A Bilateral Game Approach for Task Outsourcing in Multi-access Edge Computing

Zheng Xiao*, *Member*, *IEEE*, Dan He, Yu Chen, Anthony Theodore Chronopoulos, *Senior member*, *IEEE*, Schahram Dustdar, *Fellow*, *IEEE*, and Jiayi Du, *Member*, *IEEE*

Abstract—Multi-access edge computing (MEC) is a promising architecture to provide low-latency applications for future Internet of Things (IoT)-based network systems. Together with the increasing scholarly attention on task offloading, the problem of edge servers' resource allocation has been widely studied. Most of previous works focus on a single edge server (ES) serving multiple terminal entities (TEs), which restricts their access to sufficient resources. In this paper, we consider a MEC resource transaction market with multiple ESs and multiple TEs, which are interdependent and mutually influence each other. However, this many-to-many interaction requires resolving several problems, including task allocation, TEs' selection on ESs and conflicting interests of both parties. Game theory can be used as an effective tool to realize the interests of two or more conflicting individuals in the trading market. Therefore, we propose a bilateral game framework among multiple ESs and multiple TEs by modeling the task outsourcing problem as two noncooperative games: the supplier and customer side games. In the first game, the supply function bidding mechanism is employed to model the ESs' profit maximization problem. The ESs submit their bids to the scheduler, where the computing service price is computed and sent to the TEs. While in the second game, TEs determine the optimal demand profiles according to ESs' bids to maximize their payoff. The existence and uniqueness of the Nash equilibrium in the aforementioned games are proved. A distributed task outsourcing algorithm (***DTOA***) is designed to determine the equilibrium. Simulation results have demonstrated the superior performance of ***DTOA*** in increasing the ESs' profit and TEs' payoff, as well as flattening the peak and off-peak load.

Index Terms—Multi-access edge computing (MEC), Internet of Things (IoT), Task outsourcing, Bidding mechanism, Noncooperative game, Nash equilibrium.

1 Introduction

The Internet of Things (IoT) is a system of interrelated tens of billions of resource-hungry terminal entities (TEs), such as, sensors, wearable devices and unmanned aerial vehicles, which transfer data over a network with little or no human intervention. With the development of TEs and wireless networks, the demand for low-latency computing services has been growing exponentially. This paves the way for the development of Multi-access edge computing (MEC).

Multi-access edge computing (MEC) enables a powerful cloud at the edge of the network. MEC decentralizes networks and allows any enterprise or mobile operator to place a cloud at the edge, adjacent to the user. In the MEC paradigm, plenty of machines are placed at the edge of the network so that computing services can be deployed on them for fast execution [1]. The MEC locates edge servers (ESs) with limited storage and computing resources at the edge of networks. Since the computation capabilities and battery lives of TEs are limited, TEs offload computationally intensive tasks (e.g., program execution) to ESs (e.g., 4G/5G base stations). The ESs execute these offloaded tasks and return results to TEs. However, to take full advantage of these available computational resources of ESs, TEs' tasks need to be allocated appropriately [2] .

Due to the resource constraints of ESs, some computationally intensive tasks will be offloaded to cloud servers (CSs), which are normally distantly located. In that case, a higher transmission latency may be generated, which seriously degrades the quality of service (QoS). Moreover, task offloading from ESs to CSs incurs extra latency and energy consumption due to communication between TEs and CSs. Some important problems to be solved are how to satisfy the latency requirements of TEs and reduce the energy consumption in MEC. In order to meet a strict QoS, Nafiseh *et al.* [3] proposed a two-sided matching mechanism for edge services considering QoS requirements in terms of service response time.

In order to achieve low-latency and energy-saving computations, several studies on task offloading in MEC have been proposed. Xinchen Lyu *et al.* [4] have attempted to encapsulate the latency requirements in offloading tasks and designed a selective offloading scheme. This scheme is achieved by enabling the devices to be self-denied or self-nominated for offloading. This can save energy consumption and minimize delays for task offloading. Authors in [5] developed a threshold-based strategy to improve the QoS, which combines the advantages of ESs' with lower latency and abundant computational resources of CSs. A priority queue is also applied to solve the delay problem, wherein delay-sensitive tasks are executed ahead of delay-tolerant tasks.

