Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-020-00880-y

Revenue-optimal task scheduling and resource management for IoT batch jobs in mobile edge computing

Jiwei Huang[1,2] · **Songyuan Li**[3] · **Ying Chen**[4]

Received: 31 August 2019 / Accepted: 16 January 2020

Abstract

With the growing prevalence of Internet of Things (IoT) devices and technology, a burgeoning computing paradigm namely mobile edge computing (MEC) is delicately proposed and designed to accommodate the application requirements of IoT scenario. In this paper, we focus on the problems of dynamic task scheduling and resource management in MEC environment, with the specific objective of achieving the optimal revenue earned by edge service providers. While the majority of task scheduling and resource management algorithms are formulated by an integer programming (IP) problem and solved in a dispreferred NP-hard manner, we innovatively investigate the problem structure and identify a favorable property namely totally unimodular constraints. The totally unimodular property further helps to design an equivalent linear programming (LP) problem which can be efficiently and elegantly solved at polynomial computational complexity. In order to evaluate our proposed approach, we conduct simulations based on real-life IoT dataset to verify the effectiveness and efficiency of our approach.

Keywords Internet of Things (IoT) · Mobile edge computing · Task scheduling · Resource management · Revenue-optimal

1 Introduction

The Internet of. Things (IoT) is a promising technical field in recent years, which interconnects a variety of senors and other IoT devices over the network enabling themselves to cooperate with each other and achieve common goals [29]. With the increasing amount and types of IoT devices joining the network, several IoT technical requirements are newly put forward. Massive data generated by IoT devices at the frontend brings challenges for efficient information processing, especially for the particular scenario requiring real-time data handling (e.g., Internet of Vehicles [10]). Furthermore, much more intelligent and powerful processing capacity should be equipped at the edge, and thus helps to provide various IoT devices (e.g., smart phones, GPS traker, mobile camera, smart bands, etc.) with diverse services. Therefore, it necessitates a burgeoning computing paradigm designed for IoT business and applications, called edge computing.

Edge computing gains much more momentum with the proliferation of IoT [34]. According to the technical survey from International Data Corporation (IDC) [39], the global data generated by IoT devices will be summed up to 180 zettabytes (ZB) by 2025, of which 70% should be processed at the edge of network. In an edge computing system, job requests as well as generated data from IoT devices is served and processed proximally at the edge server. The edge server is located closer to the IoT device than the centralized cloud data center, thereby shortening the job response time.

This article is part of the Topical Collection: *Special Issue on Emerging Trends on Data Analytics at the Network Edge*
Guest Editors: Deyu Zhang, Geyong Min, and Mianxiong Dong

✉ Ying Chen
chenying@bistu.edu.cn

Jiwei Huang
huangjw@cup.edu.cn

Songyuan Li
lisy@bupt.edu.cn

[1] Department of Computer Science and Technology, China University of Petroleum, Beijing 102249, China

[2] Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum, Beijing 102249, China

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

[4] Computer School, Beijing Information Science and Technology University, Beijing 100101, China

Published online: 06 March 2020

IoT devices are dispersed in wide area, whose locations are time-varied with device mobility. The set of accessible edge servers of each IoT device depends on the current location of the IoT device and the coverage radius of edge base stations. With multiple edge servers deployed geographically, job requests proposed by various IoT devices will be transmitted to and served at one of their accessible edge servers, according to a well-designed scheduling strategy.

Since the era of cloud computing, task scheduling and resource management have always been a high-profile issue in the academic as well as industrial community [3] [24]. It mainly focuses on how to schedule the incoming job requests across multiple resource units of a cloud cluster, and there is no difference on scheduling costs when dispatching jobs to various resource units. Nevertheless, the problem of task scheduling and resource management looks different in the context of edge computing, summarized as the following two points.

- *Uneven ross-layer data transmission latency*: Each IoT device can always have multiple accessible edge servers with heterogeneous data transmission distances. Because of heterogeneous data transmission distances, latency on data transmission to different edge server varies. Therefore, uneven data transmission latency needs to be taken into account when determining which of edge servers should be dispatched to the job request [5, 6]. It would incur less scheduling costs on data transmission if the job request is scheduled to a closer edge server. Otherwise, it would cost more on data transmission.
- *Asynchrony in cross-layer job dynamics*: The principle of First-Come-First-Served is a simplified but widely recognized assumption in the scheduling design. According to the principle of First-Come-First-Served, the job request proposed earlier should be queued ahead and processed in prior. In the edge computing system, however, heterogeneity of data transmission latency results in the dissatisfaction of First-Come-First-Served. Thanks to the shorter data transmission latency, the job request proposed later than other IoT devices may be arrived at the dispatched edge server earlier, further holding the service priority. It embodies the asynchrony in cross-layer job dynamics, which makes the problem of task scheduling intractable in the edge environment.

In this paper, we put forward a cross-edge task scheduling and resource management algorithm for edge computing, whose objective is to maximize the edge providers' revenue. Each IoT device proposes a batch job request formed by several tasks, together with the associated budget. Through our task scheduling and resource allocation algorithm, edge providers can make their efforts to accomplish all of the tasks within job request, further earning the maximum revenue from job budgets.

The discrete nature of our task scheduling and resource management problem makes it challenging to solve out. Task scheduling problem is essentially a combinatorial problem while the resource capacity of edge server is specified by the number of instances. Hence, our problem is inherently an integer programming (IP) problem, which is generally NP-hard [16]. In order to solve out our problem efficiently, we find out the favorable property of totally unimodular constraints [16], which provides integral optimum guarantee for linear programming (LP). Thanks to this favorable property, we take advantage of $\lambda-$technique [26] to conduct equivalent LP transformation of our original IP problem. The effectiveness and efficiency of our proposed approach is validated through simulations based on real-life IoT dataset.

The original contribution of our work is threefold as follows.

- According to the layered structure of edge computing paradigm, we formulate the cross-edge job processing framework in an edge-enabled IoT system. Computational resource capacity of edge servers, heterogeneity of job requests, overlapped signal coverage of edge base stations and device mobility are fully taken in consideration.
- On the basis of system formulation, a cross-edge task scheduling and resource management algorithm is carefully designed, which achieves the optimal revenue earned by edge providers. In virtue of discrete characteristics of original problem formulation, we conduct equivalent LP transformation by means of totally unimodular constraints [16] and $\lambda-$technique [26]. Therefore, IoT jobs can be scheduled efficiently.
- We conduct simulations based on a real-world dataset of base stations and end users in the Melbourne CBD area [17] to verify our proposed approach. Besides, the effectiveness of our scheduling algorithm is demonstrated with a comparison with state-of-art polices.

The remainder of our paper is organized as follows. In Section 2, we survey the related work most pertinent to this paper. In Section 3, the fundamental notations of edge computing model are given, based on which the problem of task scheduling and resource management is formulated. Section 4 firstly introduces the overall framework of our task scheduling and resource management algorithm, and then conducts equivalent LP transformation aimed to solve our problem with high efficiency. In Section 5, we perform experiments based on real-life IoT dataset to verify

our algorithm. In Section 6, we finally summarize our research work in our paper and make a outlook of our future work.

2 Related work

2.1 Peer-to-peer network

Different from the traditional client-server (CS) model, peer-to-peer (P2P) networking is a distributed computation model in which interconnected peers free up part of their resources (e.g. CPU core, disk storage, network bandwidth, etc.) to share amongst each other without the need of centralized servers. Peers are not only consumers but also providers of resources.

