TOFFEE: Task Offloading and Frequency Scaling for Energy Efficiency of Mobile Devices in Mobile Edge Computing

Ying Chen, Ning Zhang, *Senior Member, IEEE*, Yongchao Zhang, Xin Chen, Wen Wu, *Student Member, IEEE*, and Xuemin (Sherman) Shen, *Fellow, IEEE*

Abstract—As an emerging computing paradigm, mobile edge computing (MEC) can improve users' service experience by provisioning the cloud resources close to the mobile devices. With MEC, computation-intensive tasks can be processed on the MEC servers, which can greatly decrease the mobile devices' energy consumption and prolong their battery lifetime. However, the highly dynamic task arrival and wireless channel states pose great challenges on the computation task allocation in MEC. This paper jointly investigates the task allocation and CPU-cycle frequency, to achieve the minimum energy consumption while guaranteeing that the queue length is upper bounded. We formulate it as a stochastic optimization problem, and with the aid of stochastic optimization methods, we decouple the original problem into two deterministic optimization subproblems. An online Task Offloading and Frequency Scaling for Energy Efficiency (TOFFEE) algorithm is proposed to obtain the optimal solutions of these subproblems concurrently. TOFFEE can obtain the close-to-optimal energy consumption while bounding the applications' queue length. Performance evaluation is conducted which verifies TOFFEE's effectiveness. Experiment results indicate that TOFFEE can decrease the energy consumption by about 15 percent compared with the RLE algorithm, and by about 38 percent compared with the RME algorithm.

Index Terms—Mobile edge computing, energy efficiency, task allocation, dynamic frequency scaling

1 Introduction

With the increasing proliferation of mobile services, the applications running on mobile devices are becoming more computation-intensive and energy-hungry [1], [2]. For the mobile device, the constrained computing and battery capacities have become the bottlenecks for processing these application tasks locally. Mobile cloud computing (MCC) is put forward as a potential solution to address this problem [3]. Mobile devices can transmit the computation tasks to MCC with the abundant cloud resources for processing. Offloading to the MCC can greatly augment the device's computing capacity and reduce their workload. However, the cloud servers usually locate far from the mobile devices. Data transmission from the devices to cloud servers would incur a large amount of energy consumption and transmission delay [4]. To mitigate these drawbacks, mobile edge computing (MEC) emerges as a promising paradigm providing the cloud resources at the radio access network (i.e., base station) [5], [6]. Unlike MCC, MEC deploys the computing resources in proximity to the mobile devices. Therefore, MEC can significantly reduce the network's energy consumption and the traffic of core network. In addition, with the help of computation offloading, the users' quality of experience (QoE), including battery consumption and response delay, would be greatly improved [7].

In the MEC system, one of the resource-limited device's major concerns is the battery lifetime [8]. To reduce the device's energy consumption, computation offloading and CPU-cycle frequency scaling have attracted increasing interests in academia and industry [7], [9], [10]. The computation offloading decisions are greatly affected by the wireless channel condition and tail energy. Transmitting tasks on the good channel condition or in batch can reduce the transmission energy consumption [8], [11]. Another solution for reducing the device's energy consumption is dynamic voltage frequency scaling (DVFS). When the application tasks are served on the device, the execution energy consumption mainly relies on the local CPU-cycle frequency. As CPU's power rises exponentially with the CPU-cycle frequency, the local execution energy can be significantly conserved by reducing the CPU-cycle frequency [12]. However, these energy conservation approaches would incur extra queueing delay for the application tasks, even making the mobile device unstable [3]. Therefore, in order to achieve the tradeoff between the mobile device's energy consumption and applications' performance (i.e., queuing delay), it is critical to determine the optimal task allocation and CPU frequency decisions effectively.

- *Y. Chen, Y. Zhang, and X. Chen are with the Computer School, Beijing Information Science and Technology University (BISTU), Beijing 100101, China. E-mail: {chenying, chenxin}@bistu.edu.cn, zhangyongchao@mail.bistu.edu.cn.*
- *N. Zhang is with Texas A&M University-Corpus Christi, Corpus Christi, TX 78412 USA. E-mail: ning.zhang@tamucc.edu.*
- *W. Wu and X. (Sherman) Shen are with University of Waterloo, Waterloo, ON N2L 3G1, Canada. E-mail: {w77wu, sshen}@uwaterloo.ca.*

Manuscript received 31 Oct. 2018; revised 1 Apr. 2019; accepted 1 June 2019. Date of publication 20 June 2019; date of current version 6 Dec. 2021. (Corresponding author: Ning Zhang.) Recommended for acceptance by A. Zomaya. Digital Object Identifier no. 10.1109/TCC.2019.2923692

TABLE 1
Notations and Definitions

Notion	Definition
$\mathbf{N}$	Applications set
τ	Slot length
P	Mobile device's transmit power
$h(t)$	Wireless channel's power gain
B	Bandwidth of the wireless channel
σ^2	Noise power at the receiver
$R_i(t)$	Task offloading rate
$A_i(t)$	Amount of ith application's input tasks
$D_i^l(t)$	Amount of tasks processed locally from the ith application
$D_i^r(t)$	Amount of ith application's offloaded tasks
$D_i(t)$	Amount of processed tasks from the ith application
φ_i	The required CPU cycles to compute 1 bit ith application task
$f(t)$	CPU-cycle frequency
$E^l(t)$	Energy consumption for local execution
$E^r(t)$	Energy consumption for transmission
$E^a(t)$	Tail energy consumption
$E_{total}(t)$	Total energy consumption
$Q_i(t)$	Queue backlog of the ith application

It is a challenging work to devise an algorithm that couples the task allocation and frequency scaling for the MEC system. First, the task arrival of each application is dynamic and stochastic over time. Allocating more or less computation on the local CPU may both lead to the large queueing delay [13]. Second, the task allocation should take into consideration of the current queue state as well as the wireless channel condition. Transmitting computation tasks intermittently or on the bad channel condition would incur extra energy consumption. However, the task arrival and wireless channel are not only affected by the characteristics of the MEC system, but also influenced by the external environment [14]. As a result, the statistical information of these stochastic processes is hardly obtained precisely in advance. It would face considerable challenges to design an algorithm that could adapt to the highly dynamics of task arrival and wireless channel.

To tackle the above challenges, we investigate the stochastic joint task allocation and CPU-cycle frequency scaling for the MEC system. We seek to achieve the minimum average energy consumption when the queueing delay is upper bounded. A stochastic optimization problem is formulated to achieve the energy efficiency for mobile device. By employing the stochastic optimization techniques, the original problem is decoupled into two deterministic subproblems. An effective algorithm, TOFFEE, is designed to solve these subproblems. TOFFEE requires no prior statistical information about the stochastic processes (task arrival and wireless network). Mathematic analysis shows the asymptotic optimality of TOFFEE. The performance of TOFFEE is also evaluated via the experiments.

For the rest of the paper, the system model and stochastic optimization problem are provided in Section 2. We propose TOFFEE to solve this optimization problem in Section 3. In Section 4, we analyze TOFFEE's performance. In Section 5, the experiment results are presented. Related works are presented in Section 6, and Section 7 is the summary of this paper.

2 System Model

Consider that a mobile device runs n different applications $\mathbf{N} = \{1, 2, \ldots, n\}$, and a MEC server is installed at the base station (BS) for providing computing services. The mobile device can send application tasks to the MEC server through wireless channel to get powerful computing ability and extend battery life. A discrete time-slotted system denoted by $t = \{0, 1, \ldots, T-1\}$ is considered. Each time slot's length is τ. In slot t, according to the wireless channel condition and queue backlog of the unprocessed data, the mobile device jointly determines the task allocation decision (i.e., the local execution and offloaded computation) and local CPU-cycle frequency. In Table 1, the main notations in the system model are provided.