However, most previous studies examined a single ES

- *Corresponding Author
- *Zheng Xiao, Dan He, Yu Chen and Jiayi Du are with College of Computer Science and Electronic Engineering, Hunan University, Hunan, China, 410082. E-mail: {zxiao, danhe}@hnu.edu.cn, cy947205926@163.com, maxdujiayi@hnu.edu.cn,.*
- *Anthony Theodore Chronopoulos is with Department of Computer Science, University of Texas, San Antonio, TX, USA 78249 and Dept Computer Engineering Informatics 26500 Rio, University of Patras, Greece. Email: antony.tc@gmail.com.*
- *Schahram Dustdar is with Distributed Systems Group, Vienna University of Technology, Vienna, Austria. Email: dustdar@dsg.tuwien.ac.at.*

serving multiple TEs. In fact, the proliferation of applications puts a heavy load on the ESs. Since the ES has limited computing capacity and can't accommodate enough tasks, some task processes may have to wait longer to wake up. Thus, for a single ES is hard to cope with concurrent tasks of multiple TEs, which will hinder the development and popularity of MEC. In the future MEC market, there will be multiple different ESs offering optional computing service to TEs. Hence, TEs can choose different ESs according to their real-time and cost requirements. In that case, multiple ESs provide computing services in parallel, which can accelerate the speed of task processing and alleviate offloading delays.

Task outsourcing has been employed as an effective paradigm by accommodating as many on-demand tasks as possible. Its principle is to distribute TEs' tasks in different time slots according to ESs' load of each time slot. Our work focuses on the problem of offloaded tasks outsourcing. It differs from most existing works which focus on the necessity of offloading or on selecting which tasks to be offloaded to ESs or CSs. The TEs' offloaded tasks are mapped to ESs according to their resource capacities. Thus, task outsourcing can enhance scalability of MEC and satisfy TEs' dynamic service demands.

Fig. 1 describes a MEC communication network in IoT, where ESs are deployed densely near TEs. The MEC network is similar to a real competitive market, in which a wide range of TEs can be grouped into virtual clusters and compete for the ESs' limited wireless resources. They are interdependent and mutually influence each other. Several base stations with ESs also compete with each other to win more TEs. The ESs can communicate with the TEs via scheduler and inform them of their real-time service prices. Through the scheduler, TEs can participate in ESs selection, and make wise decisions regarding their daily computing resources consumption.

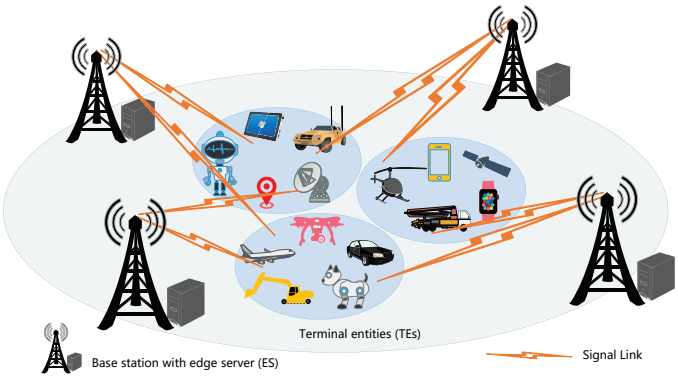

Fig. 1: The MEC network scenario in IoT.

However, one of the main challenges of task outsourcing is to consider the interests of both parties. On the one hand, ESs aim to get more profits, and strive for attracting more TEs to use their computing resources. On the other hand, a rational TE will choose a task strategy that maximizes its own payoff. Game theory can be used as an effective tool to model the interests of two or more conflicting individuals in trading markets and also load balancing in distributed systems [6]. The game solution was proved to be a Nash equilibrium solution for the noncooperative game.

In this paper, we propose a bilateral game framework among multiple ESs (the suppliers) and multiple TEs (the customers) to model the task outsourcing problem as two noncooperative games. These two games are related to each other and are played simultaneously. In the first game, the supply function bidding mechanism is employed to model the noncooperative game among ESs. In the proposed scheme, each ES, with limited or idle resources, submits a bid to reveal the available capacity "supplied" to the market. Then the scheduler collects these bids and computes a service price to clear the market so that the supply of the resource to be traded equals the demand. In particular, all TEs are charged the same service price at one time slot. The scheme can maximize the profits of ESs. In the second game, in order to reduce costs, TEs determine the amount of assigned tasks for each time slot based on the price of that time slot. If the price of one time slot is high, there would be fewer tasks assigned, and if the price is low, there would be more tasks. This framework can encourage TEs to assign fewer tasks during peak times or shift some tasks to off-peak times, which flattens the demand curve by peak clipping or valley filling.

