Computer Communications 166 (2021) 66–80

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Long-term optimization for MEC-enabled HetNets with device–edge–cloud collaboration

Long Chen [a,*], Jigang Wu [a,*], Jun Zhang [b]

[a] School of Computer Science and Technology, Guangdong University of Technology, China
[b] Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong

ARTICLE INFO

Keywords:
Long-term
Offloading
Edge computing
HetNet
Lyapunov
Collaboration

ABSTRACT

For effective computation offloading with multi-access edge computing (MEC), both communication and computation resources should be properly managed, considering the dynamics of mobile users such as the time-varying demands and user mobility. Most existing works regard the remote cloud server as a special edge server. However, service quality cannot be met when some of the edge servers cannot be connected. Besides, the computation capability of the cloud has not been fully exploited especially when edge servers are congested. We develop an on-line offloading decision and computational resource management algorithm with joint consideration of collaborations between device–cloud, edge–edge and edge–cloud. The objective is to minimize the total energy consumption of the system, subject to computational capability and task buffer stability constraints. Lyapunov optimization technique is used to jointly deal with the delay-energy trade-off optimization and load balancing. The optimal CPU-cycle frequencies, best transmission powers and offloading scheduling policies are jointly handled in the three-layer system. Extensive simulation results demonstrate that, with V varies in $[0.1, 5] \times 10^9$, the proposed algorithm can save more than 50% energy and over 120% task processing time than three existing benchmark algorithms averagely.

1. Introduction

It is reported in the Cisco Visual Networking Index [1] that smartphone traffic will exceed PC-oriented traffic by 2021. On the other hand, the number of machine-to-machine connections is expected to reach 14.6 billion by 2022 due to proliferation of Internet of Things (IoT), smart city applications and cyber physical systems. Wireless heterogeneous networks (HetNets), integrated with cloud computing [2], are serving as an enabling technology to cater for the growing demands in both communications and computations. However, cloud servers are usually far away from mobile devices and can cause tremendous transmission delays during computation offloading. This can harm the quality of service (QoS) of mobile users since intensive user tasks like deep neural networks (DNNs) on mobile devices require quick responses. Meanwhile, the battery power may drift fast due to long communication distance from device to cloud.

Multiple-access edge computing (MEC) serves as a complement for cloud computing can overcome those limitations with edge servers deployed at base stations (BSs), access points (APs) and IoT gateways [3]. Take DNN inference as an example. The data sampled by industrial internet of things devices can be used for DNN inference. With edge–cloud cooperation, the inference service delay can be reduced [4]. In unmanned aerial vehicle (UAV) path planning, the UAV edge base station can assist the computation of deep learning tasks from UAVs for path planning based on the obtained UAVs' sensing data [5]. Both in [4,5] and many other studies, the training or inferring tasks are modeled by arrival bits per time slot, thus it is also adopted by this work.

2. Motivation and contribution

SDN [17] has been well studied in the literature. Recent advances indicate that SDN has been envisioned to be an ideal technique to provide centralized control services. Those services include surveilling of entire network states, scheduling decision making [18], handling data movement between data centers [19] and managing cooperative data sharing in vehicular ad hoc network with heterogeneous network (HetNet) base stations [20]. Similar to [20], we propose a general task offloading framework for MEC in HetNets to handle the above mentioned computation intensive tasks, as shown in Fig. 1. The main control function and computation capability can be decoupled from the macro cell base station to the small cell base stations with edge servers. Meanwhile, the macro cell base station deployed with controller and connected with remote cloud server can still hold a global view to control the whole network, while acting as the computing backups for edge servers. It should be noted that, the SDN controller can also perform

* Corresponding authors.
E-mail addresses: lonchen@mail.ustc.edu.cn (L. Chen), asjgwucn@outlook.com (J. Wu), jun-eie.zhang@polyu.edu.hk (J. Zhang).

https://doi.org/10.1016/j.comcom.2020.11.011
Received 5 June 2020; Received in revised form 4 October 2020; Accepted 17 November 2020
Available online 2 December 2020

Table 1
Related works on multi-access edge computing.

	Delay	Energy	Mobility	MEC	Multi-Edge	3-layer	Edge–Edge
[6]	✓	×	✓	✓	✓	×	×
[7]	✓	×	×	✓	×	×	×
[8]	×	✓	×	✓	×	×	×
[9]	✓	✓	×	×	×	×	×
[10]	✓	✓	×	×	×	×	×
[11]	✓	✓	×	×	×	×	×
[12]	✓	✓	×	✓	×	×	×
[13]	✓	✓	×	✓	✓	×	×
[14]	✓	✓	×	✓	✓	×	×
[15]	✓	✓	✓	✓	✓	×	✓
[16]	✓	✓	×	✓	✓	✓	×
This work	✓	✓	✓	✓	✓	✓	✓

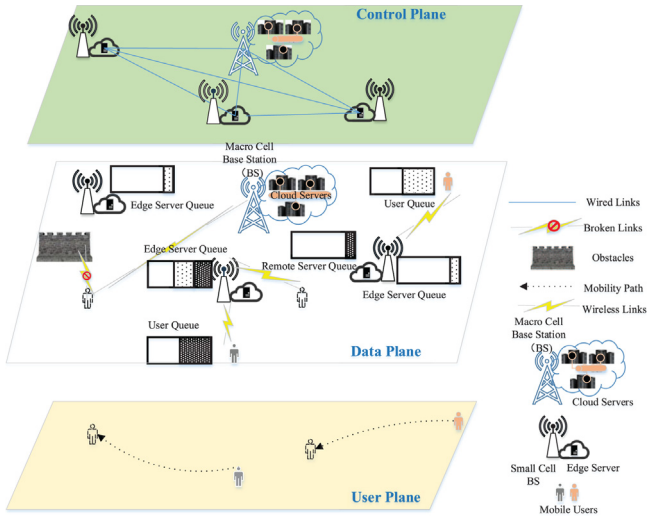


Fig. 1. An MEC-enabled HetNet. The queue lengths of the task buffer at the ith mobile device, the jth edge server and the remote cloud at time slot t are denoted as $Q_i(t)$, $T_j(t)$ and $T_0(t)$ respectively. The network entities in the control plane and user plane are directly adopted from the entities shown in the data plane.

data scheduling and transmission as indicated by [2]. The macro cell base station can connect with small cell base stations through wired links as shown in the control plane of Fig. 1. Meanwhile, the mobile users can connect with both the macro cell and the small cell base stations via OFDMA or TDMA wireless links [21] as depicted in the data plane of Fig. 1. Compared with two layer device–edge structure [22], device–edge–cloud three layer architecture possesses many potential advantages. (I) It guarantees computation connectivity [23] by integrating both edge computing and cloud computing. (II) Computation intensive tasks can be decoupled . For example, the storage and content caching tasks can be performed on the remote cloud at the macro base station while delay sensitive tasks such as authentication can be carried out at the roadside unit [24], e.g., edge servers in a vehicular ad-hoc network (VANET). Besides, when edge servers are congested, the remote cloud server may be used as backups. (III) When edge-to-edge collaboration is adopted, the system performance can be further improved.

There are some works on the optimization [6–8] of MEC systems, however, they are aimed for specific scenarios. Different from VANETs [6], IoT sensor networks [7] or single edge MEC sharing systems [8], where processing delay and system energy consumption are addressed separately, both the delay and energy consumption on mobile devices should be addressed to meet the quality of service demands of mobile applications. For joint optimization, there have been works on delay-energy trade-off in cloud computing systems [9–11] and device–edge involved MEC systems [12,13]. A QoE-aware energy-delay-price joint trade-off scheme was proposed by Hong et al. [12], where tasks from mobile devices can be offloaded to the edge server by selecting distinct service qualities and prices. Different from [12], in [13], tasks can be offloaded to multiple edge servers with the objective to minimize task completion time subject to total energy budget constraint.

Some others address edge–cloud or edge–edge collaborations for MEC systems. Deng et al. [14] developed an edge–cloud collaboration system to minimize energy consumption of devices while meeting the delay requirements. Later, to suit for device-to-device communications, Yang et al. [15] extended [14], where edge nodes can collaborate with each other. In this paper, we extend [15] to a more general case, where there are both edge–edge and edge–cloud collaborations in the network.

The joint device–edge–cloud collaboration based offloading has not been fully investigated in the MEC system, which deserves to be investigated due to the following reasons. First, service continuity cannot be met when some of the edge servers are not able to be connected. For example, due to high channel fading caused by obstacles. Second, the computation capability of the cloud has not been fully exploited especially when edge servers are congested. The constantly dynamic network environment due to channel fading, uncertain user location, changing traffic, etc., makes it more complex to deal with, which is the main focus of this work.

Although Wu et al. [16] designed the joint device–edge–cloud (3-layer) cooperation scheme on delay-energy trade-off, there is only one mobile user and the collaborations among edge servers are not addressed. Besides, edge servers are assumed to possess infinite computing capabilities, which is not practical. In real-life scenarios, there are typically many mobile devices carried by numerous mobile users in different locations. The spatial–temporal influences should be jointly addressed. Different from [16], we consider multiple mobile devices, and there are cooperations among edge servers to fully exploit the available computation edge server resources. Meanwhile, we try to deal with the case when edge servers are also resource constrained and they are shared by multiple heterogeneous applications. The detailed comparisons of related works on MEC are shown in Table 1. The survey [25] also detailed some of the existing computation offloading schemes.

To solve the delay-energy trade-off and load balancing problem, several challenges exist when designing on-line task offloading and scheduling scheme in the three-layer MEC system. Challenge I: The network is dynamic with time. A stochastic model should be constructed to capture the real-time dynamics for such a three-layer MEC system, which desires an efficient on-line scheduling. Challenge II: The proposed MEC offloading scheme should be steady over long-term running. The tasks sent from mobile users should be properly handled based on the real-time conditions in the system and future uncertainty. Challenge III: For the collaboration between edge servers, for one edge server, the admitted tasks from neighboring edge servers should be properly dealt with. That is because for one particular edge server, when choices of re-scheduling the incoming traffic to the remote cloud or to neighboring edge servers coexist, inefficient scheduling may result in longer delay or more energy consumption.

To tackle the above challenges, this paper designs an on-line algorithm for the three-layer offloading MEC system. The main contributions of this paper are:

- We consider a device–edge–cloud collaboration based SDN-MEC framework in HetNet with multiple mobile devices, multiple edge servers and a remote cloud server. With this framework, mobile devices are able to adapt to various network channel conditions.
- Collaborations between device–cloud, edge–edge and edge–cloud are jointly considered. The average weighted sum power consumption represents the energy performance, and the averaged total queue buffer length at mobile nodes represents the delay metric. The optimal CPU-cycle frequency, best transmission power and offloading scheduling policies are jointly handled.

- The optimization problem is modeled as an average weighted sum power consumption minimization problem subject to computing capability, transmission power, neighbor edge serving and system stability constraints. A light-weight on-line Lyapunov optimization algorithm named DECCO is proposed, which is able to deal with the dynamic user offloading requests and does not require future user mobility patterns.
- Extensive performance analysis indicates that the weighted sum power consumption and execution delay obey an $[O(1/V), O(V)]$ trade-off with V as a control parameter. Extensive simulation results demonstrate that, with V varies in $[0.1, 5]\times10^9$, DECCO can save about more than 50% energy and over 120% task processing delay than three existing benchmark algorithms averagely.

The remainder of paper is organized as follows. Section 3 presents the system model and Section 4 gives detailed formulation of the offloading problem. The analytical framework is in Section 5 with detailed solutions in Section 6. The performance analysis is in Section 7 and simulation results are shown in Section 8. Finally, Section 9 concludes this paper with future remarks.

3. System model

We consider an MEC-enabled HetNet, as shown in Fig. 1. Denote $\mathcal{N} = \{1, 2, \dots, N\}$ as a set of N mobile users, with each user carrying a single-core mobile device. Because if there are more than one cores in the model, we can split them into multiple separate devices with each possesses one core. Denote $\mathcal{M} = \{0, 1, 2, \dots, M\}$ as the set of servers running by network operators, where $m = 0, m \in \mathcal{M}$ represents the remote cloud server, while $m \in \mathcal{M}_e$, where $\mathcal{M}_e = \mathcal{M} \setminus \{0\}$ is the set of edge servers. Each MEC server is deployed at an access point (AP) or a small cell base station (SCB), and it is also linked to its neighboring edge servers and the macro cell base station (MCB) via the high-speed local area network. Meanwhile, the MCB is connected with the remote cloud server through high-speed fiber-optic networks. The same as most existing studies, e.g., [26], we treat cloud servers as a whole. Because one dedicated high performance cloud server has the strong computation capability, is equivalent to a number of servers with inferior computation capability. The users are assumed to be mobile while the servers are static. To better describe user mobility and the changing load in the task queues, the system is assumed to operate in a slotted time $t \in \mathcal{T} = \{1, 2, \dots\}$. The available bandwidth is ω shared by mobile users in an OFDMA manner. For ease of illustration, we list the key notations of our system model in Table 2. The same as mostly existing works [27], in this work, the computing results receiving delay is neglected.