The interactive mechanism amongst peers in P2P networks has been extensively studied in the literature. Guo et al. [9] promoted the sustainability of body sensor network (BSN) through proposing a data transmission scheduling and energy harvesting strategy, where sensor states, data dynamics, and energy dynamics were fully taken care of. Kim et al. [15] improved the efficiency of data forwarding for P2P networks by effectively making prediction of connection status using Bayesian classifier, furthering reducing the energy consumption of data transmission. Satsiou et al. [28] concentrated on the problem of fair resource allocation amongst peers based on peers' reputation maintenance, where each peer can only obtain shared resources analogous to its contribution. From the perspective of social networks, Chen et al. [4] eliminated the malicious peers in P2P file sharing networks, and made the friendly peers collaboratively achieving the common goals through reputation querying. Lee et al. [18] studied the problem of reputation management with blockchain technology applied, where peers' reputation can be self-maintained without the use of a central node responsible for reputation management. Kim et al. [14] took the advantage of ML technology to remove the bad data from untrusted peers in P2P networks and further improve the service reliability.

2.2 Edge computing

With the rapid development of IoT, an emerging computing paradigm namely edge computing has been put forward. The architecture of edge computing is designed by learning the features of P2P networking. Ramachandran et al. [27] characterized the edge as a peer of cloud. The functionality of cloud was separated into multiple edge sites deployed close to the IoT device. Job requests from IoT devices were scheduled to and served at the edge site, shortening the job response time and saving the network bandwidth for remote data transmission.

The problem of task scheduling in edge environment has been a hot topic in recent years. Lu et al. [22] analyzed the geographic routing problem in mobile vehicular network, where data link error rate and vehicle density of streets were fully considered. Wang et al. [32] focused on the problem of multi-task scheduling in hybird edge-cloud environment, where several critical issues like profit, deadline and dependence were taken into account. Zhao et al. [38] proposed a dynamic approach for cross-edge task scheduling which optimized the total system cost and energy efficiency of device. Yang et al. [35] regulated the procedure of live virtual-machine (VM) migration in hybrid edge-cloud system with the optimization objective of maximum the average QoS. Ma et al. [23] comprehended the multi-vehicle task offloading problem in edge computing as a strategic game, and achieved the cost-effective and incentive-compatible scheduling scheme when a Nash equilibrium solved out. With the techniques of Lyapunov optimization and Vickrey-Clarke-Groves (VCG) auction, Zhang et al. [37] designed an online computation offloading mechanism in edge-computing systems optimizing the long-term reward. Wang et al. [31] formulated the problem of computation offloading in multi-access edge environment as a deep sequential model based on reinforcement learning, enabling the ability to automatically discover the common patterns behind various applications. Teng el. at [30] studied the task scheduling problem from the perspective of IoT service providers through focusing on code dissemination in vehicular sensor networks, improving the coverage and cost of disseminating process. Zhang et al. [36] optimized related parameters to network resources in IoT network from two different time scales, and ultimately reduced the energy consumption resulting from network communication between IoT devices.

We have done several research works previously on the topic of edge computing. In [12], we applied stochastic queueing theory to provide delicate performance modeling and analysis for IoT services in hierarchical edge computing paradigm. In [20], we formulated the task scheduling problem in edge computing system with Markov Decision Process (MDP) to balance the tradeoff between QoS criteria and energy costs incurred by the computational workload in edge servers. In [11], we put forward a simulation-based framework for QoS-aware dynamic service selection in MEC systems, where the concept of goal softening and techniques of ordinal optimization were introduced to make our approach practical in large-scale systems. Differing from our previous works, this paper studies the task scheduling and resource management problem in MEC environment, taking into account the QoS, mobility and signal coverage, in order to maximize the revenue of the edge servers in the MEC system.

3 System model and problem formulation

3.1 Fundamental notations

System overview We consider an MEC system for IoT services, which consists of N mobile IoT devices and M edge servers. The time horizon is discretized into time slots with the length of τ, indexed by t. Each mobile IoT device moves across time slots, and arbitrarily decides whether to propose the job request at each time slot τ. The job request from mobile IoT devices is firstly pre-processed in local, then is transmitted to the dispatched edge servers for remote execution.

Tasks within the job request are regulated to be scheduled to edge servers at each decision time slot Γ. Resources allocated to the job request are also determined while scheduling. System operator is responsible to configure the length of decision time slot Γ based on the actual situation, which may cover one or more units of time slot τ. At each decision time slot Γ, task requests proposed by mobile IoT devices will be scheduled and dispatched to the edge site, constrained by the signal coverage of edge base stations and resource capacity of edge servers.

Figure 1 describes the processing architecture and dynamics of MEC system. Each mobile IoT device moves according to its own trajectory (i.e., blue, pink, or green arrowed line with different dash types in Fig. 1). The black arrowed dotted line in Fig. 1 points to the allocated edge server, while the gray broken line represents the signal coverage boundary of each edge base station.

Edge server Multiple edge servers are geo-distributed over the system, each of which is usually a micro data center placed near a base station. The base station adjacent to the edge server is called edge base station. IoT business and applications are deployed within the edge server, serving for job requests submitted by mobile IoT devices. The set of M edge servers is denoted by $\mathcal{M}$, and each of edge server is indexed by j.

There are totally A_j resource instances equipped within the edge server j. At each time slot t, $R_j(t)$ resource instances are supplied for incoming job requests. Since part