2.1 Task and Queueing Models

In this paper, the tasks generated by the $|\mathbf{N}|$ applications are computation-intensive. It is considered that the tasks can be divided into several parts for local execution or MEC execution, and the processing complexity of these tasks is proportional to the input tasks size (in bits) [5], [7], [10]. In slot t, let $A_i(t)$ (in bits) be the amount of input tasks from the ith application. For generality, the task arrival $A_i(t)$ is different among these applications. For the ith application, $D_i^l(t)$ stands for the amount of task processed locally, and $\varphi_i > 0$ denotes the required CPU cycles to compute 1 bit data, which can be obtained by using the call graph analysis method [6], [15]. φ_i depends on the nature of application, and is different among the n applications [16]. Thus, the sum of the n applications' required CPU cycles is $\sum_{i=1}^{n} \varphi_i D_i^l(t)$. Besides, $f(t)$ denotes the CPU-cycle frequency, and is upper bounded by the maximum value f^{max}, which is expressed by,

$$f(t) \leq f^{max}. \tag{1}$$

Given the CPU-cycle frequency $f(t)$, the local computing capability of the device is limited. Thus, the local computation should satisfy (2).

$$\sum_{i=1}^{n} \varphi_i D_i^l(t) \leq f(t)\tau. \tag{2}$$

Let $D_i^r(t)$ denote the amount of offloaded tasks from the ith application. Then, the total offloaded computation is $\sum_{i=1}^{n} D_i^r(t)$. Define $h(t)$ as the wireless channel's power gain, P as the device's transmit power. Thus, the achievable data transmission rate is

$$R(t) = B \log_2\left(1 + \frac{Ph(t)}{\sigma^2}\right), \tag{3}$$

In (3), σ^2 represents the power of channel noise, B denotes the channel bandwidth.

In slot t, the device's data transmission capability is also limited. Thus, the offloaded computation should satisfy (4).

$$\sum_{i=1}^{n} D_i^r(t) \leq R(t)\tau. \tag{4}$$

Recall that $D_i^r(t)$ represents the amount of task processed locally, $D_i^l(t)$ represents the amount of offloaded tasks. Thus, the ith application's total computation tasks $D_i(t)$ that leaves the queue is

$$D_i(t) = D_i^l(t) + D_i^r(t). \tag{5}$$

For each application, let $Q_i(t)$ denote the queue backlog of the unaccomplished tasks. Then, the ith application's queue length evolves as the following equation,

$$Q_i(t+1) = [Q_i(t) - D_i(t), 0]^+ + A_i(t). \tag{6}$$

Notice that the task queueing delay of each application is proportional to its queue backlog [17]. Therefore, in order to guarantee the performance of each application, all the queue backlogs need to be stable, which is

$$q_i = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\{Q_i(t)\} < \varepsilon, \exists\ \varepsilon \in \mathbb{R}^+, \tag{7}$$

where q_i is the average queue backlog over the slots. In this paper, the overhead of partitioning, migrating, and bringing the results back is ignored, as done by most literatures on computation offloading, such as [18], [19], [20].

2.2 Energy Consumption Model

The device's energy consumption includes the local execution energy, transmission energy and tail energy [15]. Specifically, tail energy is generated after the data transmission because the device would maintain at the high power state during a certain period (tail time). In the 3G/4G/LTE networks, tail time is introduced to reduce the high signaling overhead [11], [21].

The local execution energy is mainly consumed by the mobile device's CPU operation, and depends on the computation load and CPU-cycle frequency [22]. Then, define $E^l(t)$ as the energy consumed by local execution, which is

$$E^l(t) = \xi f^2(t) \sum_{i=1}^{n} \varphi_i D_i^l(t), \tag{8}$$

where ξ represents the effective switched capacitance, and is determined by the chip architecture [23].

The energy consumed by data transmission is equal to the product of transmission power and duration. Therefore, let $E^r(t)$ denote the energy consumed by data transmission, which can be derived by (9).

$$E^r(t) = P \sum_{i=1}^{n} D_i^r(t)/R(t). \tag{9}$$

In the current cellular network, the channel allocation is determined by the Radio Resource Control (RRC) protocol. The RRC protocol have three different states, including: data transmission (DT), tail (TA) and idle (ID) [21], [24]. The three states' radio powers are P, P_T and P_I, respectively. When the radio is at TA or ID, the transmission data's arrival would trigger it to DT with the higher power. After the data transmission, it would switch to TA. If no data needs to be transmitted, the radio would stay at the TA for a period of δ_T, and switch to ID.

Define Δt as the time duration after the last transmission. Then, if $\sum_{i=1}^{n} D_i^r(t) > 0$, there exists computation tasks which need to be transmitted during the slot, so the tail time in slot t is 0. Otherwise, no computation tasks need to be transmitted, and the tail time would be τ. Therefore, the tail time $\Delta t'$ in slot t is

$$\Delta t' = \begin{cases} 0, & \sum_{i=1}^{n} D_i^r(t) > 0 \\ \tau, & otherwise. \end{cases} \tag{10}$$

Then, let $E^a(t)$ represent the tail energy consumption in slot t, which can be obtained by (11).

$$E^a(t) = \begin{cases} 0, & \Delta t \geq \delta_T \ or\ \Delta t' = 0 \\ P_T * \tau, & \Delta t < \delta_T, \Delta t + \Delta t' \leq \delta_T \\ P_T * (\delta_T - \Delta t), & otherwise. \end{cases} \tag{11}$$

Thus, the device's energy consumption in slot t can be given by (12).

$$E_{total}(t) = \xi f^2(t) \sum_{i=1}^{n} \varphi_i D_i^l(t) + P \sum_{i=1}^{n} D_i^r(t)/R(t) + E^a(t). \tag{12}$$

In different time, due to the dynamics in the quality of wireless channel, the energy consumed to transmit the same amount of data may vary from each other. And the device's tail energy consumption is also relative with the last computation offloading decision. Therefore, this paper focuses on the device's long-term ($T \to \infty$) average energy consumption,

$$\bar{E} = \lim_{T\to\infty} \frac{\sum_{t=0}^{T-1} \mathbf{E}\{E_{total}(t)\}}{T}. \tag{13}$$

2.3 Optimization Problem

In this paper, we optimize the device's average energy consumption in (13). The energy consumed by local computing can be conserved by reducing the local computation and CPU-cycle frequency. The energy consumed to transmit data can be decreased by transmitting data on good channel condition or reducing the offloaded computation [8]. Moreover, transmitting data in batch would decrease the tail energy consumption [11]. However, these approaches would cause that the queue backlog of each application becomes large and the mobile device tends to be unstable. Thus, in the paper, we aim at achieving the minimum average energy consumption and bounding the queueing delay of each application. The optimization problem is,

$$\min_{\mathbf{D}^l(t), \mathbf{D}^r(t), f(t)} \quad \bar{E} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\{E_{total}(t)\}, \tag{14}$$

subject to constraints (1), (2), (4) and (7).

Because the task arrival and wireless channel condition are both dynamic and stochastic, Problem (14) is a stochastic optimization problem.

3 Algorithm Design

A task offloading and frequency scaling for energy efficiency (TOFFEE) algorithm is designed in this section. By employing the Lyapunov optimization techniques [25], [26], Problem (14) is decomposed into two deterministic optimization sub-problems. TOFFEE can obtain the optimal solutions of theses sub-problems in low time complexity. In addition, TOFFEE makes the task allocation and CPU frequency decisions without any prior statistic information of the task arrival and wireless network, which are also hard to predict or acquire precisely in practice.