3.1. Mobility model

Modeling human mobility is important in MEC networks because many mobile devices are either attached to or manipulated by humans. When users move between edge servers, different traffic loads arises at different edge servers. There are many mobility models, such as random mobility model [28], discrete time Markov mobility model and the follow-me mobility model [29]. Similar to [28], we assume a random mobility model. Denote $L_n(t)$ as the location of mobile user n at time t, then the set of all mobile users is $\boldsymbol{l}(t)$, and the values of $\boldsymbol{l}(t)$ is from the finite location set $\mathcal{L}$. Due to the mobility of mobile users, the vector $\boldsymbol{l}(t)$ is constantly changing with random transition probability $p_{l_i l_j}$, for all l_i, $l_j \in \mathcal{L}$. Knowing that the node-edge–cloud connections vary across different time slots due to mobility of mobile devices, we introduce a connectivity graph for $\mathcal{G} = \{\mathcal{N}, \mathcal{M}, \varepsilon(t)\}$, where $\varepsilon(t) = \{(n, m) | n \in \mathcal{N}, m \in \mathcal{M}, d_{nm}(t) \le d_n^{\max}\}$, where $d_n^{\max}$ is the maximum communication distance of the nth mobile user when communicating with small cell base stations. Without loss of generality, we assume the macro base station has full coverage of all mobile devices and edge servers in the MEC system. As shown in Fig. 2, each

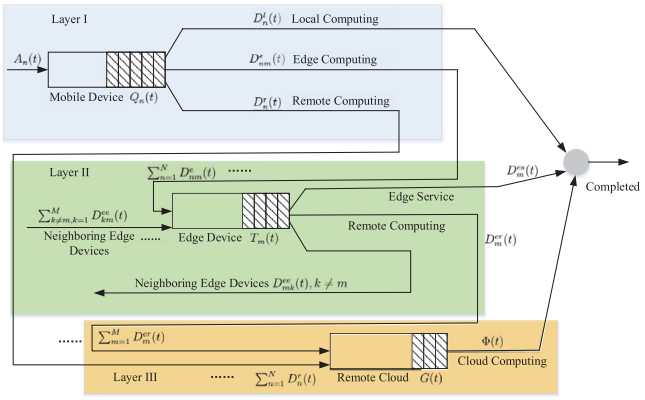


Fig. 2. The detail mathematical model to the collaborative system.

edge server should process the tasks from neighboring edge servers and meanwhile send part of its tasks to its own neighboring edge servers. The computation tasks generated at mobile device side is modeled as a finite size queue with arrival rate of $A_n(t)$. Then the tasks are subdivided for local computing, edge computing and remote computing for further processing with different processing rates. For the particular edge device with queue length of $T_m(t)$, the arrived packets from mobile devices can be processed on itself or being helped by neighboring edge devices. The cloud server acts as backup for all tasks.

3.2. Computation task and task queueing models

Similar to [22], we assume each mobile device is running independent DNN training and inferring tasks at the start of each time slot and cannot be subdivided. The mobile devices can be smartphones or laptop computers with multi-homing technology [30]. At time t, $A_n(t)$ (bits) computation tasks arrive at mobile device n and can be processed from the start of $t+1$ slot. Without loss of generality, we assume $A_n(t) \in [A_{n,min}, A_{n,max}]$ and in different slots, they are independent and identically distributed (i.i.d.) with mean value of $\mathbb{E}[A_n(t)] = \lambda_n$, $\forall n \in \mathcal{N}$.

In each time slot, part of the tasks on the nth mobile device will be executed locally, denoted as $D_n^l(t)$, while $D_{nm}^e(t)$ part will be offloaded to the mth edge server and $D_n^r(t)$ will be executed on the remote cloud through wireless communications. The remained tasks that are not yet processed (locally or offloaded) will be queued at the cache buffer on the mobile device. The queue length vector is denoted as $\boldsymbol{Q}(t) \triangleq [Q_1(t), Q_2(t), \dots, Q_N(t)]$, where $Q_n(t)$ evolves as the following equation:

$$Q_n(t+1) = \max\left\{Q_n(t) - \left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right], 0\right\} + A_n(t), \forall n \in \mathcal{N}, \forall m \in \mathcal{M}_e, \forall t \in \mathcal{T}. \quad (1)$$

In (1), $D_n^l(t) + D_{nm}^e(t) + D_n^r(t)$ is the sum amount of tasks departure from the task buffer of the nth mobile user at time t. The detailed notations can be found in Table 2.

Each MEC server maintains a buffer to store the tasks that have been offloaded by mobile devices or neighboring MEC servers but not yet executed by the server or not yet transmitted to neighboring MEC servers or cloud server. Denote the queues' length vector of the MEC servers at the beginning of slot t as $\boldsymbol{T}(t) \triangleq [T_1(t), T_2(t), \dots, T_M(t)]$, for all $m \in \mathcal{M}$. Further, denote a task scheduling decision of the MEC server at slot t as $B_m(t)$, which is the departure rate of the service queue. Therefore, $T_m(t)$ evolves according to the following equation:

$$T_m(t+1) = \max\left\{T_m(t) - B_m(t), 0\right\} + \sum_{k=1, k\neq m}^{M} D_{km}^{ee}(t)$$

Table 2
Basic Notations.

Notations	Meaning
ω	Available system bandwidth
$\mathcal{N}$	The set of mobile users, where $\mathcal{N} = \{1, 2, \ldots, N\}$
$\mathcal{M}$	The set of servers , where $\mathcal{M} = \{0, 1, 2, \ldots, M\}$
$\mathcal{M}_e$	The set of edge servers, where $\mathcal{M}_e = \mathcal{M} \setminus \{0\}$
$\mathcal{M}_c = \mathcal{M} \setminus \mathcal{M}_e$	The set of remote cloud server
$L_n(t)$	The geographic location of mobile user n at time slot t
$\boldsymbol{l}(t)$	The location vector of all mobile users at time slot t
$\mathcal{L}$	Location set of mobile users, which is finite
$A_n(t)$	The arrival task of the nth mobile user at time slot t, where $n \in \mathcal{N}$ and the unit is bits
$Q_n(t)$	Queue length of the task buffer at the nth mobile device at the beginning of time slot t
$D_n^l(t)$	Part of task of mobile device n that should be executed on the local device n at time slot t
$D_n^e(t)$	Part of task of mobile device n that should be offloaded to the nearby edge server at time slot t
$D_n^r(t)$	Part of task of mobile device n that should be offloaded to the remote cloud server at time slot t
$D_{nm}^e(t)$	The tasks that are originally generated on the nth mobile device are sent to the near edge server m at time t
$T_m(t)$	The task buffer at the mth edge server at time t, $\forall t \in \mathcal{T}$
$\boldsymbol{T(t)}$	The task vector of edge servers at time t, $\forall t \in \mathcal{T}$
$B_m(t)$	The departure rate of the mth edge server at time t, $\forall t \in \mathcal{T}$
$D_{km}^e(t)$	The task coming from neighboring edge server k to the $m$the edge server at time t
$D_m^{es}(t)$	The tasks' executed on the mth edge server
$D_{mk}^{ee}(t)$	Tasks that are shifted from the mth edge server to the kth neighboring edge server, where $k \neq m$ and $k \in \mathcal{M}$ at time t
$D_m^{er}(t)$	The tasks that are further offloaded to the remote cloud by the mth edge server at time t
$G(t)$	The queue length at time t of the remote cloud server
$\boldsymbol{\Phi}(t)$	The departure rate of the cloud server at time t

$$+ \sum_{n=1}^{N} D_{nm}^e(t), \forall n \in \mathcal{N}, \forall m \in \mathcal{M}, \forall t \in \mathcal{T}, \tag{2}$$

the second term of Eq. (2) is the aggregate tasks transmitted by neighboring edge servers, and the third term represents the tasks offloaded by source mobile devices within the vicinity of the mth edge server. The term $B_m(t)$ is defined as follows:

$$B_m(t) = D_m^{es}(t) + D_{mk}^{ee}(t) + D_m^{er}(t), \forall k \neq m, k \in \mathcal{M}, \tag{3}$$

where $D_m^{es}(t)$ is the tasks executed on the mth edge server at time t. D_{mk}^{ee} represents the tasks that are shifted from the mth edge server to the kth neighboring edge server, where $k \neq m$ and $k \in \mathcal{M}$.

For the remote cloud server, denote its queue length at time t as $G(t)$, then we have

$$G(t+1) = \max\left[G(t) - \Phi(t), 0\right] + \sum_{n=1}^{N} D_n^r(t) + \sum_{m=1}^{M} D_m^{er}(t), \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{4}$$

where $\Phi(t)$ is the departure rate of the cloud server at time t. The second term in (4) contains the tasks directly offloaded by mobile users while the third term incorporates the tasks sent by edge servers through wired links. Without loss of generality, we assume that those task buffers are empty initially, *i.e.*, $Q_n(0) = T_m(0) = G(0) = 0$, for all $n \in \mathcal{N}, m \in \mathcal{M}$.

3.3. Local execution model

To process one bit of the input computation tasks from the nth mobile device, W_n CPU cycles will be needed [22]. Let the CPU-cycle frequency of the nth mobile device in time slot t be $f_n(t)$, which cannot exceed the maximum value $f_n^{\max}$. Thus, $D_n^l(t)$ can be expressed as

$$D_n^l(t) = \tau f_n(t) W_n^{-1}, \tag{5}$$

where $\tau = 1$ ms [22]. According to [27], the energy consumption of the local computing on the nth mobile device is given by

$$p_n^l(t) = \kappa f_n(t)^3, \tag{6}$$

where κ is a circuits architecture based parameter following [27].

3.4. Edge server execution model

3.4.1. Computation offloading

We assume the wireless channels between mobile devices and the edge server follows are i.i.d. frequency flat Rayleigh block fading channels. Let the channel fading coefficient from the nth mobile user to the mth edge server in time slot t be $\gamma_{nm}(t)$, which is bounded by $\mathbb{E}\left[\gamma_{nm}(t)\right] \triangleq \bar{\gamma}_{nm} < \infty$. Thus the channel power gain between the nth mobile device and the mth edge server can be represented by

$$\Gamma_{nm}(t) = \frac{\gamma_{nm}(t) g_0}{d_{nm}(t)^{-\alpha}}, \tag{7}$$

where g_0 is the path-loss constant, α is the path-loss exponent, which is usually set as 4 [30] and d_{nm} is the distance between mobile user n and edge server m. Since OFDM is adopted, following Shannon formula, the total amount of task offloaded from the nth mobile user is expressed as

$$D_{nm}^e(t) = \begin{cases} \omega\beta_n(t)\tau \log_2\left(1 + \frac{\Gamma_{nm}(t) P_{tx,n}(t)}{N_0 \omega \beta_n(t)}\right), & \beta_n(t) > 0 \\ 0, & \beta_n(t) = 0. \end{cases} \tag{8}$$

Where $\beta_n(t)$ is the fraction of allocated bandwidth for mobile device n at time t and $P_{tx,n}(t)$ is the transmission power of mobile device n with the maximum value of $P_{n,\max}$. For the benefit of simplicity, denote the bandwidth allocation vector $\boldsymbol{\beta(t)} = \left[\beta_1(t), \beta_2, \ldots, \beta_N(t)\right]$, then $\boldsymbol{\beta(t)} \in \mathcal{B} \triangleq \left\{\boldsymbol{\beta} \in \mathbb{R}_N^+ \,|\, \sum_{n=1}^{N} \beta_n \leq 1, \beta_n \geq \epsilon_B, n \in \mathcal{N}, \quad \epsilon_B \in (0, 1/N)\right\}$. Similarly, let d_{nr} be the distance between the nth mobile device to the macro cell base station, which is attached to the cloud, then the amount of task offloaded from the nth mobile device to the cloud server is expressed as

$$D_n^r(t) = \begin{cases} \omega\beta_n(t)\tau \log_2\left(1 + \frac{\Gamma_{nr}(t) P_{tx,n}(t)}{N_0 \omega \beta_n(t)}\right), & \beta_n(t) > 0 \\ 0, & \beta_n(t) = 0. \end{cases} \tag{9}$$

3.4.2. Edge server scheduling

Edge servers are responsible to admit and compute the tasks sent from mobile users, to retransmit the packets to neighboring edge servers or to the cloud server. Let the circuits architecture based parameter of the edge server be κ_e, denote the CPU-cycle frequency of the mth edge server at time t as $f_{e,m}(t)$, which should not be higher than its maximum value $f_{e,m}^{\max}$. Then the power consumption of the computing on the mth edge server, $p_m^e(t)$ can be expressed as

$$p_m^e(t) = \kappa_e f_{e,m}(t)^3. \tag{10}$$

The CPU cycles offered by the edge server m can be allocated to the computation tasks sent from mobile devices and neighboring edge

servers. Thus, we have

$$\left[\sum_{n=1}^{N} D_{nm}^{e}(t) + \sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right] W_n \leq f_{e,m}(t)\tau, \tag{11}$$

which means the computation tasks from the mobile devices in the vicinity of the mth edge server and from neighboring edge servers cannot exceed the available computation capability of the mth edge server. For the mth edge server, it can process $D_m^{es}(t)$ tasks at time t, which can be expressed as

$$D_m^{es}(t) = \tau f_{e,m}(t) L_m^{-1}, \tag{12}$$

where L_m is the number of CPU cycles consumed at the mth edge server. Generally, L_m is smaller than W_n. The same as [31], we further assume that for any n, m, there exists finite constants $D_{nm}^{e,\max}$ for all t such that

$$0 \leq D_{nm}^{e}(t) \leq D_{nm}^{e,\max}, \tag{13}$$

and for any n, we have the following constraint

$$0 \leq D_n^r(t) \leq D_n^{r,max}. \tag{14}$$

To provide motivation and encourage cooperation between edge servers, we adopt a similar incentive factor δ as [15], which is expressed as:

$$D_{mk}^{ee}(t) \leq \delta D_m^{es}(t), k \neq m, 0 \leq \delta \leq 1, \tag{15}$$

which means the edge server should contribute more while putting less burden onto the neighbors.