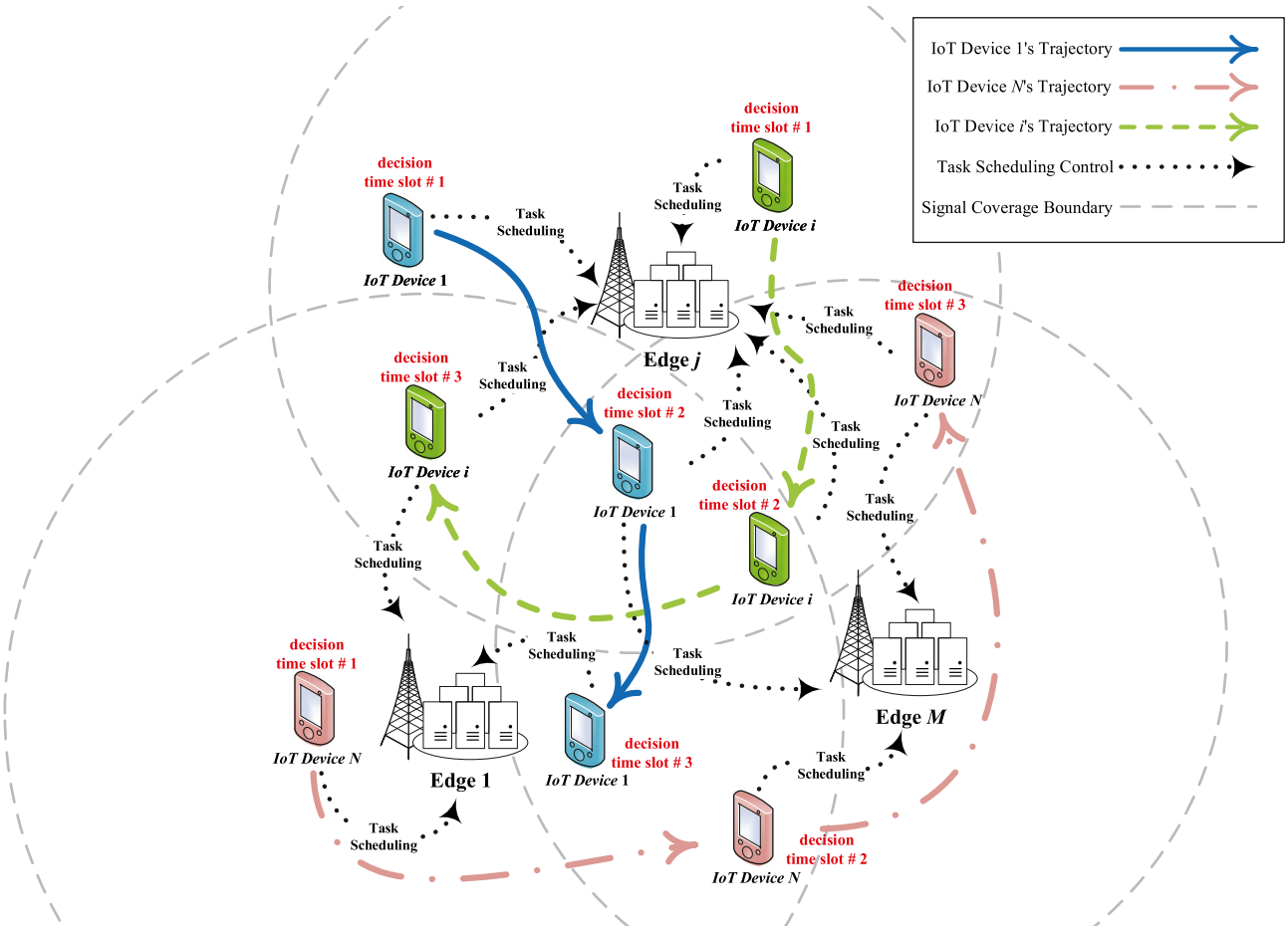

Fig. 1 Overview of Mobile Edge Computing (MEC) system

of resource instances at time slot t may have been allocated to previous job requests when conducting task scheduling to incoming job requests, the available number of resource instances at time slot $R_j(t)$ is no greater than the overall resource capacity A_j, as formulated by $R_j(t) \leq A_j$.

Signal coverage overlap across edge base stations is considered. Each edge base station j has a limited signal coverage with the maximum distance L_j. The IoT devices, whose locations are within the scope of maximum signal coverage L_j of edge base station j, form up the set of affiliated IoT devices for the edge base station j, labeled as $\mathcal{N}_j$. The affiliated IoT devices can propose their job requests and upload the IoT data to the specified base station for further edge processing. From the standpoint of IoT devices, the set of accessible edge servers/base stations for the job request of IoT device i is indicated by $\mathcal{M}_i$.

Moreover, the CPU cycle frequency of each resource instance is denoted by $f_{\mathcal{M}}$. Note that all of M edger servers are regarded as homogenous with the same CPU cycle frequency per resource instance.

Mobile IoT device The sensor hub has become one of typical representatives for IoT devices [8, 13]. It collects multiple pieces of data from different sensors and buffers them for further batch processing [7]. Higher-valued information can be concluded through integrating data from various kinds of sensors [19]. In our work, we formulate the mobile IoT device as a sensor hub which gathers multiple pieces of sensing data and packages them as a job request to the edge site for batch processing. Multiple pieces of sensing data are regarded as several tasks within a single job request.

At each decision time slot Γ, job requests from N mobile IoT devices are collected and scheduled for batch processing at edge site. The set of N mobile IoT devices is denoted by $\mathcal{N}$, and each edge server is indexed by i. Each mobile IoT device may change its position from time to time, and thus may have different accessible edge servers/base stations $\mathcal{M}_i$ at various times. Edge base stations' signal coverage and IoT device's location jointly determine the set of accessible edge servers/base stations for an IoT device. Because of the mobility of IoT devices, job requests proposed at various times may be scheduled to different edge servers.

Each IoT device i will report a budget B_i representing the maximum payment for edge computation when proposing a job request to the edge site. A number of tasks are processed in batch, which are bundled into a single job request for execution. With more tasks executed at edge site, the edge provider will gain more revenue from the budget B_i. The IoT device also submits a maximum expected job completion time $\tau_i^{\max}$. It is assumed that, only the tasks executed within the time of $\tau_i^{\max}$ are paid. Thus, the unaccomplished tasks within $\tau_i^{\max}$ will be rejected by edge providers, requiring to be re-submitted by the IoT device.

The job request proposed at each decision time slot Γ should be firstly processed at the mobile IoT device, including data preprocessing and packing. The IoT device i has the CPU frequency of f_i, and μ_i^{lc} CPU cycles are needed for local execution at the mobile IoT device. Therefore, the local computational latency τ_i^{lc} can be calculated by Eq. 1. Regarding to the edge execution, μ_i^{rc} CPU cycles are estimated for data processing at edge site, an thus the remote computational latency at edge site is formulated in Eq. 2. For the convenience of timeslot-based task scheduling introduced later, the computational latency at device and edge site is measured in time slot τ.

$$\tau_i^{lc} = \left\lceil \frac{\mu_i^{lc}}{f_i \cdot \tau} \right\rceil \tag{1}$$

$$\tau_i^{rc} = \left\lceil \frac{\mu_i^{rc}}{f_{\mathcal{M}} \cdot \tau} \right\rceil \tag{2}$$

Furthermore, data transmission between the mobile IoT device and edge servers is required during the lifecycle of a job request, resulting in a data transmission latency. Without loss of generality, we assume that there is the data size of δ_i^{rc} to be processed at edge site. Then, according to the Shannon-Hartley formula, data transmission latency between the mobile IoT device i and edge server j can be calculated as Eq. 3, where w_i represents the channel bandwidth allocated to the mobile IoT device i, while h_i denotes the channel power gain of device i and p_i is the wireless transmit power of the mobile IoT device i. Channel background noise is indicated by σ. The channel power gain h_i is negatively correlated to the data transmission distance $d_{i,j}^{tr}$, estimated by $d_{i,j}^{tr\,-4}$ [33]. Note that, data transmission latency is also estimated in time slot τ.

$$\tau_{i,j}^{tr} = \left\lceil \frac{\delta_i^{rc}}{w_i \cdot \tau \cdot \log_2(1 + h_i p_i/\sigma)} \right\rceil \tag{3}$$

Table 1 summarizes the notations used in the problem of task scheduling and resource management in MEC system.