3.1 Problem Transformation

According to Lyapunov optimization theory, we transform Problem (14) into the deterministic optimization problem in each slot. Let $\mathbf{\Theta}(t)$ denote the backlog matrix of the n applications' queues. And the Lyapunov function is,

$$L(\mathbf{\Theta}(t)) = \frac{1}{2}\sum_{i\in\mathbf{N}} Q_i^2(t). \tag{15}$$

In (15), $L(\mathbf{\Theta}(t)) \geq 0$ represents the device's queue backlog. When $L(\mathbf{\Theta}(t))$ is large, it means that at least one application's queue backlog is large. Only when all the applications' queue backlogs are small, the value of $L(\mathbf{\Theta}(t))$ is small. Therefore, in order to decrease the applications' queue backlogs and keep them at a low level, we aim at reducing the value of $L(\mathbf{\Theta}(t))$. Then, $\Delta(\mathbf{\Theta}(t))$ is defined as the *conditional Lyapunov drift*,

$$\Delta(\mathbf{\Theta}(t)) = \mathbf{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))|\mathbf{\Theta}(t)\}. \tag{16}$$

Following the Lyapunov optimization framework and combining the energy consumption with the applications performance (queueing delay), we define *drift plus energy*, $\Delta_V(\mathbf{\Theta}(t))$, which is given as follows,

$$\Delta_V(\mathbf{\Theta}(t)) = \Delta(\mathbf{\Theta}(t)) + V\mathbf{E}\{E_{total}(t)|\mathbf{\Theta}(t)\}, \tag{17}$$

where V is non-negative and indicates the weight put on the energy consumption. A larger V puts more weight on the energy consumption. In the realistic scenario, V is determined according to the preference on the queue backlog and energy consumption .

Next, Theorem 1 gives $\Delta_V(\mathbf{\Theta}(t))$'s upper bound.

Theorem 1. *In each slot t, for any V and $\mathbf{\Theta}(t)$, if there exist the upper bounds of $A_i(t)$ and $R(t)$, which are A_i^{max} and R^{max}, the $\Delta_V(\mathbf{\Theta}(t))$ of any possible algorithms would satisfy,*

$$\begin{aligned}
&\Delta(\mathbf{\Theta}(t)) + V\mathbf{E}\{E_{total}(t)|\mathbf{\Theta}(t)\} \\
&\quad\leq C + \sum_{i=1}^{n} Q_i(t)\mathbf{E}\{A_i(t) - D_i(t)|\mathbf{\Theta}(t)\} \\
&\quad + V\xi\mathbf{E}\{f^2(t)\sum_{i=1}^{n}\varphi_i D_i^l(t)|\mathbf{\Theta}(t)\} \\
&\quad + VP\sum_{i=1}^{n}\mathbf{E}\{D_i^r(t)/R(t)|\mathbf{\Theta}(t)\} + V\mathbf{E}\{E^a(t)|\mathbf{\Theta}(t)\},
\end{aligned} \tag{18}$$

where $C = \frac{1}{2}\sum_{i=1}^{n}[(A_i^{max})^2 + (\frac{f^{max}\tau}{\varphi_i} + R^{max}\tau)^2]$ is a constant.

Theorem 1's proof is in Appendix A.

3.2 Online Algorithm

We design TOFFEE to minimize $\Delta_V(\mathbf{\Theta}(t))$'s upper bound. We also prove that TOFFEE can achieve a close-to-optimal energy consumption in Section 4.

In slot t, since C and $A_i(t)$ can be regarded as the constant, we can rewrite the minimization problem as,

$$\begin{aligned}
\min_{\mathbf{D}^l(t),\mathbf{D}^r(t),f(t)} \quad &\Big\{\sum_{i=1}^{n}[V\xi f^2(t)\varphi_i D_i^l(t) - Q_i(t)D_i^l(t)] \\
&+ \sum_{i=1}^{n}[VP/R(t) - Q_i(t)]D_i^r(t) + VE^a(t)\Big\}. \\
s.t.\ &(1),(2),(4).
\end{aligned} \tag{19}$$

We first simplify the above minimization problem by determining the optimal CPU-cycle frequency, and propose Lemma 1.

Lemma 1. *For the given local computation $\sum_{i=1}^{n}\varphi_i D_i^l(t)$, the optimal CPU-cycle frequency $f^*(t)$ for Problem (19) is*

$$f^*(t) = \sum_{i=1}^{n}\varphi_i D_i^l(t)/\tau. \tag{20}$$

Proof. With the given $\sum_{i=1}^{n}\varphi_i D_i^l(t)$, the objective of Problem (19) is a non-decreasing function with $f(t)$. The optimal value of $f(t)$ must be as small as possible. However, $f(t)$ should satisfy that $f(t) \geq \sum_{i=1}^{n}\varphi_i D_i^l(t)/\tau$ in (2). Thus, we can obtain the optimal $f^*(t)$ should be $\sum_{i=1}^{n}\varphi_i D_i^l(t)/\tau$. □

Based on Lemma 1, constraint (1) can be converted to the following inequality,

$$\sum_{i=1}^{n}\varphi_i D_i^l(t) \leq f^{max}\tau. \tag{21}$$

Then, Problem (19) can be equivalently reformulated as,

$$\begin{aligned}
\min_{\mathbf{D}^l(t),\mathbf{D}^r(t)} \quad &\Big\{\frac{V\xi}{\tau^2}\sum_{i=1}^{n}[\varphi_i D_i^l(t)]^3 - \sum_{i=1}^{n} Q_i(t)D_i^l(t) \\
&+ \sum_{i=1}^{n}[VP/R(t) - Q_i(t)]D_i^r(t) + VE^a(t)\Big\}. \\
s.t.\ &(4),(21).
\end{aligned} \tag{22}$$

Note that $\mathbf{D}^l(t)$ and $\mathbf{D}^r(t)$ are decoupled in the objective and constraints of Problem (22). Therefore, Problem (22) can be decomposed into two independent subproblems. Particularly, the two sub-problems are: local computation allocation (*LCA*) and offloaded computation allocation (*OCA*). In the following, the optimal solution of each sub-problem is obtained separately.

3.2.1 Local Computation Allocation (LCA)

Considering the first two terms of the objective in (22) and the relevant constraint (21), we can obtain the optimal local computation $\mathbf{D}^l(t)$ by solving (23).

$$\min_{\mathbf{D}^l(t)} \quad \frac{V\xi}{\tau^2}\sum_{i\in\mathbf{N}}[\varphi_i D_i^l(t)]^3 - \sum_{i\in\mathbf{N}} Q_i(t)D_i^l(t). \tag{23}$$

$$s.t.\ \sum_{i\in\mathbf{N}}\varphi_i D_i^l(t)\le f^{max}\tau.$$

To get the optimal solution of *LCA*, we first assume that $\sum_{i=1}^{n}\varphi_i D_i^l(t)$ has been determined. Then, the *LCA* problem is simplified as (24).

$$\min_{\mathbf{D}^l(t)}\ -\sum_{i=1}^{n}\frac{Q_i(t)}{\varphi_i}\varphi_i D_i^l(t). \tag{24}$$