3.5. Remote server scheduling

Let L_r be the CPU cycles consumed at the cloud server and $f_r(t)$ be the CPU-cycle frequency at the cloud server at time t, we have

$$\Phi(t) = \tau f_r(t) L_r^{-1}. \tag{16}$$

The CPU cycles offered by the cloud server can be allocated to the tasks from mobile devices and edge servers. Thus, we have

$$\left[\sum_{n=1}^{N} D_n^r(t) + \sum_{m=1}^{M} D_m^{er}(t)\right] W_n \leq f_r(t)\tau, \tag{17}$$

which means the computation tasks directly sent from mobile devices and the edge servers should obey the constraint of computation capability of the cloud server at time t. Let the maximum value of $D_n^r(t)$ and $D_m^{er}(t)$ be $D_n^{r,\max}$ and $D_m^{er,\max}$ respectively, then we have the following constraints

$$0 \leq D_n^r(t) \leq D_n^{r,\max} \tag{18}$$

and

$$0 \leq D_m^{er}(t) \leq D_m^{er,\max}. \tag{19}$$

Similar to (8), the amount of task offloaded from the nth mobile user can be expressed as:

$$D_n^r(t) = \omega \beta_n(t) \tau \log_2\left(1 + \frac{\Gamma_{nr}(t) P_{tx,n}(t)}{N_0 \omega \beta_n(t)}\right), \tag{20}$$

where Γ_{nr} is obtained by replacing m to r in (7). The detail mathematical model to the collaborative system is shown in Fig. 2.

4. Problem formulation

In this section, we will firstly introduce the performance metrics, *i.e.*, the average power consumption of the MEC system and the average sum queue length of the task buffers on the mobile devices. Then the average power consumption minimization problem with user mobility and task queue stability will be formulated.

4.1. Performance metrics

We focus on the energy consumption of the mobile devices, the edge servers and the cloud server. For simplicity, the energy consumption for basic operations of the mobile devices and the computation servers are ignored [22]. Denoting the energy consumption of the nth mobile device at time t as $E_n^l(t)$, we have $E_n^l(t) = P_{tx,n}(t) + p_n^l(t)$. The energy consumption of different edge servers is adopted as $E_m^e(t) = p_m^e(t) + p_m^{e,vm}(t)$, where $p_m^{e,vm}(t)$ is the power consumption for run-time environment migration [32] to enable edge–cloud and edge–edge offloading at time t. For the power consumption of the remote cloud $E^r(t) = \kappa_r f_r(t)^3$, where κ_r is the circuits architecture based parameter of the remote cloud. Therefore, the averaged weighted sum power consumption of the whole MEC system is defined as follows:

$$E_{avg} \triangleq \bar{E} = \lim_{t\to\infty} \frac{1}{T} \mathbb{E}\left\{\sum_{t=0}^{T-1} \sum_{n\in\mathcal{N}} w_n E_n^l(t) + \sum_{t=0}^{T-1} \sum_{m\in\mathcal{M}} w_{N+m} E_m^e(t) + w_{N+M+1} \sum_{t=1}^{T-1} E^r(t)\right\}, \tag{21}$$

where parameters w_n, w_{N+m} and w_{N+M+1} are the weight parameters, which are non-negative real numbers.

According to Little's Law [33], the average data traffic delay experienced by each mobile device in the network system is proportional to the average number of waiting tasks in the task buffer queues of the mobile devices. Thus the average delay metric can be expressed as:

$$D_{avg} \triangleq \frac{\lim_{t\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left\{\sum_{n\in\mathcal{N}} Q_n(t)\right\}}{\sum_{n\in\mathcal{N}} \mathbb{E}\{A_n(t)\}}. \tag{22}$$

4.2. Average weighted power consumption minimization

We assume the system controller is deployed at the macro-cell base station. The system operation profile at time slot t is expressed as $\mathcal{O}(t) \triangleq \left[\mathbf{L}(t), \mathbf{f}(t), \mathbf{P_{tx}}(t), \beta(t), \mathbf{f_e}(t)\right]$. Where $\mathbf{L}(t) \triangleq \left[L_1(t), \ldots, L_N(t)\right]$, $\mathbf{f}(t) \triangleq \left[f_1(t), \ldots, f_N(t)\right]$, $\mathbf{P_{tx}}(t) \triangleq \left[P_{tx,1}(t), \ldots, P_{tx,N}(t)\right]$ and $\mathbf{f}_e(t) \triangleq \left[f_{e,1}(t), \ldots, f_{e,M}(t)\right]$. The average weighted sum power consumption minimization problem can be formulated as $\mathcal{P}1$,

$$\mathcal{P}1: \min_{\mathcal{O}} \bar{E} \tag{23}$$

$$s.t. \quad \boldsymbol{\beta}(t) \in \mathcal{B}, t \in \mathcal{T} \tag{24}$$

$$0 \leq f_n(t) \leq f_{n,\max}, n \in \mathcal{N}, t \in \mathcal{T} \tag{25}$$

$$0 \leq P_{tx,n}(t) \leq P_{tx,\max}, n \in \mathcal{N}, t \in \mathcal{T} \tag{26}$$

$$0 \leq f_{e,m}(t) \leq f_{e,m}^{\max}, m \in \mathcal{M}, t \in \mathcal{T} \tag{27}$$

$$0 \leq f_r(t) \leq f_{r,\max}, t \in \mathcal{T} \tag{28}$$

$$\left[\sum_{n=1}^{N} D_{nm}^{e}(t) + \sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right] W_n \leq f_{e,m}(t)\tau, \tag{29}$$

$$D_{mk}^{ee}(t) \leq \delta D_m^{es}(t), k \neq m, 0 \leq \delta \leq 1, D_{mk}^{ee}(t) \geq 0 \tag{30}$$

$$\left[\sum_{n=1}^{N} D_n^r(t) + \sum_{m=1}^{M} D_m^{er}(t)\right] W_n \leq f_r(t)\tau, D_n^r(t) \geq 0 \tag{31}$$

$$\lim_{T\to\infty} \frac{\mathbb{E}\left[Q_n(t)\right]}{T} = 0, \lim_{T\to\infty} \frac{\mathbb{E}\left[T_m(t)\right]}{T} = 0, \lim_{T\to\infty} \frac{\mathbb{E}\left[G(t)\right]}{T} = 0, \tag{32}$$

where (24) is the bandwidth allocation constraint, while (25) and (26) denote the CPU-cycle frequency and the transmission power constraints of mobile devices. Eq. (27) is the CPU-cycle frequency constraint of edge servers. Eqs. (8) and (30) are MEC server scheduling constraints while (31) is the scheduling constraint of the remote cloud. Eq. (32) ensures that all the arrived computation tasks can be processed in the MEC system within finite delay.

Remark 1. Note that problem $\mathcal{P}1$ is a constrained stochastic optimization problem, where the CPU frequencies of mobile devices and

edge servers, the offloading decisions and the best transmission power should be scheduled at the beginning of each time slot. The requirements of future traffic and available computation resources are hard to obtain due to the mobility of mobile users. Therefore, the problem $\mathcal{P}1$ is hard to solve due to the temporally correlated decisions [22,34].

5. Analytical framework

Lyapunov optimization technique is useful to transfer the time-average constraints into queue stability problems without too much statistical information [35]. With this consideration, we firstly introduce a virtual queue $Z_m^e(t)$ to deal with the service constraint (30) between edge servers and it evolves as follows:

$$Z_m^e(t+1) = \max\left[Z_m^e(t) + D_{mk}^{ee}(t) - \delta D_m^{es}(t), 0\right]. \tag{33}$$

And the virtual queue vector can be defined as $\mathbf{Z^e}(t) \triangleq \left[Z_1^e(t), \ldots, Z_M^e(t)\right]$. The quadratic Lyapunov function for a queue matrix function $\mathbf{\Theta} = \left[\mathbf{Q}(t), \mathbf{T}(t), \mathbf{G}(t), \mathbf{Z^e}(t)\right]$ is

$$L(\mathbf{\Theta}) \triangleq \frac{1}{2}\left[\sum_{n\in\mathcal{N}}\left(Q_n(t)\right)^2 + \sum_{m\in\mathcal{M}}\left(T_m(t)\right)^2 + (G(t))^2 + \sum_{m\in\mathcal{M}}\left(Z_m^e(t)\right)^2\right]. \tag{34}$$

Thus, the conditional Lyapunov drift function $\Delta(\mathbf{\Theta}(t))$ can be written as

$$\Delta(\mathbf{\Theta(t)}) = \mathbb{E}\left[L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\right], \tag{35}$$

which expresses the change of Lyapunov function at time slot t to the next time slot $t+1$. Accordingly, the Lyapunov drift-plus-penalty function can be expressed as

$$\Delta_V(\Theta(t)) = \Delta(\mathbf{\Theta(t)}) + V \cdot \mathbb{E}\left[E(t)\right], \tag{36}$$

where V is a non-negative control parameter used to balance the processing delay and the energy consumption. In the following lemma, we first find an upper bound of $\Delta_V(\Theta(t))$.

Lemma 2. *For any $\mathbf{\Theta}(t)$ that is applied in any time slot t, such that $f_n(t) \in [0, f_{n,\max}]$, $P_{tx,n}(t) \in [0, P_{tx,\max}]$, $n \in \mathcal{N}$, $f_{e,m}(t) \in [0, f_{e,\max}]$, $m \in \mathcal{M}$, $\left[\sum_{n=1}^{N} D_{nm}^{e}(t) + \sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right] W_n \le f_{e,m}(t)\tau$, $\left[\sum_{n=1}^{r} D_n^r(t) + \sum_{m=1}^{M} D_m^{er}(t)\right] W_n \le f_r(t)\tau$, $D_n^r(t) \ge 0$, $D_{mk}^{ee} \ge 0$, $D_n^r(t) \ge 0$, and $\beta(t) \in \mathcal{B}$, $\Delta_V(t)$ is upper bounded as follows:*

$$\begin{aligned}\Delta_V(\mathbf{\Theta(t)}) \le\ & \mathcal{K} - \mathbb{E}\left[\sum_{n\in\mathcal{N}} Q_n(t)\left(D_n^l(t) + D_{nm}^e(t) + D_n^r(t) - A_n(t)\right)\right] + \mathbb{E}\left[G(t) f_r(t)\tau W_n^{-1}\right] \\ &+ \mathbb{E}\left[\sum_{m\in\mathcal{M}} Z_m^e(t)\left(D_{mk}^{ee}(t) - \delta D_m^{es}(t)\right)|\mathbf{\Theta(t)}\right] \\ &+ V \cdot \mathbb{E}\left[E(t)|\mathbf{\Theta(t)}\right],\end{aligned} \tag{37}$$

where $\mathcal{K}$ is a constant and is defined as follows:

$$\begin{aligned}\mathcal{K} =\ & \frac{3}{2}\sum_{n\in\mathcal{N}}\left[\tau^2 f_{n,\max}^2 W_n^{-2} + \left(\frac{\omega\tau^2 \cdot 2\bar{\gamma}_{nm} g_0 P_{tx,\max}}{(\ln 2)^2 d_{nm}^{-\alpha} N_0}\right) + \left(\frac{\omega\tau^2 \cdot 2\bar{\gamma}_{nr} g_0 P_{tx,\max}}{(\ln 2)^2 d_{nr}^{-\alpha} N_0}\right)\right] + \sum_{n\in\mathcal{N}}\frac{A_{n,\max}}{2} \\ &+ \sum\left[\frac{\left(f_{e,m}^{\max}\right)^2 \tau^2 W_n^{-2}}{2} + \frac{f_{e,m}^{\max}\tau W_n^{-1}}{2} \cdot \left(1 + \sum_{n\in\mathcal{N}}\frac{\tau\bar{\gamma}_{nm} g_0 P_{tx,\max}}{(\ln 2) d_{nm}^{-\alpha} N_0}\right)\right] + \frac{f_{r,\max}^2 \tau^2 W_n^{-2}}{2} \\ &+ \sum_{m\in\mathcal{M}}\delta^2\tau^2\left(f_{e,m}^{\max}\right)^2 L_m^{-2}.\end{aligned} \tag{38}$$

Proof. See Appendix A.

The main idea of the proposed Lyapunov optimization based online joint transmission resource and computation resource management algorithm is to minimize the upper bound of Lyapunov drift-plus-penalty $\Delta_V(\mathbf{\Theta(t)})$ in the right-hand size of (75) for each time slot. By doing so, all the task buffers at each side will be kept at a small size, thus the task delay can be bounded. Meanwhile, the energy consumption can be minimized. The proposed algorithm is described in Algorithm 1, where the skeleton to problem solving is summarized.

Algorithm 1 The online control algorithm for joint communication and computation resource allocation

At the beginning of each time slot t, obtain $\{L_n(t)\}$, $\boldsymbol{\Theta(t)}$, $\{\Gamma_{nm}(t)\}$, $\{\Gamma_{nr}(t)\}$, $\{A_n(t)\}$.

Obtain the following parameters: $\mathbf{f}(t)$, $\mathbf{f}_e(t)$, $\mathbf{P_{tx}}(t)$, $\boldsymbol{\beta}(t)$, $\mathbf{f_c}(t)$ by solving the following optimization problem:

$$\begin{aligned}\mathcal{P}2: \min_{\mathcal{O}} & -\sum_{n\in\mathcal{N}} Q_n(t)\left(D_n^l(t) + D_{nm}^e(t) + D_n^r(t) - A_n(t)\right) + G(t) f_r(t)\tau W_n^{-1} \\ & + \sum_{m\in\mathcal{M}} Z_m^e(t)\left(D_{mk}^{ee}(t) - \delta D_m^{es}(t)\right) + V \cdot E(t) \\ s.t.\quad & \boldsymbol{\beta}(\mathbf{t}) \in \mathcal{B} \quad \text{and} \quad (25)\text{–}(31),\end{aligned}$$

where $E(t) \triangleq \sum_{n\in\mathcal{N}} w_n E_n^l(t) + \sum_{m\in\mathcal{M}} w_{N+m} E_m^e(t) + w_{N+M+1} E^r(t)$.

Update $\left\{Q_n(t)\right\}, \left\{T_m(t)\right\}, \{G(t)\}, \left\{Z_m^e(t)\right\}$ according to (1), (2), (4) and (33) respectively.

