using our proposed algorithm, and $T = 1.5s$. According to **Fig. 8**, while setting the value of η, the energy consumption increases significantly as the input data increases. When setting the value of the input data I, the energy consumption also shows an increasing trend with the increase of η, but the growth is not very obvious. In particular, the application cannot be accomplished while $I = 100KB$ and $\eta = 0.1$. Integrated with the previous simulation results and the optimization processes, the proposed algorithm not only has a great advantage in energy consumption, but also has a good performance on the executability of the computation offloading strategy. Ultimately, combining all figures and the equivalence of problem $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, we can get the conclusion that our proposed algorithm not only can obtain higher efficiency, less energy consumption and better performance, but also can be calculated more easily than others.

5. Conclusion

This work studies a computation offloading scheme for an application including multiple computation components with dependencies under a single-user MEC system. MEC server assists MD in completing its computation-intensive latency-critical application. The application is modeled as a DAG task-graph. By jointly optimizing the offloading ratio, CPU frequency, transmission power and transmission time, the objective that minimizes the energy consumption of MD is achieved. The problem is formulated as a optimization problem. The nonlinear equation with a linear inequality is obtained by using the Lagrange Multiplier method and convex optimization techniques. A double Bi-Section Search algorithm is proposed to solve the transformed nonlinear equation. Simulation results reveal that the proposed strategy greatly reduces the energy consumption while compared with the used baselines. The proposed computation offloading scheme not only reduces the difficulty of problem solving, but also prolongs the MD's standy time and achieves better performance. Meanwhile, we will extend our study of using partial computation offloading strategy to deal with the application including dependent tasks to the multiple mobile devices multiple MEC servers scenario.

References

[1] Z. Q. Jaber and M. I. Younis, "Design and implementation of real time face recognition system (rtfrs)," *International Journal of Computer Applications*, vol. 94, no. 12, pp. 15-22, 2014. Article (CrossRef Link)

[2] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[3] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?," *Computer*, vol. 43, pp. 51–56, Apr. 2010. Article (CrossRef Link)

[4] N. Vallina-Rodriguez and J. Crowcroft, "Energy management techniques in modern mobile handsets," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 179–198, First 2013. Article (CrossRef Link)

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017. Article (CrossRef Link)

[6] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal et al., "Mobile-edge computing introductory technical white paper," *ETSI, Sophia Antipolis, France, and MEC, London, U.K., Tech. Rep.*, pp. 1089–7801, 2014.

[7] Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, and E. Benkhelifa, "The future of mobile cloud computing: integrating cloudlets and mobile edge computing," in *Proc. of 23rd Int. Conf. Telecommun. (ICT). IEEE*, pp. 1–5, 2016. Article (CrossRef Link)

[8] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. of 1st Edition MCC Workshop Mobile Cloud Comput. ACM*, pp. 13–16, 2012. Article (CrossRef Link)

[9] G. I. Klas, "Fog computing and mobile edge cloud gain momentum open fog consortium, etsi mec and cloudlets," *Google Scholar*, 2015.

[10] A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Communication Letter*, vol. 6, no. 3, pp. 398–401, 2017. Article (CrossRef Link)

[11] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transaction on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015. Article (CrossRef Link)

[12] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transaction on Parallel Distribution System*, vol. 26, no. 4, pp. 974–983, 2014. Article (CrossRef Link)

[13] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Adaptive computation scaling and task offloading in mobile edge computing," in *Proc. of WCNC. IEEE*, pp. 1–6, 2017. Article (CrossRef Link)

[14] F. Wang, "Computation rate maximization for wireless powered mobile edge computing," in *Proc of 23rd Asia-Pacific Conf. Commun. (APCC). IEEE*, pp. 1–6, 2017. Article (CrossRef Link)

[15] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Transaction Wireless Communication*, vol. 17, no. 6, pp. 4177–4190, 2018. Article (CrossRef Link)

[16] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communication*, vol. 34, no. 12, pp. 3590–3605, 2016. Article (CrossRef Link)

[17] X. Hu, K.-K. Wong, and K. Yang, "Wireless powered cooperation-assisted mobile edge computing," *IEEE Transaction on Wireless Communication*, vol. 17, no. 4, pp. 2375–2388, 2018. Article (CrossRef Link)

[18] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Transaction on Wireless Communication*, vol. 17, no. 3, pp. 1784–1797, 2017. Article (CrossRef Link)

[19] Y. Liu, S. Wang, and F. Yang, "Poster Abstract: A multi-user computation offloading algorithm based on game theory in mobile cloud computing," in *Proc. of IEEE/ACM Symp. Edge Comput. (SEC). IEEE*, pp. 93–94, 2016. Article (CrossRef Link)

[20] W. Zhang, Y. Wen, and D. O. Wu, "Collaborative task execution in mobile cloud computing under a stochastic wireless channel," *IEEE Transaction on Wireless Communication*, vol. 14, no. 1, pp. 81–93, 2014. Article (CrossRef Link)