3.2 Problem formulation

We formulate the problem of task scheduling and resource management in MEC system. Each job request can be cross-edge scheduled to multiple edge servers, according to our well-defined scheduling algorithm. Let $\alpha_{i,j}(t)$ denote the number of edge server j's resource instances allocated to the job request of mobile IoT device i at the time slot t. Task scheduling is conducted per decision time slot Γ. The granularity of decision time slot Γ is configured by system

Table 1 Summary of notations

Notation	Definition
t, τ	index, length of time slot
Γ	index of decision time slot
$i, N, \mathcal{N}$	index, number, set of mobile IoT devices
$j, M, \mathcal{M}$	index, number, set of edge servers/base stations
B_i	budget proposed by IoT device i's job request
L_j	maximum distance of signal coverage for edge server/base stations j
$\mathcal{M}_i$	set of accessible edge servers/base stations for IoT device i
$\mathcal{N}_j$	set of affiliated IoT devices within signal coverage for edge server/base stations j
μ_i^{lc}, μ_i^{rc}	number of CPU cycles to IoT device i's job request for local computation, remote computation at edge site
$f_i, f_{\mathcal{M}}$	CPU frequency of IoT device i, edge server
τ_i^{lc}, τ_i^{rc}	local computational latency, remote computational latency at edge site for IoT device i's job request
δ_i^{rc}	data size to IoT device i's job request for remote computation at edge site
w_i, h_i, p_i	channel bandwidth, channel power gain, wireless transmit power of IoT device i
σ	noise power at edge site
$d_{i,j}^{tr}, \tau_{i,j}^{tr}$	data transmission distance, latency between IoT device i and edge server j
$\tau_i^{\max}$	maximum expected job completion time for IoT device i's job request
$\mathcal{T}_{i,j}$	set of available time slots for remote execution at edge server j for IoT device i's job request
$A_j, R_j(t)$	number of edge server j's overall resource instances, available resource instances at time slot t
$k, \mathcal{R}_j(t)$	index, set of possible resource allocations for edge server j at time slot t
$\alpha_{i,j}(t)$	number of edger server j's resource instances allocated to IoT device i's job request at the time slot t
$\lambda_{i,j}^k(t)$	real decision variables generated by λ-technique [26] in LP transformation

operator according to the actual situation, including one or more units of time slot t.

In this work, task scheduling is conducted with the objective of optimizing the revenue earned by edge providers, which is specifically maximizing the summation of each edge provider $j \in \mathcal{M}$'s revenue. The overall revenue is earned from the submitted budgets B_i of all N mobile IoT devices. As described in Section 3.1, it is proportional to the task completion ratio within the maximum expected job completion time $\tau_i^{\max}$, formulated as Eq. 4.

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}_i}\sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}}\frac{B_i}{\tau_i^{rc}}\cdot\alpha_{i,j}(t) \tag{4}$$

Given the limited resource capacity of edger servers at each time slot t, the total amount of allocated resources of each edge server $j \in \mathcal{M}$ cannot excess its resource capacity, formulated as the following constraint (5) regarding the resource capacity of edge servers.

$$\sum_{i\in\mathcal{N}_j}\alpha_{i,j}(t) \le R_j(t) \qquad \forall j \in \mathcal{M}, \quad \forall t \in \mathcal{T}_{i,j} \tag{5}$$

Besides, it is obvious that the task completion ratio within the maximum expected job completion time $\tau_i^{\max}$ should be no greater than 1. The constraint about task completion ratio is formulated as Eq. 6.

$$\sum_{j\in\mathcal{M}_i}\sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}}\frac{\alpha_{i,j}(t)}{\tau_i^{rc}} \le 1 \qquad \forall i \in \mathcal{N} \tag{6}$$

Through combining the optimization objective with the above-mentioned constraints, our problem of task scheduling and resource management in MEC system can be formulated formally and mathematically as follows.

$$\max_{\alpha_{i,j}(t)}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}_i}\sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}}\frac{B_i}{\tau_i^{rc}}\cdot\alpha_{i,j}(t) \tag{7}$$

subject to,

$$\sum_{i\in\mathcal{N}_j}\alpha_{i,j}(t) \le R_j(t) \qquad \forall j \in \mathcal{M}, \quad \forall t \in \mathcal{T}_{i,j} \tag{8}$$

$$\sum_{j\in\mathcal{M}_i}\sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}}\alpha_{i,j}(t) \le \tau_i^{rc} \qquad \forall i \in \mathcal{N} \tag{9}$$

$$\alpha_{i,j}(t) \in \mathcal{R}_j(t) \quad \forall i \in \mathcal{N}, \quad \forall j \in \mathcal{M}_i, \quad \forall t \in \mathcal{T}_{i,j} \tag{10}$$

where $\mathcal{R}_j(t)$ denotes the set of all possible resource allocations $\{0, 1, \ldots, R_j(t)\}$, and $\mathcal{T}_{i,j}$ indicates the set of available time slots for remote execution refering to $\{\tau_i^{lc} + \tau_{i,j}^{tr}, \tau_i^{lc} + \tau_{i,j}^{tr} + 1, ..., \tau_i^{\max}\}$.

The optimization problem defined above is an integer programming (IP) problem, which is generally NP-hard [16] to solve out. In order to schedule job requests in an efficient manner, we try to explore and make use of favorable techniques overcoming the challenges on problem solving, which will be introduced in detail in Section 4.

4 Algorithm design

In this section, we firstly outline the task scheduling and resource management framework in MEC system, and then concentrate on the equivalent LP transformation aimed to solve our problem efficiently. Theoretical analyses on the effectiveness and computational complexity are given as well.

4.1 Task scheduling framework overview

Algorithm 1 illustrates the pseudo code of cross-edge task scheduling algorithm achieving optimal earning. When the job request proposed by mobile IoT device is scheduled, the number of allocated resource instances is determined as well. The procedure of task scheduling is conducted round by round with decision time slot Γ.

At each decision time slot Γ, task scheduler firstly receives the job requests submitted by N mobile IoT devices during the last decision time slot, with location of mobile IoT device and other device information (e.g., B_i, $\tau_i^{\max}$, $etc.$) collected as well. According to the current location of mobile IoT device and signal coverage of edge base stations, the set of its accessible edge servers/base stations (i.e., $\mathcal{M}_i$) is obtained. Then, it is high time to perform LP solving aimed to acquire the results of task scheduling and resource allocation $\boldsymbol{\alpha}$. How to apply LP techniques to solve the task scheduling problem (7) will be fully discussed in Sections 4.2 and 4.3.

When scheduling the job requests according to $\boldsymbol{\alpha}$, edge providers will earn the maximum revenue from budgets. Since edge servers' resource capacity is limited, part of tasks within the job request may be rejected and unscheduled. The reason for task rejection is a less budget offering B_i with a strict requirement for maximum expected job completion time $\tau_i^{\max}$. The mobile IoT device having rejected task is suggested to re-submit its unscheduled tasks with more budget added or a relaxed requirement for maximum expected job completion time $\tau_i^{\max}$.

With the resources allocated to job requests, each edge server's resource capacity at different time slots should be updated. If the number of resource instances $\alpha_{i,j}(t)$ of edge server j is allocated to the job request i at time slot t, then the edge server j's resource capacity at time slot t will be reduced by $\alpha_{i,j}(t)$ number of resource instances, in preparation for the next round of task scheduling at the decision time slot of $\Gamma = \Gamma + 1$.

4.2 Integral optimum guarantee

A linear programming problem can yield an optimal solution in integers, if its coefficient matrix is total unimodular (TU) [16]. In our problem setting, we investigate the coefficient matrix of linear constraints (8) and (9), testifying whether to satisfy the TU property. The coefficient matrix is denoted by an r-by-c matrix $\mathbf{C}_{r\times c}$.

The matrix $\mathbf{C}_{r\times c}$ is said to be total unimodular, if it meets the following two conditions: (1) Any of it entries $a_{x,y}$ in $\mathbf{C}_{r\times c}$ belongs to $\{-1, 0, 1\}$; (2) There exist two disjoint row subsets R_1 and R_2 exactly partitioning the row set $\{1, 2, ..., r\}$, such that $\left|\sum_{x\in R_1} a_{x,y} - \sum_{x\in R_2} a_{x,y}\right| \leq 1$, for $\forall y \in \{1, 2, ..., c\}$.