Update time slot t← t+1.

Remark 3. It should be noted that the objective function of $\mathcal{P}2$, which is shown in Algorithm 1, is consisted of four terms. The first two terms are related to the service delay, the third term is related to the edge scheduling constraint, and the fourth term represents the energy consumption of the MEC system. Generally, the delay and energy consumption are conflict with each other in such a device–edge–cloud three layer MEC system. Minimizing the delay will consume more power, and vice versa. The non-negative Lyapunov control parameter V is able to balance the two factors. Particularly, a larger V will result in less energy consumption, which will result in a larger delay for computation tasks.

6. Optimal solutions

In this section, we will investigate how to derive the optimal solution of the problem $\mathcal{P}2$, which consists of the following sub-procedures in each time slot t: (1) optimal CPU-cycle frequency of local devices; (2) transmission power allocation; (3) optimal scheduling. Then we present the proposed long-term scheduling algorithm.

6.1. Optimal CPU-cycle frequency of local devices

According to the definitions of $D_n^l(t)$ and $p_n^l(t)$, there are two terms in $\mathcal{P}2$ that are proportional to the CPU-cycle frequency of local devices. Therefore, after decoupling $f_n(t)$ from $\mathcal{P}2$ and reformatting the objective function, the subproblem $\mathcal{SP}1$ for the CPU-cycle frequency scaling is obtained as follows:

$$\begin{aligned}\mathcal{SP}1: \min_{f_n(t)} & \sum_{n\in\mathcal{N}}\left[-Q_n(t)\tau f_n(t) W_n^{-1} + V w_n k (f_n(t))^3\right] \\ s.t.\quad & 0 \le f_n(t) \le f_{n,max}, \quad n \in \mathcal{N}.\end{aligned} \tag{39}$$

Since the objective function of $\mathcal{SP}1$ is convex and the constraint of it is also convex, $\mathcal{SP}1$ is a convex optimization problem. Due to this

characteristic, the objective of $\mathcal{SP}1$ can then be decoupled for each $f_n(t)$, and the optimization decision of $f_n(t)$ can be performed on each individual mobile device. As a consequence, the optimal value $f_n^\star(t)$ of $f_n(t)$ can be obtained by

$$f_n^\star(t) = \min\left[f_{n,\max}, \sqrt{\frac{Q_n(t)\tau}{3Vw_n k W_n}}\right], n \in \mathcal{N} \tag{40}$$

6.2. Optimal transmission power and bandwidth

The optimal transmission power $p_{tx,n}^\star(t)$ and bandwidth $\beta_n^\star(t)$ can be obtained by the following sub-problem:

$$\begin{aligned}\mathcal{SP}2: \min_{\beta(\mathbf{t}),\mathbf{P_{tx}}(t)} & -\sum_{n\in\mathcal{N}} Q_n(t)\left[D_{nm}^e(t) + D_n^r(t)\right] \\ & + V\cdot\sum_{n\in\mathcal{N}} w_n P_{tx,n}(t) \\ s.t. \quad & 0 \le P_{tx,n}(t) \le P_{n,\max}, \quad n\in\mathcal{N} \quad \text{and} \quad \boldsymbol{\beta}(\mathbf{t}) \in \mathcal{B}.\end{aligned} \tag{41}$$

Theorem 4. *The optimization problem of $\mathcal{SP}2$ is convex.*

Proof. Define the function $D_n(t)$ with respect to $P_{tx,n}(t)$ as $D_n(t) \triangleq \tau\omega\log_2(1+\Gamma_n(t)P_{tx,n}(t)(N_0\omega)^{-1})$, then it is concave with respect to $P_{tx,n}(t)$. Since $D_{nm}^e(t)$ is a perspective function of $D_n(t)$, that is $D_{nm}^e(t) = \beta_n(t)D_n(t)|_{P_{tx,n}(t)/\beta_n(t)}$, $D_{nm}^e(t)$ is jointly concave with respect to $\beta_n(t)$ and $P_{tx,n}(t)$. Similarly, $D_n^r(t)$ is also concave with respect to $\beta_n(t)$ and $P_{tx,n}(t)$. The constraint is linear and thus is convex. Therefore, $\mathcal{SP}2$ is a convex optimization problem [36].

$$P_{tx,n}^\star(t) = \min\left\{P_{n,\max}, \max\left\{\frac{-I+\sqrt{I^2-4HJ}}{2H}, 0\right\}\right\}, \tag{42}$$

$$H = \Gamma_{nm}(t)\Gamma_{nr}(t)Vw_n\ln 2, \tag{43}$$

$$I = Vw_n\ln 2\cdot N_0\omega\beta_n(t)\left[\Gamma_{nm}(t)+\Gamma_{nr}(t)\right] - 2Q_n(t)\omega\beta_n(t)\tau\Gamma_{nm}(t)\Gamma_{nr}(t), \tag{44}$$

$$J = Vw_n\ln 2N_0^2\omega^2\beta_n^2(t) - Q_n(t)N_0\omega^2\beta_n^2(t)\tau\left[\Gamma_{nm}(t)+\tau\Gamma_{nr}(t)\right]. \tag{45}$$

Firstly, we try to solve problem $\mathcal{SP}2$ by optimizing the transmit power and the bandwidth allocation with Lagrangian method. The convex problem $\mathcal{SP}2$ has a feasible region, which is the Cartesian product of $\{P_{tx,n}(t)\}$ and $\beta_n(t)$. The alternating procedure is guaranteed to converge to the global optimal value according to the BCD method [37] or Guass–Seidel method [38]. For a fixed bandwidth allocation $\{\beta_n(t)\}$, $n \in \mathcal{N}$, the optimal transmission power for mobile devices can be obtained by solving:

$$\begin{aligned}PTX: \min_{\mathbf{P_{tx}}(t)} & -\sum_{n\in\mathcal{N}} Q_n(t)\left[D_{nm}^e(t) + D_n^r(t)\right] \\ & + V\cdot\sum_{n\in\mathcal{N}} w_n P_{tx,n}(t), \\ s.t. \quad & 0 \le P_{tx,n}(t) \le P_{n,\max}, \quad n\in\mathcal{N}.\end{aligned} \tag{46}$$

Since $\mathcal{SP}2$ can be decomposed for the individual execution of mobile devices, the optimal $P_{tx,n}^\star(t)$ can be obtained at the stationary point of $-Q_n(t)\left[D_{nm}^e(t)+D_n^r(t)\right] + V\cdot w_n P_{tx,n}(t)$ or the boundary points, which is given in closed form expressed as (42), where H, I and J are defined as ((43)a), ((43)b) and ((43)c).

6.3. Optimal CPU-cycle frequencies for MEC server

When $\mathbf{f}(t)$, $\mathbf{P_{tx}}(t)$ and $\boldsymbol{\beta}(t)$ have been decoupled from problem $\mathcal{P}2$, we can see, the optimal CPU-cycle frequencies and scheduling policies $D_{mk}^{ee}(t)$ at the MEC server can be determined by solving the following sub-problem:

$$\begin{aligned}\mathcal{SP}3: \min_{\mathbf{f_e}(t)} & \sum_{m\in\mathcal{M}} Z_m^e(t)\left[D_{mk}^{ee}(t) - \delta D_m^{es}(t)\right] \\ & + V\cdot\sum_{n\in\mathcal{N}} w_{N+m}\kappa_e f_{e,m}(t)^3 \\ s.t. \quad & 0 \le f_{e,m}(t) \le f_{e,m}^{\max}, m\in\mathcal{M}, t\in\mathcal{T}, \\ & \left[\sum_{n=1}^{N} D_{nm}^e(t) + \sum_{k=1,k\ne m}^{M} D_{km}^{ee}(t)\right] W_n \le f_{e,m}(t)\tau.\end{aligned} \tag{47}$$

It is not difficult to verify that the sub-problem $\mathcal{P}3$ is a convex problem and standard convex algorithms can be utilized to solve $\mathcal{P}3$. However, due to the unique character of this sub-problem, we can solve the problem more efficiently. Knowing that the objective function of $\mathcal{P}3$ is an increasing convex function with respect to $f_{e,m}(t)$, the objective function will be minimal when D_{mk}^{ee} is set as the minimum value 0, which means no work is further offloaded to neighboring edge servers by the mth edge server. Combining the objective function of (47) and its first constraint, we can derive the optimal CPU-cycle frequency scheduling as follows:

$$f_{e,m}^\star(t) = \min\left\{f_{e,m}^{\max}, \max\left\{\sqrt{\frac{Z_m^e(t)\delta\tau L_m^{-1}}{3Vw_{N+m}\kappa_e}}\right\}, 0\right\}, \tag{48}$$

where $Z_m^e(t)$ is the virtual queue at the edge server m to deal with neighboring loads, defined by (33). It should be noted that, according to (33), $Z_m^e(t) \ge 0$. When there are no computation tasks in the virtual queue, the corresponding edge server will not work. With time elapses, the edge server m will increase its CPU-cycle frequency to handle more tasks since $f_{e,m}(t)$ is in proportion to $Z_m^e(t)$.

Given the optimal values of $P_{tx,n}(t)$ obtained from previous stages, we can obtain $D_{nm}^e(t)$ from (8). Then putting the obtained value $D_{nm}^e(t)$ and the optimal value $f_{e,m}^\star(t)$ into the second constraint of (47), the maximum admitted tasks from neighboring edge servers can be determined, which is no greater than $f_{e,m}^\star(t)\tau/W_n - \sum_{n=1}^{N} D_{nm}^e(t)$. Therefore, we have

$$\sum_{k=1,k\ne m}^{M} D_{km}^{ee}(t) \le \frac{f_{e,m}^\star(t)\tau}{W_n} - \sum_{n=1}^{N} D_{nm}^e(t), \forall m\in\mathcal{M}. \tag{49}$$

To derive the optimal scheduling of the mth edge server D_{mk}^{ee}, where $k \in \mathcal{M}\setminus m$, we have to derive the optimal scheduling of the mth edge server to the remote server. That is because $D_{m,k}^{ee}(t)$, $D_m^{es}(t)$ and $D_m^{er}(t)$ jointly determine the throughput of the mth edge server, *i.e.* the queue length $T_m(t)$. In the next subsection, we first derive optimal strategy of the remote server and then we will re-determine the best scheduling policy of edge servers.

6.4. Optimal CPU-cycle frequency of remote server

The optimal CPU-cycle frequency of the remote server can be determined by solving the following convex optimization problem:

$$\begin{aligned}\mathcal{SP}4: \min_{0\le f_r(t)\le f_{r,\max}} & \sum_{n=1}^{N} G(t)f_r(t)\tau W_n^{-1} + Vw_{N+M+1}\kappa_r f_r(t)^3 \\ s.t. \quad & \left[\sum_{n=1}^{N} D_n^r(t) + \sum_{m=1}^{M} D_m^{er}(t)\right] W_n \le f_r(t)\tau, \\ & D_n^r(t) \ge 0, \forall n\in\mathcal{N}.\end{aligned} \tag{50}$$

It should be noted that, to optimize the total cost, for sub-problem $\mathcal{SP}4$, when $D_n^r(t) = 0$ and $D_m^{er}(t) = 0$ for $\forall n\in\mathcal{N}, m\in\mathcal{M}$, the best strategy is to set $f_r(t) = 0$. Otherwise, the optimal CPU-cycle frequency of the remote server and be determined via

$$f_r^\star(t) = \min\left\{f_{r,max}, \frac{\left[\sum_{n\in\mathcal{N}} D_n^r(t) + \sum_{m\in\mathcal{M}} D_m^{er}(t)\right] W_n}{\tau}\right\}, \tag{51}$$

where $D_r(t)$ and $D_m^{er}(t)$ are bounded by the maximum values (18) and (19) accordingly. In the next subsection, we will derive the optimal scheduling between edge servers.

6.5. Optimal scheduling policy between edge servers

Let $i \in \mathcal{M}$ and $j \in \mathcal{M}$ be two edge servers. Define $x_{ij} = 1$ as the binary variable if their is a link between i and j in the graph $G(\mathcal{M}, \varepsilon)$, where the edge set is $\varepsilon = \{(i,j) | i \in \mathcal{M}, j \in \mathcal{M} \setminus i\}$. To balance the load, we have

$$\mathcal{SP}5: \min \max \left\{ -\left[D_i^{es}(t) + D_{ij}^{ee}(t) + D_i^{er}(t) \right] + \sum_{j \in \mathcal{M} \setminus i}^{M} D_{ji}^{ee}(t) + \sum_{n=1}^{N} D_{ni}^{e}(t) \right\} \tag{52}$$

$$s.t. \sum_{j=1, k \neq i}^{M} D_{ji}^{ee}(t) \leq \frac{f_{e,i}^{\star}(t)\tau}{W_n} - \sum_{n=1}^{N} D_{ni}^{e}(t), \forall i \in \mathcal{M}, \tag{53}$$

$$D_{ij}^{ee}(t) \leq \delta D_i^{es}(t), \forall i \in \mathcal{M}, \forall j \in \mathcal{M} \setminus i \tag{54}$$

$$\sum_{j \in \mathcal{M} \setminus i} x_{ij} \leq 1, \forall i \in \mathcal{M}, \tag{55}$$

$$\sum_{i \in \mathcal{M} \setminus j} x_{j,i} x_{ij} \leq M, \forall j \in \mathcal{M}. \tag{56}$$

Despite the min–max condition in (52), it is a weighted mixed matching problem [39], consisting of one-to-one and many-to-one matching.[1] Where, the objective function without min–max condition is the weight. As proved by [40], the weighted mixed matching problem is NP-hard. Therefore, the original problem presented in (52) is also NP-hard. On the other hand, the sub-problem is a special case to that of the min–max regret assignment problem mentioned in [41], which has been proved to be strongly NP-hard, which in turn proves the NP-hardness of $\mathcal{SP}5$. Similar to [39] and [15], the subproblem $\mathcal{SP}5$ can be modeled as a normalized modular function with same partition matroid constraints of [15] and two additional constraints, *i.e.* the constraints one and two of (52). To effectively solve the sub-problem in an online manner, we propose a greedy algorithm to effectively schedule the work loads between neighboring edge servers. Firstly, the greedy algorithm calculates the queue length of each edge server $T_m(t)$ for all $m \in \mathcal{M}$. For each edge server m, it finds the edge server with the lightest computation workload, *i.e.*, the smallest $T_m(t)$ to hold the maximum outsource task and check the constraints in (52). If constraints are met, the algorithm goes on until all edge servers have been explored.