Problem (24) can be regraded as a generalized min-weight problem. Specifically, the local computation $\varphi_i D_i^l(t)$ is weighted by $-\frac{Q_i(t)}{\varphi_i}$. It is evident that the optimal solution is

$$D_i^l(t)=\begin{cases}\frac{\sum_{i=1}^{n}\varphi_i D_i^l(t)}{\varphi_i}, & i=i^*\\ 0, & otherwise,\end{cases} \tag{25}$$

where $i^*\in \mathrm{argmax}_{i\in\{1,2,\ldots,n\}}\frac{Q_i(t)}{\varphi_i}$ represents the index of the application with the maximum value of $\frac{Q_i(t)}{\varphi_i}$. Therefore, for the given $\sum_{i=1}^{n}\varphi_i D_i^l(t)$, the optimal local computation can be obtained according to (25). However, the value of $\sum_{i=1}^{n}\varphi_i D_i^l(t)$ has not been determined yet. Then, let X equal to $\sum_{i=1}^{n}\varphi_i D_i^l(t)$, and the *LCA* problem is rewritten as follows,

$$\min_{X}\ \frac{V\xi}{\tau^2}X^3-Q_{i^*}(t)X. \tag{26}$$
$$s.t.\ 0\le X\le f^{max}\tau.$$

Problem (26) is a simple convex optimization problem, and the optimal solution X^* can be obtained by the derivation. After the value of $X=\sum_{i=1}^{n}\varphi_i D_i^l(t)$ is determined, we can obtain the *LCA*'s optimal solution according to (25).

3.2.2 Offloaded Computation Allocation (OCA)

Considering the last two terms of the objective in (22) and the relevant constraint (4), we can obtain the optimal offloaded computation $\mathbf{D}^r(t)$ by solving (27).

$$\min_{\mathbf{D}^r(t)}\ \sum_{i\in\mathbf{N}}[VP/R(t)-Q_i(t)]D_i^r(t)+VE^a(t). \tag{27}$$
$$s.t.\ \sum_{i\in\mathbf{N}}D_i^r(t)\le R(t)\tau.$$

According to (10) and (11), there exist two possible values for the tail energy $E^a(t)$, which is expressed by (28).

$$E^a(t)=\begin{cases}e1, & \sum_{i=1}^{n}D_i^r(t)>0\\ e2, & otherwise.\end{cases} \tag{28}$$

Therefore, if $\sum_{i=1}^{n}D_i^r(t)=0$, the minimum objective value of *OCA* problem equals to $V\cdot e2$, which is represented by $O1$. If $\sum_{i=1}^{n}D_i^r(t)>0$, the *OCA* problem can be transformed into (29).

$$\min_{\mathbf{D}^r(t)}\ \sum_{i=1}^{n}[VP/R(t)-Q_i(t)]D_i^r(t). \tag{29}$$
$$s.t.\ 0<\sum_{i=1}^{n}D_i^r(t)\le R(t)\tau.$$

Problem (29) is a generalized min-weight problem. Specifically, the offloaded computation $D_i^r(t)$ is weighted by $VP/R(t)-Q_i(t)$. The optimal solution is

$$D_i^r(t)=\begin{cases}R(t)\tau, & i=i^*\\ 0, & otherwise,\end{cases} \tag{30}$$

where $i^*\in \mathrm{argmin}_{i\in\mathbf{N}}\{VP/R(t)-Q_i(t)\}$. Then, when $\sum_{i=1}^{n}D_i^r(t)>0$, we can obtain its optimal objective value $O2$ as follows,

$$O2=[VP/R(t)-Q_{i^*}(t)]\cdot R(t)\tau+V\cdot e1. \tag{31}$$

Thus, according to the above discussion, the optimal solution to *DCO* problem is (32).

$$D_i^r(t)=\begin{cases}R(t)\tau, & i=i^*, O1>O2\\ 0, & i\ne i^*, O1>O2\\ 0, & O1\le O2.\end{cases} \tag{32}$$

After the optimal computation allocation $\mathbf{D}^l(t)$, $\mathbf{D}^r(t)$ and CPU-cycle frequency $f(t)$ are determined, the queue backlog $Q_i(t)$ of each application updates according to (6).

In Algorithm 1, the details of TOFFEE are presented.

Algorithm 1. Task Offloading and Frequency Scaling for Energy Efficiency (TOFFEE)

1: Observe the current queue backlog of each application $Q_i(t)$.
2: Obtain $\sum_{i=1}^{n}\varphi_i D_i^l(t)$ by solving the convex optimization problem (26).
3: **for all** $i\in\mathbf{N}$ **do**
4: Calculate $\frac{Q_i(t)}{\varphi_i}$ for the ith application.
5: **end for**
6: **for all** $i\in\mathbf{N}$ **do**
7: Search for index i^* with the maximum value of $\frac{Q_i(t)}{\varphi_i}$.
8: **end for**
9: Set $D_i^l(t)$ according to (25).
10: Set $f(t)$ according to (20).
11: Calculate the tail energy $e1$ and $e2$ according to (10) and (11).
12: **for all** $i\in\mathbf{N}$ **do**
13: Compute $VP/R(t)-Q_i(t)$ for the ith application.
14: **end for**
15: **for all** $i\in\mathbf{N}$ **do**
16: Search for index i^* with the minimum value of $VP/R(t)-Q_i(t)$.
17: **end for**
18: Calculate $O1$ and $O2$.
19: Set $D_i^r(t)$ according to (32).

4 Algorithm Analysis

To analyze the performance of TOFFEE, we conduct the theocratical analysis in this section. It can be proven that TOFFEE can arbitrarily approach the minimal average energy consumption when the queue backlog has bound.

Let $\bar{Q}$ denote the mobile device's average queue length, which is expressed as (33).

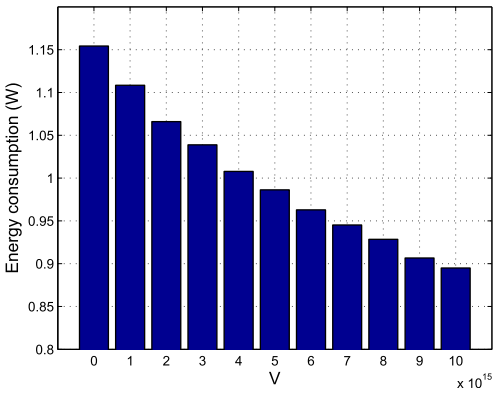


Fig. 1. Energy consumption with different V.

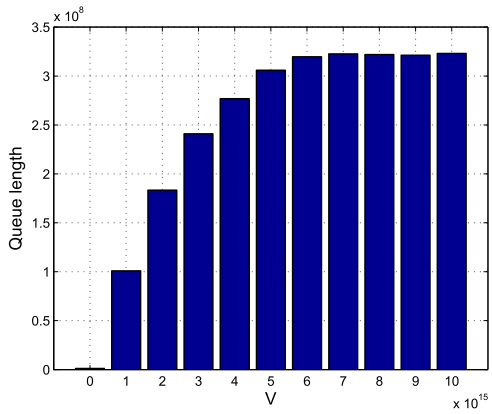


Fig. 2. Queue length with different V.

$$\bar{Q} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^{n} \mathbf{E}\{Q_i(t)\}. \tag{33}$$

Theorem 2. *For any value of V, given all the application's average arrival rates $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_n\}$, if there exists $\epsilon > 0$ which satisfies $\sum_{i=1}^{n}(\lambda_i + \epsilon) \in \Lambda$, the average energy consumption of TOFFEE would be upper bounded by,*

$$\bar{E}^{our} \leq e^* + \frac{C}{V}. \tag{34}$$

Furthermore, the time average queue backlog of TOFFEE would also be bounded by,

$$\bar{Q} \leq \frac{V(\hat{E} - \check{E}) + C}{\epsilon}, \tag{35}$$

where C is defined in (18), and e^* denotes the minimum average energy consumption.