Algorithm 1 Cross-edge task scheduling algorithm achieving optimal revenue.

Input: locations of all edge servers;
Output: results of task scheduling and resource allocation;

Initialize $R_j(t) \leftarrow A_j$, for $\forall j \in \mathcal{M}, \forall t \in \mathcal{T}$;
for each decision time slot $\Gamma \in \{1, \ldots, \Gamma^{\max}\}$ **do**
  Receive and collect the batch job requests from each mobile IoT device $i \in \mathcal{N}$;
  Initialize $\mathcal{M}_i \leftarrow \emptyset$, for $\forall i \in \mathcal{N}$;
  for each pair $(i, j) \in \mathcal{N} \times \mathcal{M}$ **do**
    Calculate the data transmission distance $d_{i,j}^{tr}$ between mobile IoT device i and edge server j;
    if data transmission distance $d_{i,j} \leq L_j$ **then**
      $\mathcal{M}_i \leftarrow \mathcal{M}_i \bigcup \{j\}$;
      $\tau_{i,j}^{tr} \leftarrow \left\lceil \frac{\delta_i^{rc}}{\lfloor w_i \cdot \tau \cdot \log_2(1+h_i p_i/\sigma) \rfloor} \right\rceil$;
    end if
  end for
  Solve the LP problem (20) to obtain the results of task scheduling and resource allocation $\boldsymbol{\alpha}$;
  Schedule the proposed job requests with edge resources allocated, according to $\boldsymbol{\alpha}$;
  Signal the unscheduled tasks to the corresponding IoT device;
  for each edge server $j \in \mathcal{M}$ **do**
    for each time slot $t \in \bigcup_{i\in\mathcal{N}_j} \mathcal{T}_{i,j}$ **do**
      Update $\mathcal{R}_j(t) \leftarrow R_j(t) - \sum_{i\in\mathcal{N}_j} \alpha_{i,j}(t)$;
    end for
  end for
end for

Lemma 1 *The coefficient matrix formed by the constraints of* Eqs. 8 *and* 9 *satisfies the total unimodular (TU) property.*

Proof : Suppose that the matrix $\mathbf{C}_{r\times c}$ represents the coefficient matrix formed by the constraints (8) and (9), where the numbers of rows and columns are respectively denoted by r and c, specifically formulated by Eqs. 11 and 12. The number of rows represents the total number of constraints, where r_1 and r_2 separately indicate the number of constraints (8) and (9). The number of columns shows the dimension of the variable $\boldsymbol{\alpha}$.

$$\begin{aligned} r &= r_1 + r_2 \\ &= \left[\sum_{j\in\mathcal{M}} \sum_{i\in\mathcal{N}_j} \left(\tau_i^{\max} - \tau_i^{lc} - \tau_{i,j}^{tr} + 1 \right) \right] + N \end{aligned} \tag{11}$$

$$c = \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{M}_i} \left(\tau_i^{\max} - \tau_i^{lc} - \tau_{i,j}^{tr} + 1 \right) \tag{12}$$

Firstly, any entry of coefficient matrix $\mathbf{C}_{r\times c}$ belongs to $\{0, 1\}$, satisfying the *Condition* 1 where any matrix entry is either -1, 0, or 1.

Secondly, in terms of the *Condition* 2, the entries of rows which belong to the row subset $\{1, 2, ..., r_1\}$ are elected as the set R_1. As well, the remaining entries of rows form up another set R_2, satisfying $R_1 \bigcap R_2 = \emptyset$. Each entry in the coefficient matrix $\mathbf{C}_{r\times c}$ is denoted by $a_{x,y}$, where x and y are the index of matrix's row and column. In view of the constraints (8), it is pointed out that the summation of entries grouped by columns in rows R_1 is a $1 \times c$ vector whose entries are 1, formulated by Eq. 13. With respect to the constraints (9), as such, the summation of entries grouped by columns in rows R_1 is also a $1 \times c$ vector with all entries equal to 1, expressed as (14). Through integrating (13) and (14) into account, we conclude that the coefficient matrix $\mathbf{C}_{r\times c}$ satisfies the *Condition* 2 specified as Eq. 15.

$$\sum_{x\in R_1} a_{x,y} = 1, \ \forall y \in \{1, 2, ..., c\} \tag{13}$$

$$\sum_{x\in R_2} a_{x,y} = 1, \ \forall y \in \{1, 2, ..., c\} \tag{14}$$

$$\left| \sum_{x\in R_1} a_{x,y} - \sum_{x\in R_2} a_{x,y} \right| = 0 \leq 1, \ \forall y \in \{1, 2, ..., c\} \tag{15}$$

In brief, it is summarized that the coefficient matrix $\mathbf{C}_{r\times c}$ both satisfies two conditions for total unimodularity, proving that the coefficient matrix formed by the constraints of Eqs. 8 and 9 satisfies the total unimodular (TU) property. □

Lemma 1 provides the legitimacy of linear relaxation on the integer constraints (10). Our task scheduling problem (7) has integral optimum as long as any optimum exists. The integer constraints (10)'s relaxation is formulated as Eq. 16.

$$\alpha_{i,j}(t) \in \left[0, R_j(t)\right] \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{M}_i, \forall t \in \mathcal{T}_{i,j} \tag{16}$$

4.3 Equivalent LP transformation

With the problem structure of TU constraints exploited, we further take advantage of λ-technique to transform the IP problem (7) to an equivalent linear programming (LP) problem which has the same optimal solution [26]. Thanks to the success of equivalent LP transformation, our scheduling problem (7) can be solved efficiently in polynomial time with the problem scale.

Recall that a decision variable $\alpha_{i,j}(t) \in \mathcal{R}_j(t) = \{0, 1, ..., R_j(t)\}$, then the objective function (4) can be linearized with the λ-representation as follows.

$$\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{M}_i} \sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}} \sum_{k=0}^{R_j(t)} \frac{B_i}{\tau_i^{rc}} \cdot k \cdot \lambda_{i,j}^k(t) \tag{17}$$

In specific, each of integer variable $\alpha_{i,j}(t)$ in our IP problem (7) is removed by sampling at each of its possible value $k \in \mathcal{R}_j(t)$, weighted by the newly introduced variables $\lambda_{i,j}^k(t) \in \mathbb{R}^+$, satisfying the following equations.

$$\alpha_{i,j}(t) = \sum_{k=0}^{R_j(t)} k \cdot \lambda_{i,j}^k(t), \quad \sum_{k=0}^{R_j(t)} \lambda_{i,j}^k(t) = 1 \tag{18}$$

$$\lambda_{i,j}^k(t) \in \mathbb{R}^+, \ \forall k \in \mathcal{R}_j(t) \tag{19}$$

Combining with linear relaxation on integer constraints (10), the equivalent LP problem is obtained as follows.