6.6. The long-term scheduling algorithm

Based on Algorithm 1 and the cascading solutions to the corresponding sub-problems, in this subsection, we propose the long-term scheduling algorithm for the collaborations among mobile devices, edge servers and the remote cloud server. The algorithm is summarized in Algorithm 2. The algorithm can be deployed at the base stations. The time complexity of the proposed algorithm can be $O(t_{\max}(M + N + N \log M))$.

Algorithm 2 Long-term Scheduling for Device–Edge–Cloud Collaborative MEC system (DECCO)

Initialization:
Set $t = 0$, $Q_m(t) = 0$, $T_m(t) = 0$, $G(t) = 0$, $Z_m^e(t) = 0$;
Set control parameter V and the weight w_n, w_{N+m}, w_{N+M+1}; Set $A_n(t) \sim U(0, 5000)$ bits/slot, $n \in \mathcal{N}$;
Set $\omega = 10$ MHz, $W_n = 700$ cycles/bit, $L_m = 600$ cycles/bit, $S = 437.5$ cycles/bit, $P_{n,\max} = 0.5$ W, $N_0 = -174$ dB/Hz, $\tau = 1$ ms, $\kappa = 10^{-27}$, $f_{n,\max} = 1$ GHz, $\kappa_e = 10^{-27}$, $f_{e,m}^{\max} = 1.5$ GHz, $\kappa_r = 10^{-27}$, $f_{r,\max} = 2.5$ GHz, N and M;
while $t < t_{\max}$ **do**
  Determine the optimal CPU-cycle frequency $f_n^{\star}(t)$ according to (40);
  Calculate $D_n^l(t)$ and $p_n^l(t)$ following (5) and (6) respectively;
  Calculate $P_{tx,n}^{\star}(t)$ according to (42);
  Compute $D_n^r(t)$ according to (20);
  Obtain $f_r^{\star}(t)$ from (51) and $f_{e,m}^{\star}(t)$ from (48);
  Obtain $D_{nm}^e(t)$ from (8);
  if $t == 1$ **then**
    for each $m \in \mathcal{M}$ **do**
      set $\upsilon D_{mk}(t) \leftarrow 0$; $\upsilon D_{km}(t) \leftarrow 0$; $D_m^{es}(t) \leftarrow 0$;
      if $\sum_{n \in \mathcal{N}} D_{nm}^e(t) + \upsilon D_{km}(t) - \upsilon D_{mk}(t) - D_m^{es}(t) - D_m^{er}(t) > 0$ **then**
        $D_m^{er}(t) \leftarrow \sum_{n \in \mathcal{N}} D_{nm}^e(t) + \upsilon D_{km}(t) - \upsilon D_{mk}(t) - D_m^{es}(t) - D_m^{er}(t)$
      end if
      Update $T_m(t)$ according to Eq. (2);
    end for
  end if
  $t \leftarrow t + 1$;
  for each $n \in \mathcal{N}$ **do**
    Update $Q_n(t)$ following equation (1);
  end for
  Recompute $D_{nm}^e(t)$, $\sum_{n \in \mathcal{N}} D_{nm}^e(t)$, $D_n^l(t)$, $p_n^l(t)$;
  for each $m \in \mathcal{M}$ **do**
    Update $Z_m^e(t)$ following equation (33);
  end for
  Recompute $f_{e,m}^{\star}(t)$, $D_m^{es}(t)$;
  for each $m \in \mathcal{M}$ **do**
    $\upsilon D_{mk}(t) \leftarrow exprnd(\delta \times D_m^{es}(t))$;
  end for
  Greedy allocate the maximum outsource task to the neighbor edge server with the minimum workload and delete the pair;
  Repeat Step 30 until all the edge servers have been explored;
  Update $G(t)$ following equation (4);
end while

[1] Similar to [15], we assume one edge server can select only one neighbor edge server.

7. Theoretical analysis

This section presents the theoretical analysis for the long-term scheduling for collaborative MEC system. We characterize the performance bound of proposed algorithm under the assumption that there exists an $\epsilon > 0$ and a finite constant E_{ξ} such that for any task scheduling algorithm $\mathbf{\Theta}^{\star}(\mathbf{t})$, the following slater-type conditions hold:

$$\mathbb{E}\{A_n(\mathbf{\Theta}^{\star}(\mathbf{t})) - D_n^l \mathbf{\Theta}^{\star}(\mathbf{t}) - D_{nm}^e(\mathbf{\Theta}^{\star}(\mathbf{t})) - D_n^r(\mathbf{\Theta}^{\star}(\mathbf{t}))\} \leq -\epsilon, \forall n \in \mathcal{N}, \tag{57}$$

$$\mathbb{E}\left\{D_{mk}^{ee}(\mathbf{\Theta}^{\star}(\mathbf{t}))\right\} - \mathbb{E}\left\{\delta D_m^{es}(\mathbf{\Theta}^{\star}(\mathbf{t}))\right\} \leq 0, \forall m, k \in \mathcal{M}, \tag{58}$$

$$\mathbb{E}\left\{E(\mathbf{\Theta}^{\star}(\mathbf{t}))\right\} = E_{\xi}, \tag{59}$$

$$\mathbb{E}\left\{G(\mathbf{\Theta}^{\star}(\mathbf{t}))\right\} = G_{\eta}. \tag{60}$$

It should be noted that according to [35], (57) and (58) are used to ensure the stability of mobile device queue $Q_n(t)$ and the virtual queue $Z_m(t)$. Based on the above slater-type conditions, the average energy consumption bound can be obtained by the following theorem.

Theorem 5. *The long-term optimization algorithm proposed in this paper for the device–edge–cloud collaboration system can achieve the following properties.*
(I) The upper bound of average energy consumption for the system is

$$E_{avg} \leq \frac{\mathcal{K} + \psi}{V} + \frac{1}{\mu} E_{opt}. \tag{61}$$

(II) The upper bound of the average delay for the mobile device is

$$D_{avg} \leq \frac{(\mathcal{K} + \psi) + V(E_{\xi} - E_{opt})}{\epsilon \sum_{n \in \mathcal{N}} \mathbb{E}\{A_n(t)\}}, \tag{62}$$

where $\psi = G_{\eta} f_{r,\max} \tau W_n^{-1}$.

Proof. See Appendix B.

From Theorem 5, it is observed that the average energy consumption of the system is inversely proportional to the control parameter V while the average delay is proportional to V. Therefore, the trade-off between energy and delay exists, which will also be investigated via simulation.

8. Simulation results

8.1. Simulation setup

We use MATLAB version R2016b software to conduct simulations. The simulations are tested and analyzed on a desktop computer having Intel i5-4690 CPU with peak frequency of 3.5 GHz and an 8 GB random access memory. We assume OFDMA or TDMA based spectrum allocation has been deployed so that we can concentrate on the scheduling process. To imitate the random mobility of mobile devices, we assume the position changing probability mentioned in Section 3.1 as $p_{l_i l_j} = 0.2$, and mobile users stay at the vicinity of a small cell base station for about $300 \sim 800$ time slots. The detail simulation codes can be found on GitHub with links https://github.com/ustchen/decco. In simulations, by default we assume $N = 40$ mobile devices are located in a 150 m $\times$ 150 m area, and the maximum communication range between the mobile device and the remote cloud server is set as 60 m while the maximum communication range between the mobile device and the edge server is set as 12 m. There are $M = 6$ edge servers by default. The mobile users follow a random mobility model in the network. Similar to [22], we assume that the small scale fading channel power gains have unit mean. We set $\omega = 10$ MHz, $N_0 = -174$ dBm/Hz, $g_0 = -40$ dB, $d_0 = 1$ m, $\delta = 0.3$, $w_n = 1$, $P_{n,\max} = 0.5$ W, $A_n(t)$ is uniformly distributed within $[0, A_{n,\max}]$, $W_n = 700$ cycles/bit, $n \in \mathcal{N}$, $L_m = 600$ cycles/bit, $m \in \mathcal{M}$ and $L_s = 437.5$ cycles/bit. We set $f_{n,\max} = 1$ GHz, $f_{e,m}^{\max} = 1.5$ GHz and $f_{r,\max} = 2.5$ GHz since edge and remote servers possess multiple cores [22]. In addition, we set $\tau = 1$ ms, $\kappa = \kappa_e = \kappa_r = 10^{-27}$ and the results in all the simulation are averaged for 9000 time slots with slot length of τ.

We first validate the theoretical results derived from Theorem 5 for the proposed algorithm in Fig. 3(a). Four scenarios with $w_{N+m} = 0.01$, $w_{N+M+1} = 0.01$; $w_{N+m} = 0.5$, $w_{N+M+1} = 0.5$; $w_{N+m} = 0.5$, $w_{N+M+1} = 0.01$; and $w_{N+m} = 0.01$, $w_{N+M+1} = 0.5$ respectively are considered. It can be observed that the average energy consumption decreases inversely proportional to V and converges to the upper bound shown in (61). The weight w_{N+M+1} plays a key role on the energy consumption, because the higher weight of the remote cloud server is, the higher average energy consumption will be. Meanwhile, when w_n and w_{N+M+1} are the same, a smaller weight w_{N+m} of the edge server will result in a higher energy consumption because of the slowdown of edge server CPU-cycle frequencies. We then compare the average queue length per user with the changing of control parameter V. The results are depicted in Fig. 3(b). For all the four cases, they show similar delay performance.

By default, we set $w_n = 1$, $w_{N+m} = 0.1$ and $w_{N+m} = 0.5$ because local computing consumes the most energy and usually the price of remote cloud service is higher than edge service, e.g., the WiFi access point is usually free of charge or cost little for mobile users to access edge services and the remote cellular communication usually charges a fee by day or by month.

Intuitively, due to the long distance from mobile devices, the remote cloud server in the three-layer network of this paper introduces longer delay than the two-layer edge server computing [15], thus the delay performance using the proposed algorithm may suffer. Therefore, we address a congestion scenario where the remote cloud server act as a backup for edge servers. Accordingly, the default channel bandwidth between mobile device and the small cell base station is assumed as $\omega\beta_n(t) = 1$ MHz [42] and the bandwidth between mobile device and the macro base station is $\omega\beta_n(t) = 100$ kHz [43,44], when there are crowds of mobile users in the network. Note that, this assumption is also reasonable under the general network architecture. For example, the WiFi up-link speed is usually over 10 times faster than the cellular up-link speed.

For performance comparison, we use the same simulation setting mentioned above and we have modified existing methods to make fair comparison. We consider the following baseline algorithms.

(1) Neighbor Edge: Only local execution, edge computing and the collaboration between edge servers, without the remote cloud computing. For comparison purpose, we have modified existing work [15] based on the network setting proposed in this work. In neighbor edge algorithm, there are edge–edge collaboration, but no cloud computing.

(2) Two-layer Edge [22]: Only local computing and edge computing can happen in the network, there are no remote cloud servers. In [22], the edge server can be viewed as a special cloud computing server. Therefore, we treat it as a cloud computing case by disabling the edge services in our model.

(3) Local Computing (Baseline): All the tasks are executed on the mobile devices, no offloading happens.

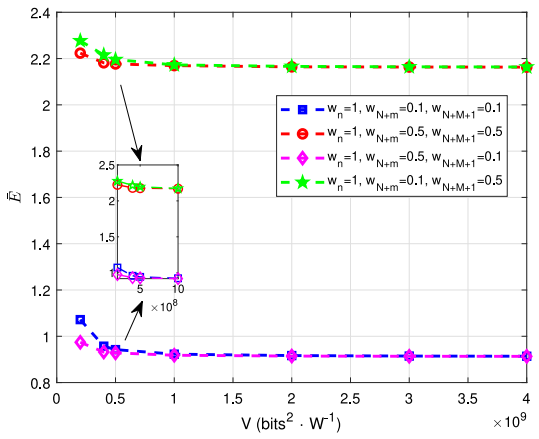
8.2. Energy performance

In Fig. 4, we reveal the average energy consumption performance for different scheduling algorithms. One can observe that the average energy consumption for all cases is inversely proportional to V. Moreover, the two-layer edge computing algorithm [22] demonstrates similar performance with neighbor edge computing algorithm [15] on the average energy consumption. For the default setting $w_{N+M+1} = 0.5$, the proposed DECCO algorithm can save approximately 20%, 16%, and 14% energy than neighbor edge computing, two-layer edge computing and local computing when V is below 1.85. When the weight w_{N+M+1} reduces to 0.3, the performance gains increase to 23.2%, 21.2% and 23.9% for the three algorithms respectively when V is below 3.38. That is because the average energy consumption is proportional to the weight factor w_{N+M+1}. When w_{N+M+1} drops, the average energy consumption will decrease accordingly. When the weight factors $w_{N+m} = w_{N+M+1} = 0.1$, which means edge servers and the remote cloud server have equal influence on the total energy consumption, the DECCO algorithm outperform all the existing algorithms. Thus DECCO can save about 51.5% 50.0% 55.6% energy than neighbor edge computing, two-layer edge computing and local computing algorithms respectively.