In summary, the contributions of this paper are:

- A bilateral-game framework is developed to model the interactions among TEs and ESs.
- A supply function bidding mechanism is proposed, where each ES submits a bid to reveal the available capacity "supplied" to the market.
- A ***DTOA*** is designed to compute the Nash equilibrium.
- Simulations show that the proposed mechanism achieves the maximization of bilateral interests.

The remainder of this paper is organized as follows. In Section 2, the related work about task outsourcing in MEC is introduced. Section 3 models the task outsourcing problem in the ESs' and TEs' sides. In Section 4, a ***DTOA*** is designed to compute the Nash equilibrium in both sides. Section 5 presents simulations showing the performance of the new approach using ***DTOA***. Finally, conclusions are presented in Section 6.

2 Related Work

In recent years, significant attention has been devoted to the resource allocation in MEC networks [7]. Shi Yan *et al.* [8] studied the access selection for unmanned aerial vehicles (UAV) and bandwidth allocation of the base station (BS) in a UAV assisted IoT communication network. Wherein, the access competition among groups of UAVs is modeled as a dynamical evolutionary game. The bandwidth allocation of BSs is formulated as a noncooperative game. Authors in [9] examined resource allocation for a multi-TE MEC offloading system, which is formulated as a convex optimization problem for minimizing the weighted sum of mobile energy consumption.

Chunlin Li *et al.* [10] analysed a radio and computing resource allocation problem between an access point and multiple devices in MEC system. They designed a time

average computation rate maximization algorithm to determine the optimal transmit power, and time allocation for the wireless devices. Junhui Zhao *et al.* [11] studied a cloud-MEC collaborative computation offloading problem that offloads tasks to automobiles in MEC vehicular networks. They developed a tasks allocation optimization and collaborative computation offloading scheme to decide the optimal strategies. An offloading algorithm for shortening the computation time and increasing the system utility was also designed. Yunlong Gao *et al.* [12] studied the optimal tradeoff between resource consumption and user experience in designing MEC systems. Cosmin Avasalcai *et al.* [13] introduced a decentralized resource management algorithm with the purpose of deploying IoT applications at the edge of the network such that end-to-end delay is minimized.

Since the resources of an ES are limited, the ES can not undertake the tasks coming from multiple TEs. So multiple ESs are needed. Nevertheless, the matching problem between multiple ESs and multiple TEs becomes a key issue. Heli Zhang *et al.* [14] modeled the matching relationship between ESs and TEs as a commodity trading by applying a multi-round sealed sequential combinational auction mechanism. In [15], the authors studied task offloading in vehicular MEC environments and modelled the interactions between edges and tasks as a matching game. They further developed two standalone heuristic algorithms to minimize the average delay while taking the energy consumption and vehicle mobility constraints into consideration. A three-tier IoT fog network was proposed in [16], in which all fog nodes, data service operators and data service subscribers are jointly optimized to achieve the optimal resource allocation in a distributed fashion.

Furthermore, authors in [17, 18] adopted a price-based mechanism to design efficient resource allocation in a MEC network. For example, [17] proposed a price-based distributed method to manage the offloaded tasks from users. Wherein, edge cloud sets prices to maximize its revenue and each user makes an optimal decision to minimize her/his own cost. The work [18] proposed a price-based resource allocation mechanism among the MEC server and multiple base stations (BSs). The MEC server tries to provide prices to BSs so as to maximize its own revenue while the BSs determine the computing space to improve the quality of experience.

To summarize the related work above, we observe that the existing resource allocation and matching problem in MEC generally involves edge nodes and clients using resource from an edge node. However, most of these studies focus either on the system performance or ESs' benefits, while ignoring the TEs' pursuit of maximizing payoff. Against this backdrop, our paper tries to balance the objectives of both ESs and TEs. In this paper, we also adopt a priced-based supply bidding mechanism to solve the resource allocation problem.