Theorem 2's proof is in Appendix B.

Remark. Theorem 2 shows that the energy consumption $\bar{E}$ decreases when V becomes larger; meanwhile, the queue length $\bar{Q}$ increases. Thus, TOFFEE is able to reach the energy consumption and queue length tradeoff. Although $\bar{Q}$ increases with V, it can also be upper bounded. In addition, according to (34), TOFFEE can approach the optimal energy consumption with the sufficiently large value of V. In practice, V can be determined referring to characteristics of the mobile device and applications.

Then, TOFFEE's time complexity is given. For the two inner loops (line 6-8 and line 15-17) in Algorithm 1, TOFFEE traverses each application once. Therefore, each loop would stop in $O(n)$ operations, in which n represents the application types' number. Thus, TOFFEE's time complexity is $O(n)$. According to the above analysis result, our TOFFEE would be feasible for the large systems.

5 Evaluation

We conduct the parameter analysis and comparison experiments to evaluate TOFFEE's performance. In the experiments, there are three types of applications in the device, which are video transcoding, chess game and 6-queues puzzle [27]. The amount of each application's input task is set to follow certain fixed distributions. Specifically, the ith application's input tasks in each slot t follows a uniform distribution within $[0, A_i^{max}]$ bits. In fact, TOFFEE does not need any statistical information about the task arrival. For each application, the required CPU cycles to compute 1 bit task is uniformly distributed in $[1000, 2000]$ cycles/bit [18]. Similar to [28], the wireless channel is considered as a Ralyigh fading channel, where the channel gain $h(t)$ follows the exponential distribution and the mean is 1. In addition, the slot length $\tau = 1$ s, $P = 1.6$ W, $B = 1$ MHz, $\sigma^2 = 10^{-6}$ W, $f^{max} = 1$ GHz, $P_T = 1.1$ W, $\delta_T = 10$ s, $\xi = 10^{-27}$ [9], [24].

5.1 Parameter Analysis

5.1.1 Effect of Tradeoff Parameter V

Figs. 1 and 2 evaluate the effects of different V on the energy consumption and queue length, respectively. Fig. 1 shows that the energy consumption becomes smaller when V rises, which is in accordance with (34) in Theorem 2. It is because when increasing V, more weight would be put on the energy consumption. In this case, TOFFEE would adaptively tune the computation allocation decisions to decrease the energy consumption. Fig. 2 shows that the queue length becomes larger when V increases, confirming to (35) in Theorem 2. It also shows that although the queue length rises when V rises, it can still be upper bounded. We can also see that TOFFEE can always keep the device stable. Together with Figs. 1 and 2, by adjusting V, TOFFEE reaches the arbitrary tradeoff between the energy consumption and queue length. Additionally, when V is sufficiently large, the minimum energy consumption can be reached by TOFFEE and the queues are stabilized.

5.1.2 Effect of Arrival Rate

Figs. 3 and 4 show TOFFEE's energy consumption and queue length with different arrival rates. In the experiment, each application's arrival rate is $\alpha \cdot A_i(t)$. Three different arrival rates are considered, in which $\alpha = 0.5$, 1 and 1.5, respectively. As shown in Fig. 3, the energy consumption increases when arrival rate becomes larger. The reason is that as arrival rate increases, more computation tasks would be allocated to be computed locally and offloaded. As a result, the total energy consumption including local execution energy and transmission energy would also rise. As expected, we can observe in Fig. 4 that the queue length rises when arrival rate becomes

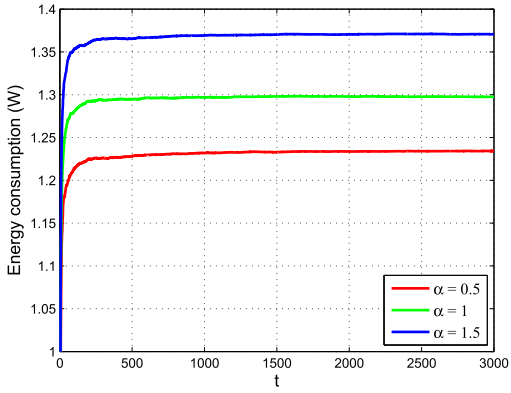


Fig. 3. Energy consumption with different arrival rate.

larger. Additionally, for the three different arrival rates, TOFFEE can all quickly stabilize the energy consumption and queue length. It indicates that TOFFEE can adaptively allocate the computation workload to keep the device stable in a short time.

5.1.3 Effect of Slot Length

In this experiment, we evaluate the queue length with different slot lengths, which is shown in Fig. 5. Three different slot lengths (i.e., time intervals) are considered, where $\tau = 0.5$s, 1s and 1.5s, respectively. It can be seen that for the three settings, all the queue lengths of TOFFEE would stabilize quickly. It can also be seen that as the slot length becomes larger, the queue length would increase. The reason is that when the slot length τ is set larger, it is harder for TOFFEE to adapt to system dynamics. However, when τ is set too small, it may incur high overhead to acquire the system parameters or variables.

5.2 Comparison Experiments

In the experiments, to verify TOFFEE's effectiveness, two baseline algorithms are introduced to compare with TOFFEE [28], [29], [30]:

- *Round-robin Local Execution (RLE)*: The computation tasks of different applications are processed in different slots in turn. And in each slot, all the computation tasks are computed locally.
- *Round-robin MEC Execution (RME)*: The computation tasks of different applications are processed in different slots in turn. And in each slot, all applications' tasks in the device are offloaded.

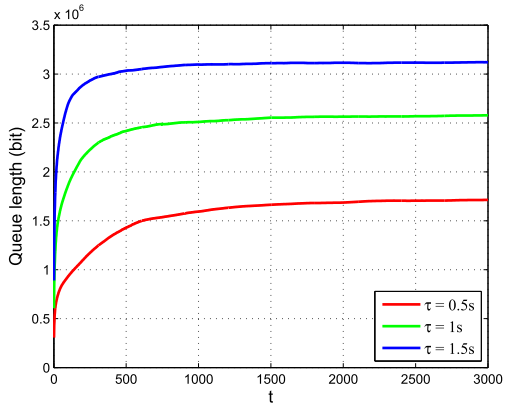


Fig. 5. Queue length with different slot lengths.

Fig. 6 presents the energy consumption of three different algorithms. The TOFFEE's energy consumption is the lowest among the three algorithms. We can see that TOFFEE can reduce the energy consumption by about 15 percent compared with the RLE algorithm, and by about 38 percent compared with the RME algorithm. It demonstrates that TOFFEE can decrease the device's energy consumption effectively. It is because that TOFFEE can dynamically allocate the computation workload and schedule the CPU-cycle frequency to adapt to the dynamics of the task arrival and channel condition. And Fig. 7 plots the three different algorithms' queue length. It shows that the RLE's queue length rises linearly, which causes the device instability. It is because that the mobile device's computing ability is limited, and the computation tasks arrived per slot exceed the device's processing capacity. As a result, more and more computation tasks would be backlogged in the mobile device, and queue length would also increase continuously. However, we can also see that the queue length of TOFFEE and RME are both small. It shows that TOFFEE can maintain the queue backlog at a low level. Combining Figs. 6 and 7, TOFFEE can decrease the energy consumption effectively while guaranteeing the performance of each application.