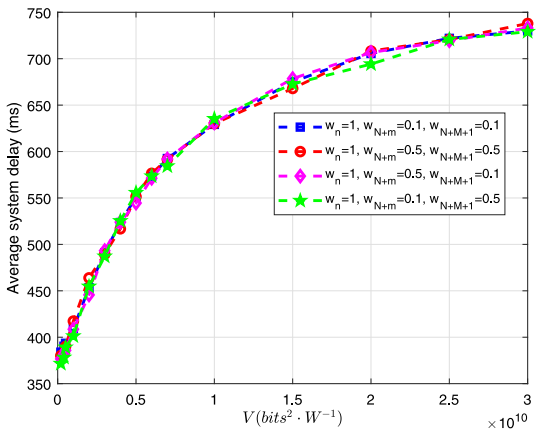
8.3. Delay performance

We next study the delay performance for DECCO with the other three algorithms. It can be observed from Fig. 5 that, the average service delay is proportional to V and DECCO outperforms existing algorithms under any V value. In addition, DECCO can bring the delay reduction by about 120.46%, 120.44% and 122.91% over neighbor edge computing, two-layer edge computing and local computing algorithms for different V values. That is because DECCO tends to put more tasks being executed on the remote cloud server with higher computation capability than edge servers, which results in lower average service delay.

Then, we investigate the influence of the transmission rate between edge server and the remote cloud server (edge–cloud transmission rate) $D_m^{er}(t)$. The results are plotted in Fig. 6 in conjunction with the energy consumption results. It is observed that, the average service delay with high edge–cloud transmission rate is smaller than the case with low edge–cloud transmission rate when V is below 2.75×10^9. That is because the remote cloud has not reached to its computation capability limit when V is lower than 2.75×10^9. When V is greater than 2.75×10^9, the average service delay increases fast for the high edge–cloud transmission rate case. Besides, for all V values, the high edge–cloud transmission rate case produces more energy consumption than the low transmission rate case, due to the increased number of data processed at the remote cloud server.



(a) Average weighted sum energy consumption of the system vs. V, $N = 12$, $M = 6$, $\delta = 0.3$ and $A_{n,\max} = 3$kbits.



(b) Average delay of the system vs. V, $N = 12$, $M = 6$, $\delta = 0.3$ and $A_{n,\max} = 3$kbits.

Fig. 3. The energy and delay performance with control parameter V.

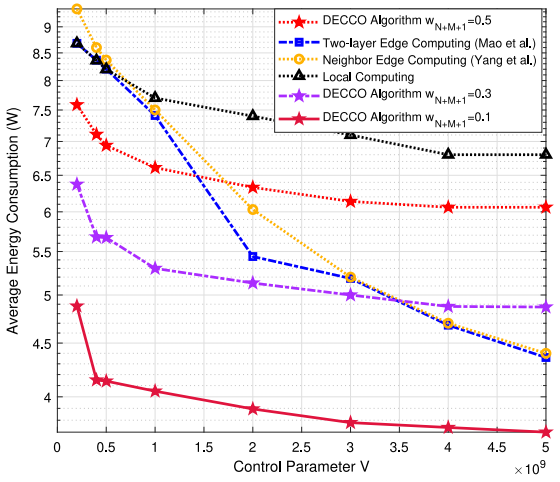


Fig. 4. Average energy consumption with control parameter V.

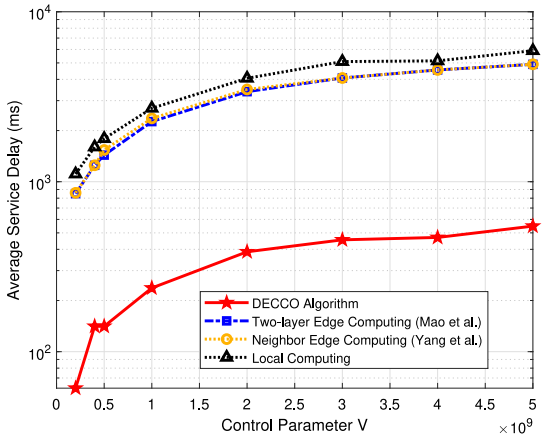


Fig. 5. Average service delay with control parameter V.

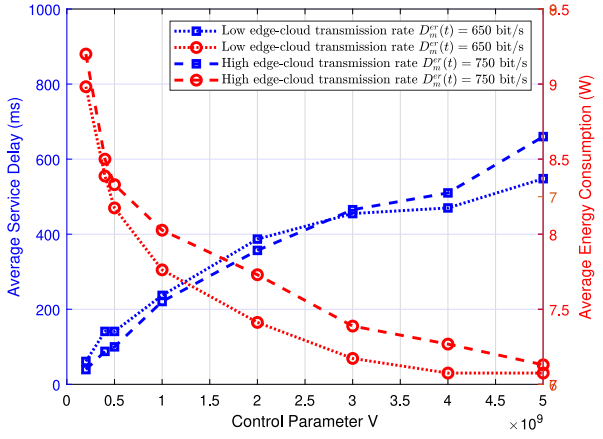


Fig. 6. Average service delay and energy consumption with control parameter V.

8.4. Energy delay trade-off

Figs. 7 and 8 investigate the trade-off between average energy consumption and average service delay of DECCO algorithm, when V increases from 2×10^8 to 2.5×10^{10}. In Fig. 7, the trade-off between average energy consumption and average service delay is studied with different δ, which is the coefficient to allow neighboring processing between edge servers. It is observed that of all the three cases, the energy consumption decreases and service delay increases with the increase of V. Moreover, given a fixed V, DECCO algorithm consumes less energy and obtains shorter delay with a larger δ, averagely. For example, when $V = 10^{10}$, the average energy consumption is 5.25 W and the average delay is 449 ms when $\delta = 0.5$, which is smaller than 5.81 W and 474.5 ms when $\delta = 0.3$. That is because a larger δ implies a larger capability to serve neighbor edge servers, which will improve both energy and delay performance.

Fig. 8 further explores the impact of task arrival rate on the energy-delay trade-offs with fixed $\delta = 0.5$. Similar to Fig. 7, with the V increases, the average energy consumption decreases and the average service delay increases. Therefore, there is an $[O1/V, O(V)]$ trade-off between average energy consumption and average total processing delay. Given a fixed V, a higher task arrival rate A_n for the nth mobile device is, the higher energy and longer delay will be. For example,

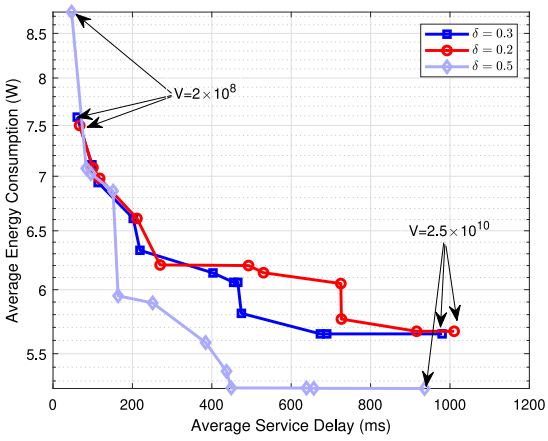


Fig. 7. Average energy consumption with average service delay for changing δ value.

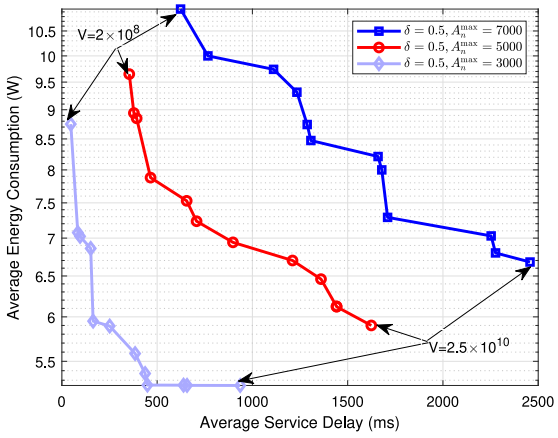


Fig. 8. Average energy consumption with average service delay for changing task arrival rate.

given $V = 4 \times 10^9$, the average energy consumption is 6.94 W and the average delay is 708 ms when the maximum task arrival rate $A_n^{\max} = 5000$ bits/slot, which is lower than 8.21 W and 1659.5 ms when $A_n^{\max} = 7000$ bits/slot. This means that, when there are heavy tasks, more energy and time will be needed to process when system achieves stability.

8.5. Convergence analysis

Figs. 9–11 show the convergence performance of average queue length of mobile devices and edge servers, as well as the remote server. As depicted in Fig. 9, as time passes by, the proposed DECCO algorithm converges faster than the other three algorithms. On average, the maximum accumulated queue length of mobile devices and edge servers for DECCO is about 31.3%, 32.1% and 31.2% shorter than local computing, two-layer edge computing and neighbor edge computing algorithms. The curve of cumulative distribution function (CDF) for average queue length of the edge server is shown in Fig. 10. It can be observed that, for different cases, the CDF values of both neighbor edge computing and DECCO algorithms increase with the increasing number of edge queue length and finally converge to 1. In addition, different from DECCO, the neighbor edge computing algorithm [15] has similar CDF values with different V values. That is because the queue length of edge servers evolve to the similar stable value when V is large for the neighbor edge computing algorithm. However, for DECCO, when the average edge queue length is fixed, a higher V value leads to a higher CDF value. That is because when V becomes larger, the system queue

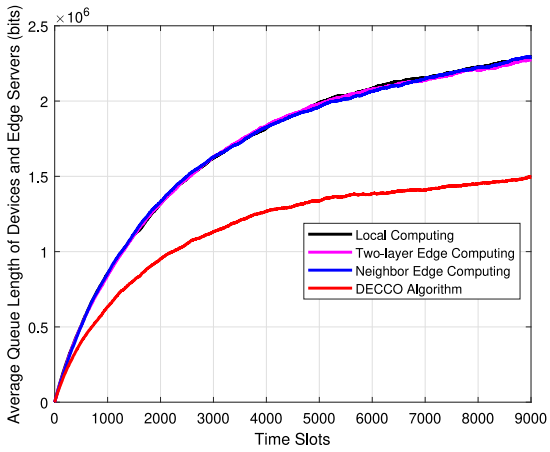


Fig. 9. Average accumulated queue length of mobile devices and edge servers.

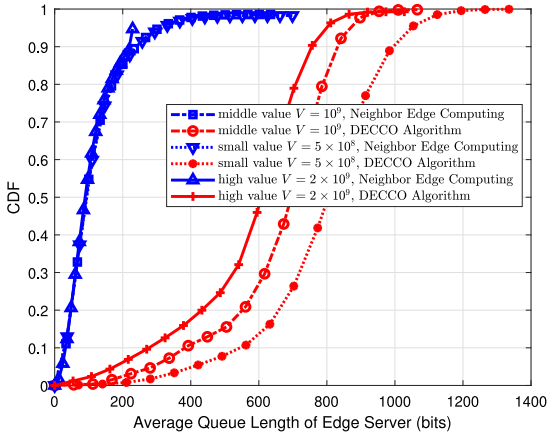


Fig. 10. CDF of the averaged queue length of edge servers.

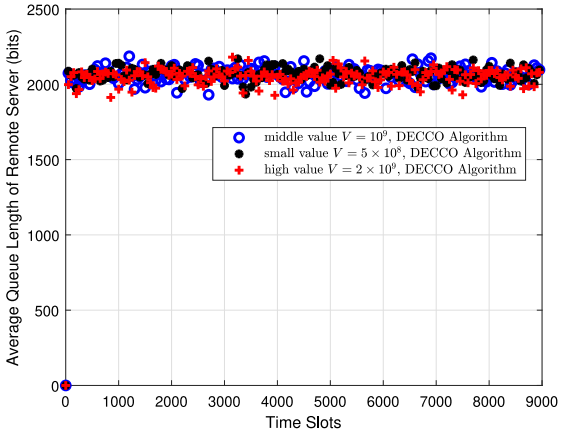


Fig. 11. Average queue length of the remote server with time slots.

length will increase, which means there are more time slots occupied with larger queue length for the larger V case than that of the smaller V case. Finally, Figs. 11 and 12 illustrate the evolution of remote server queue length under different control parameters. Fig. 11 is the sampled figure with sample interval of 50 of total 9000 slots to neatly depict the original results whereas Fig. 12 is the averaged value based on the obtained full data set. It can be drawn from the two figures that, the queue length of the remote server converges sharply for all the cases.

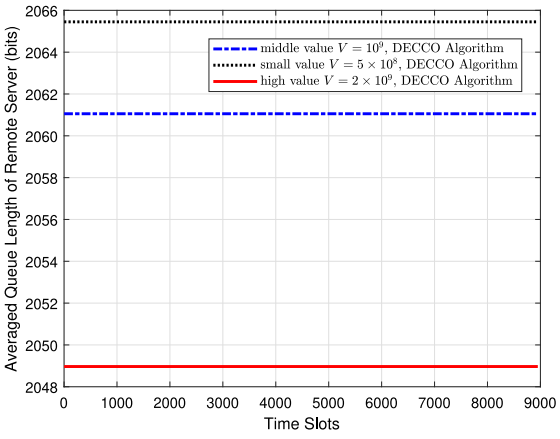


Fig. 12. The averaged queue length of the remote server.

Besides, a smaller V results in a bit larger average queue length of the remote server. That is probably because there have been larger number of subtasks sent from mobile devices.

9. Conclusion

We have investigated a three-layer on-line computation offloading and resource management scheme for the long-term optimization of MEC-enabled HetNets. An efficient on-line algorithm named DECCO is able to deal with the scenario where the up-link transmission between mobile devices and access points deployed with edge servers are congested. Extensive simulation results demonstrated that by carefully choosing the control parameter V and system weight factors, the proposed device–edge-remote MEC architecture outperforms existing two-layer edge computing, neighbor edge computing and local computing algorithms both on the average energy consumption and service delay. In the future, we will conduct real-world experiments to test the performance of proposed schemes. We will also design training algorithms to determine system parameters automatically.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was motivated when the first author was visiting then Prof. Jun Zhang at the department of electronic and computer engineering, the Hong Kong University of Science and Technology. This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 61702115 and 62072118. It was supported by the Hong Kong Research Grants Council under Grant No. 16209418.