3 System Model

3.1 Interaction between TEs and ESs

As shown in Fig. 2, we consider a scheduler-based resources transaction market, which consists of M ESs and N TEs in a MEC network. ESs act as suppliers who sell computing resources to TEs and TEs act as customers who purchase resources from ESs. The bidirectional interaction between ESs and TEs is performed through a scheduler, which serves as a third-party agency outsourcing TEs' tasks to ESs.

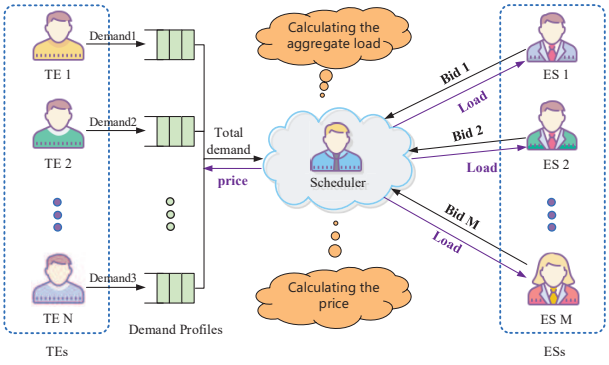

Fig. 2: Diagram of a resources transaction market.

On the one hand, TEs submit their demand profiles to the scheduler via a communication network. On the other hand, ESs compete with each other for acquiring more TEs, and submit bids based on strategies of their opponents and their own resource capacities. As a response, the scheduler calculates the service price and the aggregated load based on ESs' bids and TEs' total demand. They are mutually dependent upon each other as decisions on either side can have a bearing on those of the other side. After receiving the real-time price signal, the TEs will update their demand profiles. Since the aggregate load depends on the TEs' demand profiles, the behavior of TEs will affect the ESs' bidding strategies. The aforementioned process is repeated until both customers and suppliers are satisfied.

We divide one day into a set of T ($T = 24$) time slots, denoted as $\mathcal{T} = \{1, \cdots, T\}$. The set of ESs and TEs are represented as $\mathcal{M} = \{1, \cdots, M\}$ and $\mathcal{N} = \{1, \ldots, N\}$. How many resources should ESs provide to the market and how the ESs' bids affect the TEs' demand profiles are questions worth investigation. We next present the model of both sides in the MEC resources transaction market.

3.2 Cost and profit of ES

For ES $j \in \mathcal{M}$, let $C_{j,t}(.)$ denote the cost function of ES j at time slot t ($t \in \mathcal{T}$). Let $R_{j,t}(.)$ denote the revenue function of ES j at the tth time slot by providing the computational load. The profit of ES equals the revenue by providing computing service minus its cost of system overhead. Therefore, the profit $P_{j,t}$ of ES j at time slot t can expressed as follows:

$$P_{j,t} = R_{j,t}(.) - C_{j,t}(.). \tag{1}$$

We consider that each ES is selfish and tries to maximize its own profit. Thus, the interaction among the profit maximizer ESs can be modeled as a noncooperative game. The ESs are the players while the bid profiles are the strategies. Let $\lambda_{j,t}$ denote the bid of ES j at time slot t. The target of each ES j is to find the optimal bid $\lambda_{j,t}$ to maximize its profit, which can be defined as:

$$\underset{\lambda_{j,t}}{\text{maximize}} \quad P_{j,t} \quad j \in \mathcal{M}, t \in \mathcal{T}. \tag{2}$$

By substituting Equ. (1) into Equ. (2), we can get

$$\underset{\lambda_{j,t}}{\text{maximize}} \quad R_{j,t}(.) - C_{j,t}(.) \quad j \in \mathcal{M}, t \in \mathcal{T}. \tag{3}$$