Table 2 shows the execution times (in millisecond) of the three different algorithms. Every experiment is run 500 times, and the average result is computed. It shows that RLE's execution time is the smallest among the three algorithms, and

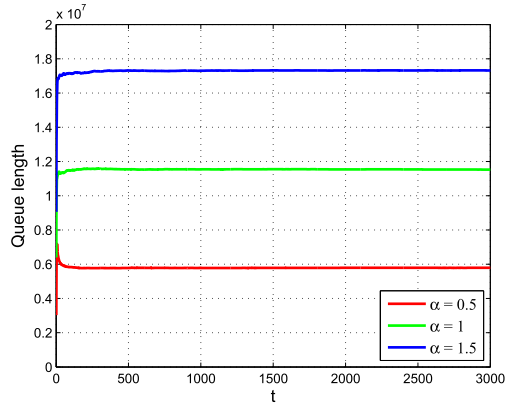


Fig. 4. Queue length with different arrival rate.

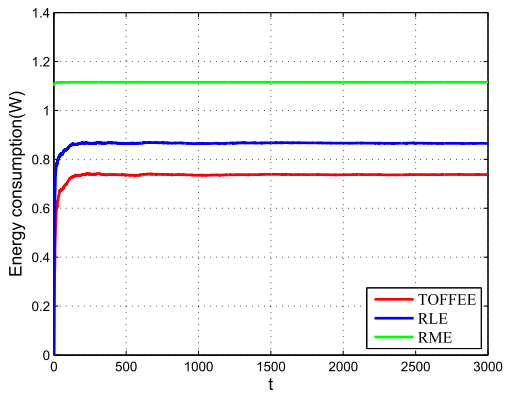


Fig. 6. Energy consumption with the three algorithms.

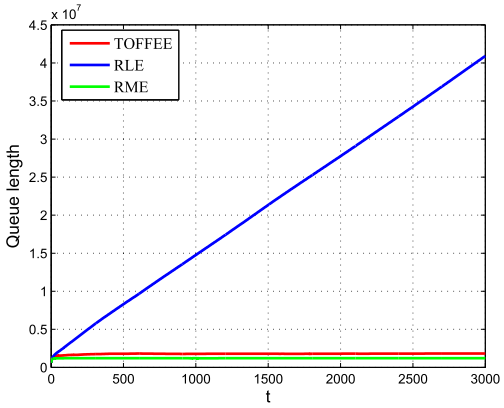


Fig. 7. Queue length with the three algorithms.

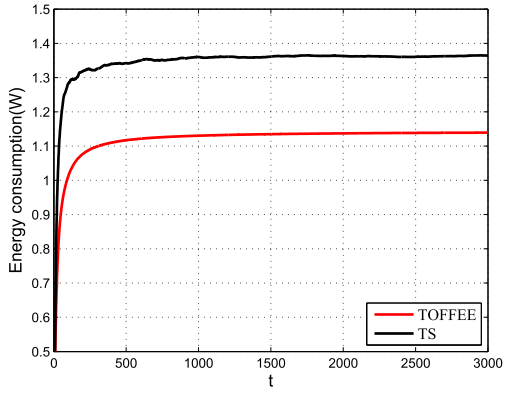


Fig. 8. Energy consumption of TOFFEE and TS.

TOFFEE's execution time is larger than the other two algorithms. It is because that TOFFEE needs to collect the system information (e.g, queue length, wireless channel and tail time) and refers these information to select the optimal decision. Although the execution time of TOFFEE algorithm is the largest, it only takes 0.776(ms) to make task offloading and frequency scaling decisions.

Additionally, to further evaluate TOFFEE's performance, task scheduling (TS) algorithm proposed in [31] is adopted to compare with TOFFEE. Fig. 8 plots the energy consumption of TOFFEE and TS. We can see that the energy consumption of TOFFEE is smaller than that of TS. This is because our TOFFEE optimizes not only the energy consumed by local execution and data transmission, but also the tail energy consumption. Fig. 9 shows the queue length of TOFFEE and TS. We can observe that TOFFEE's queue length is also smaller than that of TS. The reason is that TOFFEE could adaptively tune the task allocation decisions according to the dynamic and changing wireless channel condition.

6 Related Work

Recently, the mobile edge computing has been extensively studied. In [20], Sun et al. focused on the energy-aware mobility management for the MEC system, and formulated a delay minimization problem with the energy consumption constraint. Then, a user-oriented mobility management scheme was designed to address this problem. Dinh et al. [7] determined the task allocation decision and CPU frequency together, and formulated a minimization problem to optimize the device's energy consumption as well as the tasks' execution delay. Ma et al. [32] studied the workload scheduling in the cloud assisted MEC system, and formulated it as an optimization problem to reduce the system cost and delay. Then, an algorithm with linear complexity was designed. Xiao et al. [33] focused on the workload offloading problem in the cooperative fog computing system, and maximized the users' QoE with the power efficiency constraint. These works mainly focused on the short-term performance and assumed that task arrival or wireless channel was static or could be fetched in advance. However, the task arrival and wireless channel in the real environment are all highly dynamic and hardly predicted precisely. Thus, the computation offloading policy should take into consideration of the dynamics of real system environment. In our work, we focus on the system's long-term performance. To capture the highly dynamics and randomness of the task arrival and wireless network, a stochastic optimization problem is proposed in this paper.

Mao et al. [9] focused on the computation offloading for the green MEC system with a single-user. They jointly optimized the transmit power, CPU-cycle frequency and computation offloading decisions to minimize the user's execution cost. Yang et al. [13] worked on the task offloading, and formulated an energy minimization problem with the constraints of computation capability and delay requirement. Based on the artificial fish swarm algorithm, they proposed an offloading scheme to solve this problem. You et al. [18] studied the resource allocation problem for multiuser MEC system, and minimized the mobile devices' energy consumption with the computation latency constraint. Zhang et al. [19] optimized resource allocation decision in the single and multi-cell MEC system. To achieve the energy-delay tradeoff, an iterative search algorithm was proposed. For the energy consumed to offload tasks, the above works only considered the energy consumed by data transmission. The tail energy was not taken into account in these works. However, in the current cellular network, the tail energy is also critical

TABLE 2
Execution Time of Different Algorithms

Algorithm	Execution time (ms)
TOFFEE	0.7760
RLE	0.0166
RME	0.0204

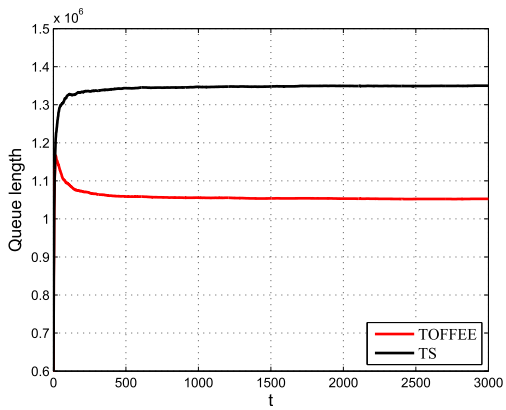


Fig. 9. Queue length of TOFFEE and TS.

for energy-efficient offloading [11], [34]. To deal with this issue, this paper jointly optimizes the transmission energy consumption and tail energy consumption, and seeks to reduce both of them.

7 Conclusion

We have jointly studied the stochastic task allocation and CPU-cycle frequency scaling for the MEC system to minimize the energy consumption while guaranteeing task queue length is upper bounded. We have proposed one stochastic optimization based algorithm, TOFFEE, which requires no statistical information related with the tasks arrival or wireless network states in advance. Theoretical analysis shows that through increasing V's value, TOFFEE can approach the minimum energy consumption arbitrarily and bound the queue length. Experiments demonstrate that TOFFEE can decrease the device's energy consumption effectively and maintain a short queue length.