Appendix A. Proof for Lemma 2

By squaring both sides of the local computing task buffer dynamics, which is depicted in (1), we have

$$\begin{aligned}
Q_n^2(t+1) &= \left(\max\left\{Q_n(t) - \left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right], 0\right\}\right)^2 + A_n^2(t) + 2A_n(t) \cdot \max\left\{Q_n(t) - \left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right], 0\right\} \\
&\le \left\{Qn(t) - \left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right]\right\}^2 + A_n^2(t) + 2A_n(t)Q_n(t) \\
&= Q_n^2(t) - 2Q_n(t)\left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t) - A_n(t)\right] + \left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right]^2 + A_n^2(t). \qquad (63)
\end{aligned}$$

By moving $Q_n^2(t)$ to the left-hand side of (63), dividing both sides by 2, and summing up the inequalities for $n = 1, \ldots, N$ and $m \in \mathcal{M}$, we have

$$\begin{aligned}
&\frac{1}{2}\sum_{n\in\mathcal{N}}\left[Q_n^2(t+1) - Q_n^2(t)\right] \\
&\le -\sum_{n\in\mathcal{N}} Q_n(t)\left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t) - A_n(t)\right] \\
&+ \sum_{n\in\mathcal{N}} \frac{\left[D_n^l(t) + D_{nm}^e(t) + D_n^r(t)\right]^2 + A_n^2(t)}{2}. \qquad (64)
\end{aligned}$$

Similarly, we can derive the dynamics of the edge server task buffer in (2) as follows:

$$\begin{aligned}
T_m^2(t+1) &= \left(\max\left\{T_m(t) - B_m(t), 0\right\}\right)^2 \\
&+ \left(\sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right)^2 + \left(\sum_{n=1}^{N} D_{nm}^e(t)\right)^2 \\
&+ 2\left(\max\left\{T_m(t) - B_m(t), 0\right\} \cdot \sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right) \\
&+ 2\sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\sum_{n=1}^{N} D_{nm}^e(t) \\
&+ 2\left(\max\left\{T_m(t) - B_m(t), 0\right\}\right) \cdot \sum_{n=1}^{N} D_{nm}^e(t) \\
&\le T_m^2(t) + \left(\sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right)^2 + \left(\sum_{n=1}^{N} D_{nm}^e(t)\right)^2 \\
&+ 2T_m(t)\sum_{n=1}^{N} D_{km}^{ee}(t) + 2\sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\sum_{n=1}^{N} D_{nm}^e(t) \\
&+ 2T_m(t)\sum_{n=1}^{N} D_{nm}^e(t). \qquad (65)
\end{aligned}$$

From constraint (29), we can obtain

$$\begin{aligned}
&\left(\sum_{k=1,k\neq m}^{M} D_{km}^{ee}(t)\right)^2 \\
&\le \left(f_{e,m}(t)\tau W_n^{-1}\right)^2 + \left(\sum_{n=1}^{N} D_{nm}^e(t)\right)^2. \qquad (66)
\end{aligned}$$

Plugging (66) into (65) and moving $T_m^2(t)$ to the left-hand side of (65), we can obtain the following inequality as

$$\begin{aligned}
T_m^2(t+1) - T_m^2(t) &\le f_{e,m}(t)\tau W_n^{-1}\left[f_{e,m}(t)\tau W_n^{-1}\right. \\
&\left. +2\left(1 + \sum_{n=1}^{N} D_{nm}^e(t)\right)\right]. \qquad (67)
\end{aligned}$$

Summing up the inequalities for $m = 1, \ldots, M$, and dividing both sizes by 2, we have

$$\begin{aligned}
&\frac{1}{2}\sum_{m\in\mathcal{M}}\left[T_m^2(t+1) - T_m^2(t)\right] \\
&\le \sum_{m\in\mathcal{M}} \frac{f_{e,m}(t)\tau W_n^{-1}}{2}\left[f_{e,m}(t)\tau W_n^{-1}\right. \\
&\left. +\left(1 + \sum_{n=1}^{N} D_{nm}^e(t)\right)\right]. \qquad (68)
\end{aligned}$$

Similar manipulations for the remote cloud task queue $G(t)$, the dynamics of $G(t)$ is

$$G^2(t+1) = \left(\max\left[G(t) - \Phi(t), 0\right]\right)^2 + \left(\sum_{n=1}^{N} D_n^r(t)\right)^2$$

$$+\left(\sum_{m=1}^{M} D_m^{er}(t)\right)^2 + 2\left[G(t)-\Phi(t),0\right]\cdot\sum_{n=1}^{N} D_n^{r}(t)$$
$$+\ 2\max\left[G(t)-\Phi(t),0\right]\cdot\sum_{m=1}^{M} D_m^{er}(t)$$
$$+\ 2\sum_{n=1}^{N} D_n^{r}(t)\cdot\sum_{m=1}^{M} D_m^{er}(t)$$
$$\le G^2(t)+\left(\sum_{n=1}^{N} D_n^{r}(t)\right)^2+\left(\sum_{m=1}^{M} D_m^{er}(t)\right)^2$$
$$+\ 2G(t)\cdot\sum_{n=1}^{N} D_n^{r}(t)+2G(t)\cdot\sum_{m=1}^{M} D_m^{er}(t)$$
$$+\ 2\sum_{n=1}^{N} D_n^{r}(t)\cdot\sum_{m=1}^{M} D_m^{er}(t). \tag{69}$$

From constraint (31), we have

$$\left(\sum_{m=1}^{M} D_m^{er}(t)\right)^2 \le \left(f_r(t)\tau W_n^{-1}\right)^2+\left(\sum_{n=1}^{N} D_n^{r}(t)\right)^2. \tag{70}$$

Plugging (70) into (69), we obtain

$$\frac{G^2(t+1)-G^2(t)}{2} \le \frac{\left(f_r(t)\tau W_n^{-1}\right)^2}{2}+G(t)f_r(t)\tau W_n^{-1}. \tag{71}$$

For the virtual queue $Z_m^e(t)$, we have

$$\left(Z_m^e(t+1)\right)^2-\left(Z_m^e(t)\right)^2=\left(\max\left[Z_m^e(t)-\delta D_m^{es}(t),0\right]\right.$$
$$\left.+D_{mk}^{ee}(t)\right)^2-\left(Z_m^e(t)\right)^2$$
$$\le \left(D_{mk}^{ee}(t)\right)^2+\left(\delta D_m^{es}(t)\right)^2+2Z_m^e(t)\left(D_{mk}^{ee}(t)-\delta D_m^{es}(t)\right), \tag{72}$$

where for any $Z\ge 0$, $x\ge 0$, $y\ge 0$, we have $(\max[Z-y,0]+x)^2 \le Z^2+x^2+y^2+2Z(x-y)$. Dividing both size of (72) by 2, and summing up the inequalities for $m=1,\dots,M$, we obtain

$$\frac{1}{2}\sum_{m=1}^{M}\left[\left(Z_m^e(t+1)\right)^2-\left(Z_m^e(t)\right)^2\right]\le$$
$$\frac{1}{2}\sum_{m=1}^{M}\left(D_{mk}^{ee}(t)^2+\delta^2 D_m^{es}(t)^2\right)$$
$$+\sum_{m=1}^{M} Z_m^e(t)\left[D_{mk}^{ee}(t)-\delta D_m^{es}(t)\right]. \tag{73}$$

By summing up (64) (68) (71) (73), we have

$$L(\Theta(t+1))-L(\Theta(t))$$
$$\le\sum_{n\in\mathcal{N}}\frac{\left[D_n^l(t)+D_{nm}^e(t)+D_n^r(t)\right]^2+A_n^2(t)}{2}$$
$$+\sum_{m\in\mathcal{M}}\frac{f_{e,m}(t)\tau W_n^{-1}}{2}\left[f_{e,m}(t)\tau W_n^{-1}+(1+\right.$$
$$\left.\sum_{n=1}^{N} D_{nm}^e(t)\right)\Big]$$
$$+\frac{\left(f_r(t)\tau W_n^{-1}\right)^2}{2}+\frac{1}{2}\sum_{m=1}^{M}\left(D_{mk}^{ee}(t)^2+\delta^2 D_m^{es}(t)^2\right)$$
$$-\sum_{n\in\mathcal{N}} Q_n(t)\left[D_n^l(t)+D_{nm}^e(t)+D_n^r(t)-A_n(t)\right]$$
$$+G(t)f_r(t)\tau W_n^{-1}+\sum_{m=1}^{M} Z_m^e(t)\left[D_{mk}^{ee}(t)-\delta D_m^{es}(t)\right]. \tag{74}$$

Since $\log_2(1+x)\le\frac{x}{\ln 2}$ and $\log_2^2(1+x)\le\frac{2x}{(\ln 2)^2}$ for any $x\ge 0$, we obtain $\mathbb{E}\left[\left(D_n^l(t)\right)^2|\mathbf{\Theta(t)}\right] \le \tau^2 f_{n,\max}^2 W_n^{-2}$, $\mathbb{E}\left[D_{nm}^e(t)|\mathbf{\Theta(t)}\right] \le \frac{2\omega\tau^2\bar{\gamma}_{nm}g_0P_{tx,\max}}{(\ln 2)^2 d_{nm}^{-\alpha}N_0}$, $\mathbb{E}\left[\left(D_n^r(t)\right)^2|\mathbf{\Theta(t)}\right] \le \frac{2\omega\tau^2\bar{\gamma}_{nr}g_0P_{tx,\max}}{(\ln 2)^2 d_{nr}^{-\alpha}N_0}$. For any $a,b\ge 0$, we have $2ab\le a^2+b^2$, we then obtain $\mathbb{E}\left[2D_n^l(t)D_{nm}^e(t)|\mathbf{\Theta(t)}\right] \le \mathbb{E}\left[\left(D_n^l(t)\right)^2|\mathbf{\Theta(t)}\right]+\mathbb{E}\left[\left(D_{nm}^e(t)\right)^2|\mathbf{\Theta(t)}\right]$, $\mathbb{E}\left[2D_n^l(t)D_n^r(t)|\mathbf{\Theta(t)}\right] \le \mathbb{E}\left[\left(D_n^l(t)\right)^2|\mathbf{\Theta(t)}\right]+\mathbb{E}\left[\left(D_n^r(t)\right)^2|\mathbf{\Theta(t)}\right]$ and $\mathbb{E}\left[2D_{nm}^e(t)D_n^r(t)|\mathbf{\Theta(t)}\right] \le \mathbb{E}\left[\left(D_{nm}^e(t)\right)^2|\mathbf{\Theta(t)}\right]+\mathbb{E}\left[\left(D_n^r(t)\right)^2|\mathbf{\Theta(t)}\right]$. Finally, by adding $V\cdot E(t)$ at both sides of (74) and taking the expectation on both sides on the condition of $\mathbf{\Theta(t)}$, we can obtain that

$$\Delta_V(\mathbf{\Theta(t)}) \le \mathcal{K}-\mathbb{E}\left[\sum_{n\in\mathcal{N}} Q_n(t)\left(D_n^l(t)+D_{nm}^e(t)\right.\right.$$
$$\left.\left.+D_n^r(t)-A_n(t)\right)\right]+\mathbb{E}\left[G(t)f_r(t)\tau W_n^{-1}\right]$$
$$+\ \mathbb{E}\left[\sum_{m\in\mathcal{M}} Z_m^e(t)\left(D_{mk}^{ee}(t)-\delta D_m^{es}(t)\right)|\mathbf{\Theta(t)}\right]$$
$$+\ V\cdot\mathbb{E}\left[E(t)|\mathbf{\Theta(t)}\right], \tag{75}$$

where $\mathcal{K}$ is defined as follows:

$$\mathcal{K}=\frac{3}{2}\sum_{n\in\mathcal{N}}\left[\tau^2 f_{n,\max}^2 W_n^{-2}+\left(\frac{\omega\tau^2\cdot 2\bar{\gamma}_{nm}g_0P_{tx,\max}}{(\ln 2)^2 d_{nm}^{-\alpha}N_0}\right)\right.$$
$$\left.+\left(\frac{\omega\tau^2\cdot 2\bar{\gamma}_{nr}g_0P_{tx,\max}}{(\ln 2)^2 d_{nr}^{-\alpha}N_0}\right)\right]+\sum_{n\in\mathcal{N}}\frac{A_{n,\max}}{2}$$
$$+\ \sum\left[\frac{\left(f_{e,m}^{\max}\right)^2\tau^2 W_n^{-2}}{2}+\frac{f_{e,m}^{\max}\tau W_n^{-1}}{2}\cdot\left(1\right.\right.$$
$$\left.\left.+\sum_{n\in\mathcal{N}}\frac{\tau\bar{\gamma}_{nm}g_0P_{tx,\max}}{(\ln 2)d_{nm}^{-\alpha}N_0}\right)\right]+\frac{f_{r,\max}^2\tau^2W_n^{-2}}{2}$$
$$+\ \sum_{m\in\mathcal{M}}\delta^2\tau^2\left(f_{e,m}^{\max}\right)^2 L_m^{-2}, \tag{76}$$

which concludes this proof.