We denote by $f_{j,t}$ the task load that ES j willing to generate in the time slot t. We assume that the service price of different ESs in one time slot is the same and denoted as $p_e(t)$ at time slot t. The revenue of each ES is equal to the product of its load and the service price. Hence, the revenue of ES j at time slot t can be represented as

$$R_{j,t} = f_{j,t} \cdot p_e(t). \tag{4}$$

Similar to [19], the ES j's cost function is defined as a quadratic function

$$C_{j,t}\left(f_{j,t}\right) = a_{j,2} f_{j,t}^2 + a_{j,1} f_{j,t} + a_{j,0},$$

where $a_{j,2}$, $a_{j,1}$ and $a_{j,0}$ are positive coefficients and model the fact that different ESs incur different costs for serving the tasks. We note that the cost function is increasing and convex. Substituting Equ. (4) into Equ. (3), the optimization problem can be further rewritten as

$$\begin{aligned} \underset{\lambda_{j,t}}{\text{maximize}} \quad & f_{j,t} \cdot p_e(t) - C_{j,t}(f_{j,t}) \\ \text{subject to} \quad & f_{j,t} \geq 0, \quad j \in \mathcal{M}, t \in \mathcal{T}. \end{aligned} \tag{5}$$

3.3 Payoff and payout of TE

The demand of each TE consists of two parts: a base demand and a shiftable demand. On the one hand, a base demand is primarily concerned with real-time tasks, which have high priority. On the other hand, a shiftable demand has low priority real-time requirements and it can be assigned at any time slot. The shiftable demand profile of TE i $(i \in \mathcal{N})$ is defined as $\boldsymbol{\chi}_i = (\chi_{i,1}, \ldots, \chi_{i,T})$ and the base demand of TE i at time slot t is denoted as $r_{i,t}$, which is known and fixed.

The utility of TE i represents the profit that TE i receives when it completes tasks and is denoted as $U_i(.)$. Exactly, the utility function of TE i is the utility for the tasks rather than the service time or applications. Similar to [20], we employ the quadratic utility function because it is non-decreasing and its marginal benefit is non-decreasing,

$$U_i(x) = \begin{cases} w_{i,t} x - \dfrac{\alpha_{i,t}}{2} x^2, & 0 \leq x \leq \dfrac{w_{i,t}}{\alpha_{i,t}} \\ \dfrac{w_{i,t}^2}{2\alpha_{i,t}}, & x > \dfrac{w_{i,t}}{\alpha_{i,t}} \end{cases}, \tag{6}$$

where $x = (\chi_{i,t} + r_{i,t})$, $w_{i,t}$ and $\alpha_{i,t}, i \in \mathcal{N}$ are coefficients that reflects the dynamic changes of TE i's demand.

The payout function quantifies the payout that TE i needs to pay the ESs task completion. Without loss of generality, we define the payout of TE i' as the product of demand and the service, i.e.

$$Payout_{i,t} = (\chi_{i,t} + r_{i,t}) \cdot p_e(\boldsymbol{\lambda}_t, L_t). \tag{7}$$

The payoff function quantifies the final benefits of TE i and represents the satisfaction of using the service. Thus, we denote the payoff of TE i as its utility minus payout i.e.

$$Payoff_i = Utility_i - Payout_i. \tag{8}$$

Let u_i denote the payoff of TE i. By substituting Equ. (6) and Equ. (7) into Equ. (8), we can obtain

$$\begin{aligned} u_i\left(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}\right) = \sum_{t \in \mathcal{T}} \Big(& U_i\left(\chi_{i,t} + r_{i,t}\right) \\ & - \left(\chi_{i,t} + r_{i,t}\right) p_e\left(\boldsymbol{\lambda}_t, L_t\right) \Big), \end{aligned} \tag{9}$$

where $\boldsymbol{\chi}_{-i}$ denotes the vector of the demand profile of other TEs and $\boldsymbol{\chi}_{-i} = (\boldsymbol{\chi}_1, \ldots, \boldsymbol{\chi}_{i-1}, \boldsymbol{\chi}_{i+1}, \ldots, \boldsymbol{\chi}_N)$. In Equ. (9), the utility is a function related to $(\chi_{i,t} + r_{i,t})$.

Each TE tries to maximize its payoff by determining its shiftable demand profile. Thus, the interaction between TEs can be modeled as a noncooperative game. The TEs are participants while the shiftable demand profiles are the strategies of the noncooperative game.