Appendix A
Proof of Theorem 1

Taking square on (6) and applying $([a-b,0]^+)^2 \le a^2+b^2-2ab$, we have (36).

$$Q_i^2(t+1) \le Q_i^2(t) + A_i(t)^2 + D_i^2(t) - 2Q_i(t)D_i(t) + 2A_i(t)[Q_i(t)-D_i(t),0]^+. \tag{36}$$

Let $\bar{D}_i(t)$ represent the actual amount of tasks which is processed during slot t. And since $A_i(t)$ and $\bar{D}_i(t)$ are non-negative, we have (37)

$$\frac{1}{2}[Q_i^2(t+1)-Q_i^2(t)] \le \frac{1}{2}[A_i^2(t)+D_i^2(t)] + Q_i(t)[A_i(t)-D_i(t)]. \tag{37}$$

Taking conditional expectations on (37) and summing over all the applications, (38) can be obtained.

$$\Delta(\Theta(t)) \le \frac{1}{2}\sum_{i=1}^{n}\mathbf{E}\{A_i^2(t)+D_i^2(t)|\Theta(t)\} + \sum_{i=1}^{n}Q_i(t)\mathbf{E}\{A_i(t)-D_i(t)|\Theta(t)\}. \tag{38}$$

Since $f(t) \le f^{max}$ and $R_i(t) \le R_i^{max}$, we can obtain $D_i^l(t) \le \frac{f^{max}\tau}{\varphi_i}$ and $D_i^r(t) \le R^{max}\tau$. Recall that it holds $D_i(t) = D_i^l(t) + D_i^r(t)$. Then, (39) can be obtained.

$$D_i(t) \le \frac{f^{max}\tau}{\varphi_i} + R^{max}\tau. \tag{39}$$

Exploiting the fact that $A_i(t) \le A_i^{max}$ and applying (39), we have (40).

$$\sum_{i=1}^{n}\mathbf{E}\{A_i^2(t)+D_i^2(t)|\Theta(t)\} \le \sum_{i=1}^{n}\left[(A_i^{max})^2 + \left(\frac{f^{max}\tau}{\varphi_i}+R^{max}\tau\right)^2\right]. \tag{40}$$

Add $V\mathbf{E}\{E_{total}(t)|\Theta(t)\}$ to (38), and let C equal to $\frac{1}{2}\sum_{i=1}^{n}[(A_i^{max})^2 + (\frac{f^{max}\tau}{\varphi_i}+R^{max}\tau)^2]$. And substituting (12), we can obtain (18).

Appendix B
Proof of Theorem 2

We first present Lemma 2 to help derive the upper bounds of $\bar{E}$ and $\bar{Q}$.

Lemma 2. *If the arrival rates of all the applications are within the system capacity region, i.e., $\sum_{i=1}^{n}\lambda_i \in \Lambda$, there exists an optimal strategy $\boldsymbol{\pi}^* = \{\mathbf{D}^{l*}(t), \mathbf{D}^{r*}(t), f^*(t)\}$, which is independent of the queue length and makes the task allocation and CPU-cycle frequency scaling decisions following the fixed distribution. And the optimal strategy $\boldsymbol{\pi}^*$ would satisfy the following,*

$$\mathbf{E}\{\bar{E}^{\boldsymbol{\pi}^*(t)}\} = e^*(\boldsymbol{\lambda});$$
$$\mathbf{E}\{A_i(t)\} \le \mathbf{E}\{D_i^{l*}(t) + D_i^{r*}(t)\},$$

where $e^*(\boldsymbol{\lambda})$ represents the optimal average energy consumption with the $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_n\}$.

Proof. Caratheodory's theorem can be used to prove Lemma 2 [25], and we omit the detail proof here for simplicity. □

Recall that the arrival rate $A_i(t)$ of each application is upper bounded by A_i^{max}. Thus, the energy consumption for transmitting and processing these tasks would also be upper bounded $\hat{E}$ and lower bounded $\check{E}$. Then, by using Lemma 2, Theorem 2 can be proven.

By applying Lemma 2, if it holds that $\sum_{i=1}^{n}(\lambda_i+\epsilon) \in \Lambda$, where $\epsilon > 0$, there exists an optimal strategy π^* which satisfies the following,

$$\mathbf{E}\{\bar{E}^{\pi^*(t)}\} = e^*(\boldsymbol{\lambda}+\epsilon), \tag{41}$$

$$\mathbf{E}\{A_i(t)\} \le \mathbf{E}\{D_i^{l*}(t) + D_i^{r*}(t)\} - \epsilon. \tag{42}$$

Recall that TOFFEE minimizes $\Delta_V(\Theta(t))$'s upper bound. Then, for the optimal strategy π^*, we can obtain that

$$\Delta(\Theta(t)) + V\mathbf{E}\{E^{our}(t)|\Theta(t)\} \le C + V\mathbf{E}\{e^{\pi^*}(t)|\Theta(t)\} + \sum_{i\in\mathbf{N}}Q_i(t)\mathbf{E}\{A_i(t)-D_i^{l*}(t)-D_i^{r*}(t)|\Theta(t)\}. \tag{43}$$

Using (41) and (42), we can convert (43) into the following inequality,

$$\Delta(\Theta(t)) + V\mathbf{E}\{E^{our}(t)|\Theta(t)\} \le C + Ve^*(\boldsymbol{\lambda}+\epsilon) - \epsilon\sum_{i\in\mathbf{N}}Q_i(t). \tag{44}$$

Taking iterated expectations on (44), and summing over the slots, we have (45).

$$\mathbf{E}\{L(\Theta(T))\} - \mathbf{E}\Big\{L(\Theta(0))\} + V\sum_{t=0}^{T-1}\mathbf{E}\{E^{our}(t)\} \le CT + VTe^*(\boldsymbol{\lambda}+\epsilon) - \epsilon\sum_{t=0}^{T-1}\sum_{i\in\mathbf{N}}\mathbf{E}\{Q_i(t)\}\Big\}. \tag{45}$$

For the generality, at the beginning time $t = 0$, assume that the queue lengths of all the applications are equal to zero. Then, it yields that $L(\mathbf{\Theta}(0)) = 0$. In addition, since $Q_i(T) \geq 0$, $\mathbf{E}\{L(\mathbf{\Theta}(T))\}$ would also be non-negative. Thus, (46) can be obtained.

$$\begin{aligned} &V\sum_{t=0}^{T-1}\mathbf{E}\{E^{our}(t)\} \\ &\leq CT + VTe^{\star}(\boldsymbol{\lambda}+\epsilon) - \epsilon\sum_{t=0}^{T-1}\sum_{i\in\mathbf{N}}\mathbf{E}\{Q_i(t)\}. \end{aligned} \tag{46}$$

Since $\epsilon > 0$, it holds that $\epsilon\sum_{t=0}^{T-1}\sum_{i\in\mathbf{N}}\mathbf{E}\{Q_i(t)\} \geq 0$. Then, dividing by VT, it yields (47).

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{E}\{E^{our}(t)\} \leq \frac{C}{V} + e^{\star}(\boldsymbol{\lambda}+\epsilon). \tag{47}$$

Let $T \to \infty$, and $\epsilon \to 0$, (34) can be obtained.

According to (46), we can also have

$$\begin{aligned} &\epsilon\sum_{t=0}^{T-1}\sum_{i\in\mathbf{N}}\mathbf{E}\{Q_i(t)\} \\ &\leq CT + VTe^{\star}(\boldsymbol{\lambda}+\epsilon) - V\sum_{t=0}^{T-1}\mathbf{E}\{E^{our}(t)\}. \end{aligned} \tag{48}$$

Recall that the energy consumption is upper bounded by $\hat{E}$ and lower bounded by $\check{E}$. Then, we have (49).