$$\max_{\alpha_{i,j}(t),\ \lambda_{i,j}^k(t)} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{M}_i} \sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}} \sum_{k=0}^{R_j(t)} \frac{B_i}{p_i} \cdot k \cdot \lambda_{i,j}^k(t) \tag{20}$$

subject to,

$$\sum_{i\in\mathcal{N}_j} \sum_{k=0}^{R_j(t)} k \cdot \lambda_{i,j}^k(t) \leq R_j(t) \quad \forall j \in \mathcal{M}, \quad \forall t \in \mathcal{T}_{i,j} \tag{21}$$

$$\sum_{j\in\mathcal{M}_i} \sum_{t=\tau_i^{lc}+\tau_{i,j}^{tr}}^{\tau_i^{\max}} \sum_{k=0}^{R_j(t)} k \cdot \lambda_{i,j}^k(t) \leq \tau_i^{rc} \quad \forall i \in \mathcal{N} \tag{22}$$

$$\sum_{k=0}^{R_j(t)} \lambda_{i,j}^k(t) = 1 \quad \forall i \in \mathcal{N}, \quad \forall j \in \mathcal{M}_i, \quad \forall t \in \mathcal{T}_{i,j} \tag{23}$$

$$\alpha_{i,j}(t), \ \lambda_{i,j}^k(t) \in \mathbb{R}^+ \quad \forall i \in \mathcal{N}, \quad \forall j \in \mathcal{M}_i, \quad \forall k \in \mathcal{R}_j(t), \quad \forall t \in \mathcal{T}_{i,j} \tag{24}$$

Theorem 1 *An optimal solution to problem (7) is an optimal solution to problem (20), which can be solved at polynomial computational complexity in the number of mobile IoT devices and edge servers.*

Proof : According to Lemma 1, the coefficient matrix with the TU property ensures the integral optimum of the LP problem (20), which is also the optimal solution to the IP problem (7).

Reference to the literature [2], an LP problem can be solved in time that is polynomial with the problem's input size (i.e., number of variables and constraints) using existing methodologies like the interior-point, simplex, and ellipsoid algorithms. Note that, the number of λ-varibles in the LP problem (20) is $O(NMT^*R^*)$, where T^* and R^* are formulated as Eq. 25. And the number of constraints of Eqs. 21–23 is $O(NMT^*)$.

$$T^* = \left| \bigcup_{i \in \mathcal{N}, j \in \mathcal{M}_i} \mathcal{T}_{i,j} \right|, \qquad R^* = \max_{j \in \mathcal{M}} A_j \tag{25}$$

In summary, the LP problem (20) can be solved at polynomial computational complexity of $O(NMT^*R^*)$. Thanks the optimum equivalence between problem (7) and (20) proved earlier, the optimal solution of problem (7) which is formulated by IP can be worked out at polynomial computational complexity in the number of mobile IoT devices (i.e. N) and edge servers (i.e., M), through LP solving. □

By Theorem 1, our task scheduling problem (7) can be precisely solved with a polynomial computational complexity in the scale of MEC system, including the number of mobile IoT devices and edge servers. Henceforth, the revenue-optimal task scheduling results in MEC system can be solved with efficient LP algorothms (e.g., interior-point, simplex, and ellipsoid algorithms, etc.) and solvers (e.g., IBM CPLEX [1], YALMIP [21]).

5 Evaluation

5.1 Experimental setup

In order to evaluate the performance of our proposed cross-edge task scheduling algorithm, we conduct experiments whose scenario is simulated based on the geolocation information of base stations and users within the Melbourne CBD area in the EUA dataset [17]. Twenty of base stations are randomly selected from the EUA dataset and applied as edge server/base stations in our experiments. The geolocation information of users in the EUA dataset is used as the initial location of mobile IoT devices, and the mobility of IoT devices is implemented according to a Gaussian random walk model.

In Table 2, the values of main system parameters are presented, where the parameters of L_i, B_i, $\tau_i^{\max}$ are randomly generated within the with respect to each mobile IoT device. Parameters marked with ⋆ are configured according to [25], while other parameters are set after our in-depth investigation. Experiments are carried out in MATLAB with the aid of the YALMIP optimization toolbox [21] to invoke IBM CPLEX solver [1] used for LP problem solving.

Table 2 Experimental parameter setting

Param.	Value	Param.	Value
τ	2 ms ⋆	Γ	50 ms
w_i	1.5 Mbps	p_i	1 W ⋆
σ	10^{-13} W ⋆	δ_i^{rc}	2,000 bits
L_i	550 - 750 m	B_i	10 - 15 pence
A_j	5	$\tau_i^{\max}$	140 - 155 ms

5.2 Experimental results

With the purpose of verifying the effectiveness of our proposed approach, we compare our cross-layer task scheduling algorithm with an intuitive baseline algorithm which is *Greedy Scheduling*. In the greedy scheduling algorithm, job requests from each mobile IoT device will be scheduled to the nearest edge server; meanwhile, cross-edge scheduling should not be permitted.

First, we simulate the procedure of task scheduling by decision time slots under the above two algorithms. At each decision time slot, fifty mobile IoT devices propose job requests according to the Bernoulli distribution with the probabilistic parameter randomly determined between (0.4, 0.5). Besides, each mobile IoT device moves based on the two-dimensional Gaussian random walk of $(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$, where μ_x and μ_y are randomly generated between (-0.5 m, 0.5 m) while σ_x and σ_y equal to 1 m.

Figure 2 shows the overall revenue of the edge providers across various decision time slots under different algorithms. It can be easily seen that the overall revenue earned under our cross-edge task scheduling algorithm is always higher than that under greedy algorithm. This is because the cross-edge scheduling mechanism makes the edge resources more sufficiently utilized. As regards the greedy algorithm, nevertheless, job requests are merely scheduled to the single nearest edge server, resulting in imbalance on resource utilization amongst the edge servers. Edge servers surrounded by a large number of devices will receive active requests for resource allocation leading to request queueing and resource overutilization, while edge servers having less devices surrounded will rarely receive requests for resource allocation, incurring more idle resources. More sufficient utilization of edge resources in our proposed algorithm implies more tasks within the job request are accomplished within the decision time

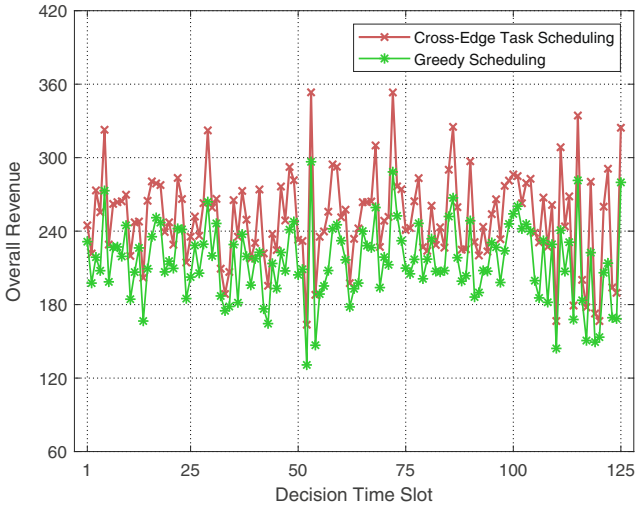

Fig. 2 Edge providers' overall revenue under different algorithms

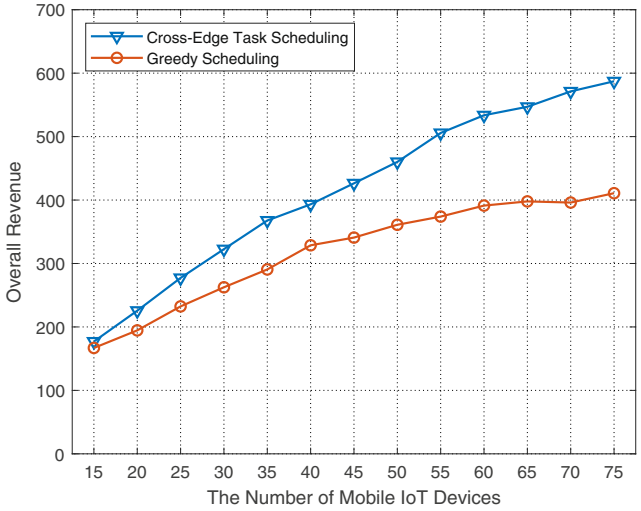

Fig. 4 Edge providers' overall revenue with various mobile IoT device scales

slot, as illustrated in Fig. 3. In summary, our cross-edge task scheduling algorithm develops the utilization of edge resources, and thereby earns a higher revenue.