Let $\boldsymbol{\chi}_i^*$ denote the optimal demand profile of TE i in the Nash equilibrium and Q_i^{total} denote the total daily shiftable demand of TE i which is fixed and known. Let L_t denote the aggregate load demand of the ESs at time slot t and $L_t = \sum_{j \in \mathcal{N}} \left(\chi_{j,t} + r_{j,t}\right)$. Considering TE i, the optimization problem can be formulated as follows when other TEs' profiles are fixed:

$$\begin{aligned} \underset{\boldsymbol{\chi}_i}{\text{maximize}} \quad & u_i\left(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}\right) \\ \text{subject to} \quad & \sum_{t \in \mathcal{T}} \chi_{i,t} = Q_i^{\text{total}}, \\ & \chi_{i,t} \geq 0, \forall i \in \mathcal{N}. \end{aligned} \tag{10}$$

3.4 Market mechanism with supply function bidding

In this section, we employ a supply function bidding mechanism to model the relationship between market demand for services and its price. We use a class of supply functions with parameters. The bids submitted by ESs reveal their available resource capacities "supplied" to the market.

TABLE 1: Definitions of Mathematical Notations

Notation	Definition
L_t	TEs' total load demand at time slot t
$f_{j,t}$	The supply function of the ES j at time slot t
$p_e(t)$	The computing service price at time slot t
$p_1, \cdots, p_K$	K break points of the price-wise linear function of all ESs
$\lambda_{j,t}^k$	The slope of the function between the break points p_{k-1} and p_k
$\lambda_{j,t}^1$	The slope of the function between the origin and break point p_1

The notations used in the supplier side model are presented in Table 1. We assume that the supply function $f_{j,t}$ is chosen from the family of increasing and convex price-wise linear functions of $p_e(t)$ [21]. Fig. 3(a) shows an increasing and convex piece-wise linear supply function. The abscissa $p_e(t)$ indicates the price and the ordinate $f_{j,t}$ denotes the load supplied by the TE j at time slot t. There exists K break points on the abscissa of the Fig. 3(a). $\lambda_{j,t}^k \geq 0$ represents the slope of the function between the break points p_{k-1} and p_k. Fig. 3(b) shows the affine supply function.

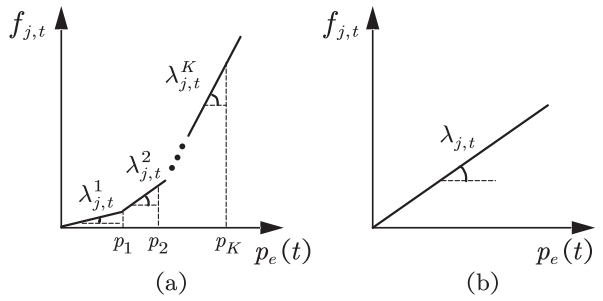

Fig. 3: (a) Piece-wise linear. (b) Affine supply functions.

At time slot t ($t \in \mathcal{T}$), we use the vector $\boldsymbol{\lambda}_{j,t} = (\lambda_{j,t}^1, \cdots, \lambda_{j,t}^K)$ to denote the bid profile of ES j ($j \in \mathcal{M}$). Thus, we obtain

$$f_{j,t}\left(p_e(t), \boldsymbol{\lambda}_{j,t}\right) = \begin{cases} \lambda_{j,t}^1 p_e(t), & 0 \le p_e(t) \le p_1 \\ \lambda_{j,t}^k p_e(t) + \lambda_{j,t}^{k-1} p_{k-1}, & p_{k-1} < p_e(t) \le p_k \end{cases} \tag{11}$$

It is assumed that each ES submits $\boldsymbol{\lambda}_{j,t}$ as a bid profile to the scheduler at time slot t. For each ES j, the bid profile describes the number of tasks that it is willing to admit. We can use $\boldsymbol{\lambda}_t$ to represent the bid profiles of all ESs at time slot t and $\boldsymbol{\lambda}_t = \{\boldsymbol{\lambda}_{1,t}, \cdots, \boldsymbol{\lambda}_{M,t}\}$. In response to ESs, the scheduler sets the price $p_e(t)$ to clear market. In economics, market clearing means the supply of what is traded equals the demand, so that there is no leftover supply or demand. In this case, the demand of all TEs is the same as the load supplied by all ESs. Although the fluctuation in TEs' demand will drive changes in ESs' bid profiles, the demand and supply remains balanced. The equivalence further builds up the connection between the supplier game and the customer game. Hence, it can expressed as

$$\sum_{j \in \mathcal{M}} f_{j,t}\left(p_e(t), \boldsymbol{\lambda}_{j,t}\right) = L_t, \quad t \in \mathcal{T}. \tag{12}$$