Appendix B. Proof of Theorem 5

To obtain the optimal solution $\mathbf{\Theta}^\star(\mathbf{t})$ of problem $\mathcal{P}1$ and meet the slater-type conditions, combining (57) (58) (59) (60) and (36), we have

$$\Delta_V(\mathbf{\Theta(t)}) \le \mathcal{K}+VE_\xi+\mathbb{E}\left\{G(t)f_{r,\max}\tau W_n^{-1}\right\}$$
$$-\mathbb{E}\left\{\sum_{n\in\mathcal{N}} Q_n(t)\epsilon|\mathbf{\Theta(t)}\right\}. \tag{77}$$

Taking the expectations of (77) and combining with the conditional Lyapunov drift function, we obtain

$$\mathbb{E}\left\{L(\mathbf{\Theta}(\tau+\mathbf{1}))\right\}-\mathbb{E}\left\{L(\mathbf{\Theta}(\tau))\right\}+\mathbb{E}\left\{E(t)\right\}$$
$$\le \mathcal{K}+\psi+VE_\xi-\epsilon\sum_{n\in\mathcal{N}}\mathbb{E}\left\{Q_n(t)|\mathbf{\Theta(t)}\right\}. \tag{78}$$

For time slot $t>0$, using the law of telescoping sums, we obtain

$$\mathbb{E}\left\{L(\mathbf{\Theta(t)})\right\}-\mathbb{E}\left\{L(\mathbf{\Theta(0)})\right\}+V\sum_{\tau=0}^{t-1}\sum_{n\in\mathcal{N}}\mathbb{E}\left\{E(t)\right\}$$
$$\le \mathcal{K}t+VE_\xi t-\epsilon\sum_{\tau=0}^{t-1}\sum_{n\in\mathcal{N}}\mathbb{E}\left\{Q_n(t)\right\}+\psi t. \tag{79}$$

Dividing both sides of (79) by ϵt, we have

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{n\in\mathcal{N}}\mathbb{E}\left\{Q_n(t)\right\}\le\frac{(\mathcal{K}+\psi)+V(E_\xi-E_{opt})}{\epsilon}, \tag{80}$$

and substitution into (22), we can obtain the upper bound of average delay of mobile devices. Combining (36) and (79), we have

$$\Delta(\mathbf{\Theta(t)})+V\mathbb{E}\left\{E(t)\right\}\le\mathcal{K}+\psi+\frac{V}{\mu}E_{opt}, \tag{81}$$

where μ is the approximation ratio. Since $\Delta(\boldsymbol{\Theta}(\mathbf{t})) > 0$, we have

$$\mathbb{E}\{E(t)\} \leq \frac{\mathcal{K}+\psi}{V} + \frac{1}{\mu}E_{opt}. \quad (82)$$

References

[1] Cisco, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022 White Paper, 2017, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.pdf [Online; updated on September 15, 2017].

[2] M. Chen, Y. Hao, C.-F. Lai, D. Wu, Y. Li, K. Hwang, Opportunistic task scheduling over co-located clouds in mobile environment, IEEE Trans. Serv. Comput. 11 (3) (2018) 549–561.

[3] L. Chen, J. Wu, X.-X. Zhang, G. Zhou, Tarco: Two-stage auction for d2d relay aided computation resource allocation in hetnet, IEEE Trans. Serv. Comput. (2018) http://dx.doi.org/10.1109/TSC.2018.2792024.

[4] W. Wu, P. Yang, W. Zhang, C. Zhou, S. Shen, Accuracy-guaranteed collaborative DNN inference in industrial IoT via deep reinforcement learning, IEEE Trans. Ind. Inf. (2020).

[5] S. Wan, J. Lu, P. Fan, K.B. Letaief, Towards big data processing in iot: Path planning and resource management of UAV base stations in mobile-edge computing system, IEEE Internet Things J. 7 (7) (2019) 5995–6009.

[6] Y. Dai, D. Xu, S. Maharjan, Y. Zhang, Joint load balancing and offloading in vehicular edge computing and networks, IEEE Internet Things J. (2018).

[7] Y. Mao, J. Zhang, K.B. Letaief, Dynamic computation offloading for mobile-edge computing with energy harvesting devices, IEEE J. Sel. Areas Commun. 34 (12) (2016) 3590–3605.

[8] C. You, K. Huang, H. Chae, B.-H. Kim, Energy-efficient resource allocation for mobile-edge computation offloading, IEEE Trans. Wireless Commun. 16 (3) (2017) 1397–1411.

[9] J. Yang, X. Xu, W. Tang, C. Hu, W. Dou, J. Chen, A task scheduling method for energy-performance trade-off in clouds, in: High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on, IEEE, 2016, pp. 1029–1036.

[10] G. Koutsandria, E. Skevakis, A.A. Sayegh, P. Koutsakis, Can everybody be happy in the cloud? Delay, profit and energy-efficient scheduling for cloud services, J. Parallel Distrib. Comput. 96 (2016) 202–217.

[11] L. Mashayekhy, M.M. Nejad, D. Grosu, Q. Zhang, W. Shi, Energy-aware scheduling of mapreduce jobs for big data applications, IEEE Trans. Parallel Distrib. Syst. 26 (10) (2015) 2720–2733.

[12] S.-T. Hong, H. Kim, Qoe-aware computation offloading to capture energy-latency-pricing tradeoff in mobile clouds, IEEE Trans. Mob. Comput. (2018) http://dx.doi.org/10.1109/TMC.2018.2871460.

[13] M. Chen, Y. Hao, Task offloading for mobile edge computing in software defined ultra-dense network, IEEE J. Sel. Areas Commun. 36 (3) (2018) 587–597.

[14] R. Deng, R. Lu, C. Lai, T.H. Luan, H. Liang, Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption, IEEE Internet Things J. 3 (6) (2016) 1171–1181.

[15] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, J. Wang, Debts: Delay energy balanced task scheduling in homogeneous fog networks, IEEE Internet Things J. 5 (3) (2018) 2094–2106.

[16] H. Wu, Y. Sun, K. Wolter, Energy-efficient decision making for mobile cloud offloading, IEEE Trans. Cloud Comput. (2018) http://dx.doi.org/10.1109/TCC.2018.2789446.

[17] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, ACM SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.

[18] J. Liu, H. Guo, J. Xiong, N. Kato, J. Zhang, Y. Zhang, Smart and resilient ev charging in sdn-enhanced vehicular edge computing networks, IEEE J. Sel. Areas Commun. 38 (1) (2019) 217–228.

[19] G.S. Aujla, N. Kumar, A.Y. Zomaya, R. Ranjan, Optimal decision making for big data processing at edge-cloud environment: An sdn perspective, IEEE Trans. Ind. Inf. 14 (2) (2017) 778–789.

[20] G. Luo, H. Zhou, N. Cheng, Q. Yuan, J. Li, F. Yang, X.S. Shen, Software defined cooperative data sharing in edge computing assisted 5g-vanet, IEEE Trans. Mob. Comput. (2019).

[21] L.P. Qian, Y. Wu, B. Ji, X.S. Shen, Optimal ADMM-based spectrum and power allocation for heterogeneous small-cell networks with hybrid energy supplies, IEEE Trans. Mob. Comput. (2019) http://dx.doi.org/10.1109/TMC.2019.2948014.

[22] Y. Mao, J. Zhang, S. Song, K.B. Letaief, Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems, IEEE Trans. Wireless Commun. 16 (9) (2017) 5994–6009.

[23] J. Liu, N. Kato, H. Ujikawa, K. Suzuki, Device-to-device communication for mobile multimedia in emerging 5g networks, ACM Trans. Multimedia Comput. Commun. Appl. (TOMM) 12 (5s) (2016) 75.

[24] W. Guo, L. Huang, L. Chen, H. Xu, C. Miao, R-mac: Risk-aware dynamic mac protocol for vehicular cooperative collision avoidance system, Int. J. Distrib. Sens. Netw. 9 (5) (2013) 686713.

[25] H. Wu, Multi-objective decision-making for mobile cloud offloading: A survey, IEEE Access (2018) 3962–3976.

[26] M.H. Chen, B. Liang, M. Dong, Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point, in: IEEE Conference on Computer Communications (INFOCOM), 2017, pp. 1–9.

[27] S. Guo, B. Xiao, Y. Yang, Y. Yang, Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing, in: The 35th Annual IEEE International Conference on Computer Communications (INFOCOM), IEEE, 2016, pp. 1–9.

[28] C. Bettstetter, G. Resta, P. Santi, The node distribution of the random waypoint mobility model for wireless ad hoc networks, IEEE Trans. Mob. Comput. (3) (2003) 257–269.

[29] T. Ouyang, Z. Zhou, X. Chen, Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing, in: IWQoS, 2018, arXiv preprint arXiv:1809.05239.

[30] L. Chen, J. Wu, H.-N. Dai, X. Huang, Brains: Joint bandwidth-relay allocation in multi-homing cooperative d2d networks, IEEE Trans Veh Technol 67 (6) (2018) 5387–5398.

[31] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, K.K. Leung, Dynamic service migration and workload scheduling in micro-clouds, in: Proceedings of the 33rd International Symposium on Computer Performance, Modeling, Measurements and Evaluation (IFIP WG 7.3 Performance), 2015.

[32] Y. Song, H. Wang, Y. Li, B. Feng, Y. Sun, Multi-tiered on-demand resource scheduling for vm-based data center, in: Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on, IEEE, 2009, pp. 148–155.

[33] J.D. Little, S.C. Graves, Little's law, in: Building Intuition, Springer, 2008, pp. 81–100.

[34] Y. Mao, J. Zhang, S. Song, K.B. Letaief, Power-delay tradeoff in multi-user mobile-edge computing systems, in: Global Communications Conference (GLOBECOM), IEEE, 2016, pp. 1–6.

[35] M.J. Neely, Stochastic network optimization with application to communication and queueing systems, Synthesis Lectures on Communication Networks 3 (1) (2010) 1–211.

[36] S. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.

[37] M. Razaviyayn, M. Hong, Z.-Q. Luo, A unified convergence analysis of block successive minimization methods for nonsmooth optimization, SIAM J. Optim. 23 (2) (2013) 1126–1153.

[38] L. Grippo, M. Sciandrone, On the convergence of the block nonlinear gauss–seidel method under convex constraints, Operations research letters 26 (3) (2000) 127–136.

[39] D. Wu, L. Zhou, Y. Cai, Social-aware rate based content sharing mode selection for d2d content sharing scenarios, IEEE Trans. Multimed. 19 (11) (2017) 2571–2582.

[40] K. Shanmugam, N. Golrezaei, A.G. Dimakis, A.F. Molisch, G. Caire, Femtocaching: Wireless content delivery through distributed caching helpers, IEEE Trans. Inform. Theory 59 (12) (2013) 8402–8413.

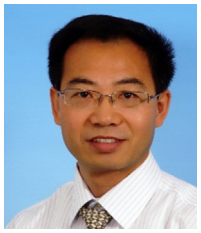
[41] H. Aissi, C. Bazgan, D. Vanderpooten, Complexity of the min–max and min–max regret assignment problems, Operations research letters 33 (6) (2005) 634–640.

[42] A. Khreishah, J. Chakareski, A. Gharaibeh, Joint caching, routing, and channel assignment for collaborative small-cell cellular networks., IEEE J. Sel. Areas Commun. 34 (8) (2016) 2275–2284.

[43] J. Zhang, X. Hu, Z. Ning, E.C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, B. Hu, Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks, IEEE Internet Things J. 5 (4) (2018) 2633–2645.

[44] S.R. Chaudry, A.U. Sheikh, Performance of a dual-rate ds-cdma-dfe in an overlaid cellular system, IEEE Trans. Veh. Technol. 48 (3) (1999) 683–695.

Dr. Long Chen received his Bachelor degree of Engineering in Computer Science from Anhui University, Hefei, Anhui, China in 2011. He received doctoral degree in the University of Science and Technology of China in 2016. Now he is a faculty member in Guangdong University of Technology. His main research interests are mobile computing, game theory, parallel and distributed computing.



Prof. Jigang Wu received B.Sc. degree from Lanzhou University, China in 1983, and doctoral degree from USTC in 2000. He was with the Center for High Performance Embedded Systems, Nanyang Technological University, Singapore, from 2000 to 2010, as a research fellow. He was Dean, Tianjin distinguished professor of School of Computer Science and Software, Tianjin Polytechnic University, China, from 2010 to 2015. Now he is distinguished professor of Guangdong University of Technology. He has published more than 300 papers in the IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Very Large Scale Integration, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Systems, Man, and Cybernetics, and IEEE Transactions on Vehicular Technology -and international conferences. His research interests include network computing, cloud computing, machine intelligence, reconfigurable architecture. He is a member of the IEEE. He serves in China Computer Federation as technical committee member in the branch committees, High Performance Computing, Theoretical Computer Science, and Fault Tolerant Computing.

Dr. Jun Zhang received the B.Eng. degree in Electronic Engineering from the University of Science and Technology of China in 2004, the M.Phil. degree in Information Engineering from the Chinese University of Hong Kong in 2006, and the Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin in 2009. He is an Assistant Professor in the Department of Electronic and Information Engineering at the Hong Kong Polytechnic University (PolyU). Previously he was a Research Assistant Professor with the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST). His research interests include wireless communications and networking, mobile edge computing and edge learning, distributed learning and optimization, and big data analytics. Dr. Zhang co-authored the book Fundamentals of LTE (Prentice-Hall, 2010). He is a co-recipient of the 2016 Marconi Prize Paper Award in Wireless Communications (the best paper award of IEEE Transactions on Wireless Communications), the 2014 Best Paper Award for the EURASIP Journal on Advances in Signal Processing, an IEEE GLOBECOM Best Paper Award in 2017, an IEEE ICC Best Paper Award in 2016, and an IEEE PIMRC Best Paper Award in 2014. Two papers he co-authored received the Young Author Best Paper Award of the IEEE Signal Processing Society in 2016 and 2018, respectively. He also received the 2016 IEEE ComSoc Asia-Pacific Best Young Researcher Award. He is a Senior Member of IEEE, an editor of IEEE Transactions on Wireless Communications and Journal of Communications and Information Networks. He also served as a MAC track TPC co-chair for IEEE Wireless Communications and Networking Conference (WCNC) 2011.