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i\in\mathbf{N}}\mathbf{E}\{Q_i(t)\} \leq \frac{C + V(\hat{E}-\check{E})}{\epsilon}. \tag{49}$$

Let $T \to \infty$, we have (35).

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61872044, in part by the Key Research and Cultivation Projects at Beijing Information Science and Technology University under Grant 5211823411, and in part by the Supplementary and Supportive Project for Teachers at Beijing Information Science and Technology University under Grant 5111823401.

REFERENCES

[1] D. Chen, N. Zhang, Z. Qin, X. Mao, Z. Qin, X. Shen, and X. Y. Li, "S2M: A lightweight acoustic fingerprints-based wireless device authentication protocol," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 88–100, Feb. 2017.

[2] J. Chen, K. Hu, Q. Wang, Y. Sun, Z. Shi, and S. He, "Narrowband Internet of Things: Implementations and applications," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2309–2314, Dec. 2017.

[3] F. Liu, P. Shu, and J. C. S. Lui, "AppATP: An energy conserving adaptive mobile-cloud transmission protocol," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3051–3063, Nov. 2015.

[4] X. Wang, K. Wang, S. Wu, D. Sheng, H. Jin, K. Yang, and S. Ou, "Dynamic resource scheduling in mobile edge cloud with cloud radio access network," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 11, pp. 2429–2445, Nov. 2018.

[5] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, Aug. 2017.

[6] X. Hu, K. Wong, and K. Yang, "Wireless powered cooperation-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2375–2388, Apr. 2018.

[7] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[8] H. Wu, Y. Sun, and K. Wolter, "Energy-efficient decision making for mobile cloud offloading," in *Proc. IEEE Trans. Cloud Comput.*, 2018, doi: 10.1109/TCC.2018.2789446.

[9] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[10] 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.

[11] Y. Cui, S. Xiao, X. Wang, Z. Lai, Z. Yang, M. Li, and H. Wang, "Performance-aware energy optimization on mobile devices in cellular network," *IEEE Trans. Mobile Comput.*, vol. 16, no. 4, pp. 1073–1089, Apr. 2017.

[12] M. E. T. Gerards, J. L. Hurink, and J. Kuper, "On the interplay between global DVFS and scheduling tasks with precedence constraints," *IEEE Trans. Comput.*, vol. 64, no. 6, pp. 1742–1754, Jun. 2015.

[13] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji, "Mobile edge computing empowered energy efficient task offloading in 5G," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6398–6409, Jul. 2018.

[14] J. Yang, Q. Yang, K. S. Kwak, and R. R. Rao, "Powercdelay tradeoff in wireless powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3280–3292, Apr. 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] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. 2nd USENIX Conf. Hot Topics Cloud Comput.*, 2010, pp. 4–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1863103.1863107

[17] S. M. Ross, *Introduction to Probability Models*. Cambridge, MA, USA: Academic Press, 2014.

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

[19] J. Zhang, X. Hu, Z. Ning, E. C. H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

[20] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Nov. 2017.

[21] W. Hu and G. Cao, "Quality-aware traffic offloading in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3182–3195, Nov. 2017.

[22] K. Wang, K. Yang, and C. Magurawalage, "Joint energy minimization and resource allocation in C-RAN with mobile cloud," *IEEE Trans. Cloud Comput.*, vol. 6, no. 3, pp. 760–770, Jul.–Sep. 2018.

[23] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[24] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," in *Proc. Int. Conf. Mobile Syst. Appl. Serv.*, 2012, pp. 225–238.

[25] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Mateo, CA, USA: Morgan & Claypool, 2010.

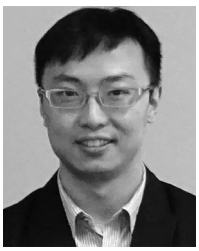
[26] D. Zhang, Z. Chen, L. X. Cai, H. Zhou, S. Duan, J. Ren, X. Shen, and Y. Zhang, "Resource allocation for green cloud radio access networks with hybrid energy supplies," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1684–1697, Feb. 2018.

[27] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[28] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

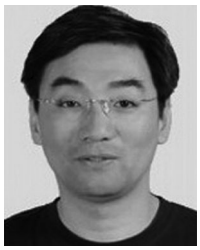
[29] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, "Optimal schedule of mobile edge computing for Internet of Things using partial information," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2606–2615, Nov. 2017.
[30] W. Zhang, Y. Wen, J. Cai, and D. O. Wu, "Toward transcoding as a service in a multimedia cloud: Energy-efficient job-dispatching algorithm," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2002–2012, Jun. 2014.
[31] Z. Jiang and S. Mao, "Energy delay trade-off in cloud offloading for mutli-core mobile devices," in *Proc. IEEE Global Commun. Conf.*, Dec. 2015, pp. 1–6.
[32] X. Ma, S. Zhang, W. Li, P. Zhang, C. Lin, and X. Shen, "Cost-efficient workload scheduling in cloud assisted mobile edge computing," in *Proc. IEEE/ACM 25th Int. Symp. Quality Serv.*, Jun. 2017, pp. 1–10.
[33] Y. Xiao and M. Krunz, "QoE and power efficiency tradeoff for fog computing networks with fog node cooperation," in *Proc. IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.
[34] Y. Geng and G. Cao, "Peer-assisted computation offloading in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4565–4578, Jul. 2018.

Ying Chen received the BEng degree from the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China, in 2012, and the PhD degree in computer science and technology from Tsinghua University, Beijing, in 2017. She is currently an associate professor with Computer School, Beijing Information Science and Technology University, Beijing. She was a joint PhD student in University of Waterloo from 2015 to 2016, and the Google PhD Fellowship Award recipient in 2017.

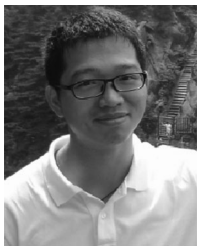


Ning Zhang received the PhD degree from the University of Waterloo, Canada, in 2015. After that, he was a postdoc research fellow at University of Waterloo and University of Toronto, Canada, respectively. He is an assistant professor at Texas A&M University-Corpus Christi. He is a senior member of the IEEE.

Yongchao Zhang received the BEng degree from the Beijing Information Science and Technology University, in 2017. He is currently working toward the master degree at the Beijing Information Science and Technology University.



Xin Chen received the PhD degree from the Beijing Institute of Technology, Beijing, China. He is currently a professor in Beijing Information Science and Technology University.



Wen Wu received the BE degree from the South China University of Technology, Guangzhou, China, in 2012 and the MASc degree from the University of Science and Technology of China, Hefei, China, in 2015. He is currently working toward the PhD degree at the University of Waterloo. He is a student member of the IEEE.

Xuemin (Sherman) Shen received the BSc degree from Dalian Maritime University, Dalian, China, in 1982, and the MSc and PhD degrees from Rutgers University, New Brunswick, New Jersey, in 1987 and 1990, respectively, all in electrical engineering. He is a professor and University Research chair in the Department of Electrical and Computer Engineering, University of Waterloo, Waterloon, ON, Canada. He was the associate chair for Graduate Studies from 2004 to 2008. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, wireless body area networks, and vehicular ad hoc and sensor networks. He received the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award in 2004 and 2008 from the University of Waterloo, the Premiers Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada, and the Distinguished Performance Award in 2002 and 2007 from the Faculty of Engineering, University of Waterloo. He is a distinguished lecturer of the IEEE Communications Society. He is a registered professional engineer of Ontario, Canada, a fellow of the IEEE, Canadian Academy of Engineering, and Engineering Institute of Canada.