Second, we also tune the number of mobile IoT devices which propose job requests to evaluate the revenue earning under two different algorithms, as depicted in Fig. 4. Although edge providers can earn more revenue with the increasing number of mobile IoT devices, different scheduling algorithm makes a difference on the specific gain of revenue. Edge providers under our cross-edge task scheduling algorithm always earn higher than that under greedy algorithm This is because, our cross-edge task scheduling algorithm has a better tolerance on device expansion. Job requests can be handled across multiple edge servers in a distributed manner when our cross-edge algorithm is applied. In comparison, job requests can be only dispatched to the single nearest edge server in the greedy algorithm. Thanks to the cross-edge scheduling, tasks within the job request are more efficiently processed, thus relieving the request congestion due to device expansion and handling more tasks with more revenue earned.

Third, computational complexity of our algorithm is testified as well, by means of measuring the runtime of our cross-edge task scheduling algorithm. As demonstrated in Fig. 5, algorithmic runtime is measured under different system scales (i.e., number of mobile IoT devices or edge servers). The algorithmic runtime result at each particular system scale is averaged over 1,000 runs. Our algorithm is performed on the machined whose processor is Intel Core i5-3470 @3.20Ghz with the memory size of 8 GB. Under the growth of system scale (i.e., number of mobile

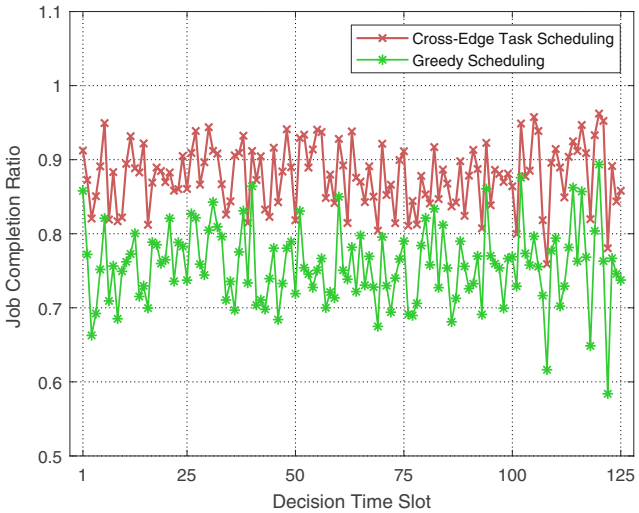

Fig. 3 Mobile IoT devices' job completion ratio under different algorithms

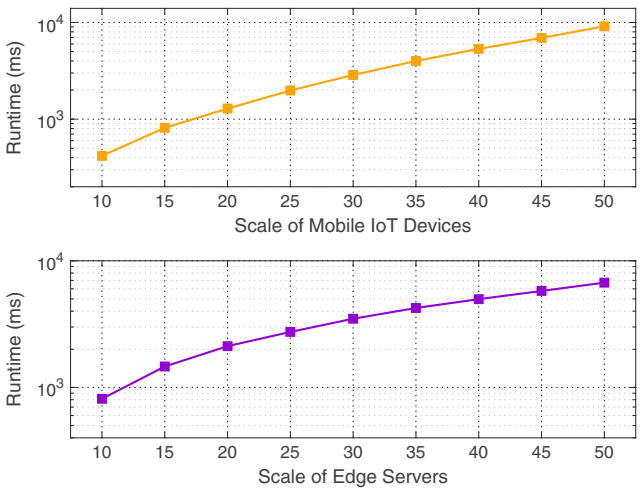

Fig. 5 Algorithmic runtime at different system scales (i.e., number of mobile IoT devices or edge servers)

IoT devices or edge servers), our algorithmic runtime is always kept as a polynomial increase. It further verifies the theoretical analysis of computational complexity in Theorem 1 that our algorithm can be solved at polynomial computational complexity in the number of mobile IoT devices and edge servers.

6 Conclusion and future work

In this paper, we study the problem of task scheduling and resource management for IoT batch jobs in MEC system, where the optimal edge providers' revenue is achieved. Based on the system formulation, our task scheduling problem maximizing revenue is intuitively formulated by an IP problem. By virtue of the NP-hardness solving IP problem, we find out the favorable property of TU coefficient matrix, and further introduce the λ-technique to transform the original IP problem to an equivalent LP problem. Thanks to the polynomial computational complexity for LP problem solving, our task scheduling problem can be worked out in an efficient manner. Experiments based on real-life IoT dataset are conducted to verify our proposed approach.

There are several avenues for future work. On the one hand, we plan to design a more sophisticated task scheduling algorithm considering the future job dynamics, rather than merely maximizing revenue of job requests proposed at the current decision time slot. On the other hand, we can also comprehend the problem of multi-device job scheduling in edge computing from the perspective of game theory, and try to find out several game-theoretical properties (e.g., fairness, individual rationality, envy freeness) which helps to the design of MEC systems. Some other interesting research topics on edge computing can be explored if our proposed methodology is implemented in real-life IoT systems.

Acknowledgements This work is supported by the National Natural Science Foundation of China (No. 61972414), Beijing Natural Science Foundation (No.4202066) and the Fundamental Research Funds for the Central Universities (No. 2462018YJRC040).