[21] S. E. Mahmoodi, K. Subbalakshmi, and V. Sagar, "Cloud offloading for multi-radio enabled mobile devices," in *Proc. of IEEE Int. Conf. Commun. (ICC). IEEE*, pp. 5473–5478, 2015. Article (CrossRef Link)

[22] Y.-H. Kao, B. Krishnamachari, M.-R. Ra, and F. Bai, "Hermes: Latency optimal task assignment for resource-constrained mobile computing," *IEEE Transaction on Mobile Computing*, vol. 16, no. 11, pp. 3056–3069, 2017. Article (CrossRef Link)

[23] S. Khalili and O. Simeone, "Inter-layer per-mobile optimization of cloud mobile computing: a message-passing approach," *Transaction on Emerging Telecommunications Technology*, vol. 27, no. 6, pp. 814–827, 2016. Article (CrossRef Link)

[24] S. E. Mahmoodi, R. Uma, and K. Subbalakshmi, "Optimal joint scheduling and cloud offloading for mobile applications," *IEEE Transaction on Cloud Computing*, 2016. Article (CrossRef Link)

[25] P. Di Lorenzo, S. Barbarossa, and S. Sardellitti, "Joint optimization of radio resources and code partitioning in mobile edge computing," *arXiv preprint arXiv:1307.3835*, 2013.

[26] S. Cao, X. Tao, Y. Hou, and Q. Cui, "An energy-optimal offloading algorithm of mobile computing based on HetNets," in *Proc. of 2015 International Conference on Connected Vehicles and Expo (ICCVE), Shenzhen, China*, pp. 254–258, 2015. Article (CrossRef Link)

[27] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proc. of IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation, Orlando, FL, USA*, pp. 4104–4108, 1997.

[28] A. Bhattcharya and P. De, "Computation offloading from mobile devices: Can edge devices perform better than the cloud?," in *Proc. of ARMS-CC. ACM*, pp. 1–6, 2016. Article (CrossRef Link)

[29] M. Safar, I. Ahmad, and A. Al-Yatama, "Energy-aware computation offloading in wearable computing," in *Proc. of Int. Conf. Comput. Appl. IEEE*, pp. 266–278, 2017. Article (CrossRef Link)

[30] A. R. Jensen, M. Lauridsen, P. Mogensen, T. B. Sørensen, and P. Jensen, "Lte ue power consumption model: For system level energy and performance optimization," in *Proc. of IEEE VTC Fall. IEEE*, pp.1–5, 2012. Article (CrossRef Link)

[31] G. Auer, O. Blume, V. Giannini, I. Godor, M. Imran, Y. Jading, E. Katranaras, M. Olsson, D. Sabella, P. Skillermark et al., "D2. 3: Energyefficiency analysis of the reference systems, areas of improvements andtarget breakdown," *Earth*, vol. 20, no. 10, 2010.

[32] S. Cui, A. J. Goldsmith, and A. Bahai, "Power estimation for Viterbi decoders," *Wireless Systems Lab, Stanford Univ., Stanford, CA, USA, Tech. Rep.*, 2003.

[33] O. Munoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transaction on Vehicular Technology*, vol. 64, no. 10, pp.4738–4755, 2014. Article (CrossRef Link)

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

[35] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for mobile edge computing," in *Proc. of IEEE WiOpt. IEEE*, pp. 1–6, 2018. Article (CrossRef Link)

Yang Li received the MS degrees from the College of Computer Science and Technology, Jilin University in 2014. Currently, he is PhD candidate at the College of Computer Science and Technology, Jilin University, China. His main research interests include mobile cloud computing, mobile edge computing, distributed computing.

Gaochao Xu received the BS, MS, and PhD degrees from the College of Computer Science and Technology, Jilin University in 1988, 1991, and 1995, respectively. Currently, he is the professor and PhD supervisor at the College of Computer Science and Technology, Jilin University, China. His main research interests include distributed system, grid computing, cloud computing, Internet of things, information security, software testing, and software reliability assessment, etc. As a person in charge or a principal participant, he has finished more than 10 national, provincial, and ministerial level research projects of China.

Jiaqi Ge received the B.S. and M.S. degree from the College of Computer Science and Technology, Jilin University, China, in 2018, where she is currently pursuing the Ph.D. degree. Her main research interests include mobile cloud computing, mobile edge computing, and the collaborative cloud and edge computing in the Internet of Things.

Peng Liu received the M.S. degree from the College of Computer Science and Technology, Jilin University, China, in 2016, and received Ph.D. degree in 2019. Now he is a lecturer in College of Information and Computer Engineering, Northeast Forestry University, Harbin, China. His main research interests include mobile cloud computing, mobile edge computing, edge caching, and the Internet of Things.

Xiaodong Fu received the M.Eng. degree from Jilin University in 2005. She is currently a Senior Engineer at the College of Computer Science and Technology, Jilin University, China. Her main research interests include distributed computation and software reliability analysis.