References

1. CPLEX optimizer: High-performance mathematical programming solver. https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/
2. Bertsimas D, Tsitsiklis JN (1997) Introduction to linear optimization, vol 21. Athena Scientific Belmont, MA
3. Chen C, Wang W, Li B (2018) Performance-aware fair scheduling: Exploiting demand elasticity of data analytics jobs. In: IEEE INFOCOM 2018 - IEEE conference on computer communications, pp 504–512
4. Chen K, Shen H, Sapra K, Liu G (2015) A social network based reputation system for cooperative P2P file sharing. IEEE Transactions on Parallel and Distributed Systems 26(8):2140–2153
5. Chen Y, Zhang N, Zhang Y, Chen X, Wu W, Shen XS (2019) Energy efficient dynamic offloading in mobile edge computing for Internet of Things. IEEE Transactions on Cloud Computing. https://doi.org/10.1109/TCC.2019.2898657
6. Chen Y, Zhang N, Zhang Y, Chen X, Wu W, Shen XS (2019) TOFFEE: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing. IEEE Transactions on Cloud Computing. https://doi.org/10.1109/TCC.2019.2923692
7. Chien T, Chiou L, Sheu S, Lin J, Lee C, Ku T, Tsai M, Wu C (2016) Low-power MCU with embedded ReRAM buffers as sensor hub for IoT applications. IEEE Journal on Emerging and Selected Topics in Circuits and Systems 6(2):247–257
8. Cicioglu M, Calhan A (2019) Dynamic HUB selection process based on specific absorption rate for WBANs. IEEE Sens J 19(14):5718–5722
9. Guo L, Chen Z, Zhang D, Liu J, Pan J (2019) Sustainability in body sensor networks with transmission scheduling and energy harvesting. IEEE Internet of Things Journal 6(6):9633–9644
10. Hu L, Liu A, Xie M, Wang T (2019) UAVs joint vehicles as data mules for fast codes disseminationfor edge networking in smart city. Peer-to-Peer Networking and Applications 12(6):1550–1574
11. Huang J, Lan Y, Xu M (2018) A simulation-based approach of QoS-aware service selection in mobile edge computing. Wireless communications and mobile computing, Article ID 5485461, pp 1–10
12. Huang J, Li S, Chen Y, Chen J (2018) Performance modelling and analysis for IoT services. International Journal of Web and Grid Services 14(2):146–169
13. Huang J, Zhang C, Zhang J (2020) A multi-queue approach of energy efficient task scheduling for sensor hubs. Chinese Journal of Electronics. https://doi.org/10.1049/cje.2020.02.001
14. Kim DY, Lee A, Kim S (2019) P2P computing for trusted networking of personalized IoT services. Peer-to-Peer Networking and Applications. https://doi.org/10.1007/s12083-019-00737-z
15. Kim S, Kim DY (2017) Efficient data-forwarding method in delay-tolerant P2P networking for IoT services. Peer-to-Peer Networking and Applications 11(6):1176–1185
16. Korte BBH, Vygen J (2007) Algorithms and combinatorics, combinatorial optimization: theory and algorithms, vol 21, 4th edn., chap 5. Springer, pp 110–115
17. Lai P, He Q, Abdelrazek M, Chen F, Hosking J, Grundy J, Yang Y (2018) Optimal edge user allocation in edge computing with variable sized vector bin packing. In: 16th International conference on service-oriented computing (ICSOC), pp 230–245
18. Lee Y, Lee KM, Lee SH (2019) Blockchain-based reputation management for custommanufacturing service in the peer-to-peer networking environment. Peer-to-Peer Networking and Applications. https://doi.org/10.1007/s12083-019-00730-6
19. Lengyel L, Ekler P, Ujj T, Balogh T, Charaf H (2015) SensorHUB : An IoT driver framework for supporting sensor networks and data analysis. International Journal of Distributed Sensor Networks, Article ID 454379, pp 1–12
20. Li S, Huang J (2017) Energy efficient resource management and task scheduling for IoT services in edge computing paradigm. In: IEEE International symposium on parallel and distributed processing with applications (ISPA), pp 846–851
21. Lofberg J (2004) YALMIP : a toolbox for modeling and optimization in MATLAB. In: IEEE International symposium on computer aided control systems design (CACSD), pp 284–289
22. Lu T, Shan C, Wei L (2017) Fog computing enabling geographic routing for urban area vehicular network. Peer-to-Peer Networking and Applications 11(4):749–755

23. Ma W, Liu X, Mashayekhy L (2019) A strategic game for task offloading among capacitated UAV-mounted cloudlets. In: IEEE International Congress on Internet of Things (ICIOT), pp 61–68
24. Mao H, Schwarzkopf M, Venkatakrishnan S, Meng Z, Alizadeh M (2019) Learning scheduling algorithms for data processing clusters. In: Proceedings of the 2019 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), pp 270–288
25. Mao Y, Zhang J, Letaief KB (2016) Dynamic computation offloading for mobile-edge computing with energy harvesting devices. IEEE Journal on Selected Areas in Communications 34(12):3590–3605
26. Meyer RR (1977) A class of nonlinear integer programs solvable by a single linear program. SIAM Journal on Control & Optimization 15(6):935–946
27. Ramachandran U, Gupta H, Hall A, Saurez E, Xu Z (2019) Elevating the edge to be a peer of the cloud. In: IEEE 12th International conference on cloud computing (CLOUD), pp 17–24
28. Satsiou A, Tassiulas L (2010) Reputation-based resource allocation in P2P systems of rational users. IEEE Transactions on Parallel and Distributed Systems 21(4):466–479
29. Ali Shah PAA, Habib M, Sajjad T, Umar M, Babar M (2017) Applications and challenges faced by internet of things - A Survey. In: Ferreira J., Alam M. (eds) Future Intelligent Vehicular Technologies. Future 5V 2016. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 185. Springer, Cham
30. Teng H, Liu Y, Liu A, Xiong NN, Cai Z, Wang T (2019) A novel code data dissemination scheme for Internet of Things through mobile vehicle of smart cities. Future Generation Computer Systems 94:351–367
31. Wang J, Hu J, Min G, Zhan W, Ni Q, Georgalas N (2019) Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning. IEEE Communications Magazine 57(5):64–69
32. Wang T, Wei X, Tang C, Fan J (2017) Efficient multi-tasks scheduling algorithm in mobile cloud computing with time constraints. Peer-to-Peer Networking and Applications 11(4):793–807
33. Xu C, Lei J, Li W, Fu X (2016) Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Transactions on Networking 24(5):2795–2808
34. Xu J, Ota K, Dong M (2019) Energy efficient hybrid edge caching scheme for tactile Internet in 5G. IEEE Transactions on Green Communications and Networking 3(2):483–493
35. Yang L, Yang D, Cao J, Sahni Y, Xu X (2019) QoS guaranteed resource allocation for live VM migration in edge clouds. In: IEEE International Conference on Edge Computing (EDGE), pp 56–63
36. Zhang D, Qiao Y, She L, Shen R, Ren J, Zhang Y (2019) Two time-scale resource management for green Internet of Things networks. IEEE Internet of Things Journal 6(1):545–556
37. Zhang D, Tan L, Ren J, Awad MK, Zhang S, Zhang Y, Wan PJ (2019) Near-optimal and truthful online auction for computation offloading in green edge-computing systems. IEEE Transactions on Mobile Computing. https://doi.org/10.1109/TMC.2019.2901474
38. Zhao H, Deng S, Zhang C, Du W, He Q, Yin J (2019) A mobility-aware cross-edge computation offloading framework for partitionable applications. In: IEEE international conference on web services (ICWS), pp 193–200
39. Zwolenski M, Weatherill L (2014) The digital universe rich data and the increasing value of the Internet of Things. Australian Journal of Telecommunications & the Digital Economy 2(3):1–9

Jiwei Huang received the B.Eng. and Ph.D. degrees in computer science and technology from Tsinghua University, in 2009 and 2014, respectively. He was a Visiting Scholar with the Georgia Institute of Technology. He is currently an Associate Professor and the Dean of the Department of Computer Science and Technology, China University of Petroleum, Beijing, China. His research interests include services computing and performance evaluation. He is a member of the IEEE and ACM.

Songyuan Li is currently a master candidate in the State Key Laboratory of Networking and Switching Technology at Beijing University of Posts and Telecommunication. He received the B.Eng. degree from Beijing University of Posts and Telecommunications in 2018. His current research interests include services computing and performance optimization.

Ying Chen received the B.Eng. degree from Beijing University of Posts and Telecommunications in 2012, and the Ph.D. degree from Tsinghua University in 2017. She is an associate professor in the Computer School at Beijing Information Science and Technology University.