According to Equ. (11) and Equ. (12), we have

$$L_t = \begin{cases} \sum_{j \in \mathcal{M}} \left(\lambda_{j,t}^1 p_e(t)\right), & 0 \le p_e(t) \le p_1 \\ \sum_{j \in \mathcal{M}} \left(\lambda_{j,t}^k p_e(t) + \lambda_{j,t}^{k-1} p_{k-1}\right), & p_{k-1} < p_e(t) \le p_k \end{cases}. \tag{13}$$

According to Equ. 13, we can further calculate the service price function as follows:

$$p_e(t) = \begin{cases} \dfrac{L_t}{\sum_{j \in \mathcal{M}} \lambda_{j,t}^1}, & 0 \le p_e(t) \le p_1 \\ \dfrac{L_t - \sum_{j \in \mathcal{M}} \left(\lambda_{j,t}^{k-1} p_{k-1}\right)}{\sum_{j \in \mathcal{M}} \lambda_{j,t}^k}, & p_{k-1} < p_e(t) \le p_k. \end{cases} \tag{14}$$

At time slot t, the service price of different ESs is the same.

In [22], the affine supply function $f_{j,t}\left(p_e(t), \boldsymbol{\lambda}_{j,t}\right) = \lambda_{j,t}^1 p_e(t)$ is used as a special case of the aforementioned piece-wise linear functions. Almost all the results of affine supply functions can be generalized to the piece-price affine supply function [23]. As for Equ. (14), it can be concluded that the affine function is equivalent to the piece-wise linear supply function between two break points. Each piecewise function of Fig. 3(a) can be regarded as a linear function in Fig. 3(b). As a matter of fact, the term $\lambda_{j,t}^{k-1} p_{k-1}$ is fixed when we are between break points p_{k-1} and p_k. Therefore, without loss of generality, we generalize the results from the affine functions to piece-wise linear functions. Therefore, the computing service price can be given as follows for an affine supply function:

$$p_e(t) = \frac{L_t}{\sum_{j \in \mathcal{M}} \lambda_{j,t}^1}, \quad t \in \mathcal{T}. \tag{15}$$

For simplicity, we use the notation $\lambda_{j,t}$ instead of $\lambda_{j,t}^1$ to represent the affine supply function of ES j. Meanwhile, we use $\boldsymbol{\lambda}_t = (\lambda_{1,t}, \ldots, \lambda_{M,t})$ to denote the bids profile for all ESs at time slot t. As Equ. (15) shows, the computing service price is related to $\lambda_{j,t}$ ($j \in \mathcal{M}$) and L_t. Hence, the price function can be denoted as $p_e\left(\boldsymbol{\lambda}_t, L_t\right)$. As suggested by Equ. (11), supply function $f_{j,t}$ for ES j can be expressed as

$$f_{j,t}\left(p_e\left(\boldsymbol{\lambda}_t, L_t\right), \lambda_{j,t}\right) = \frac{\lambda_{j,t} L_t}{\sum_{r \in \mathcal{M}} \lambda_{r,t}}, \quad t \in \mathcal{T}. \tag{16}$$

Similar to the computing service, the supply function can be represented by $f_{j,t}\left(\boldsymbol{\lambda}_t, L_t\right)$. Let $\boldsymbol{\lambda}_{-j,t}$ denote the submitted bids of other ESs except for ES j. So it can be defined as $\boldsymbol{\lambda}_{-j,t} = (\lambda_{1,t}, \ldots, \lambda_{j-1,t}, \lambda_{j+1,t}, \ldots, \lambda_{M,t})$. Hence, According to Equ. (5) and Equ. (16), the profit function of ES j is rewritten as

$$P_{j,t}(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}) = \frac{\lambda_{j,t} L_t^2}{\left(\sum_{r \in \mathcal{M}} \lambda_{r,t}\right)^2} - C_j\left(\frac{\lambda_{j,t} L_t}{\sum_{r \in \mathcal{M}} \lambda_{r,t}}\right). \tag{17}$$

When other ESs' bids are fixed, the ES j tries to find the optimal bid $\lambda_{j,t}^*$ by solving the following optimization problem:

$$\begin{aligned} \underset{\lambda_{j,t}}{\text{maximize}} \quad & \frac{\lambda_{j,t} L_t^2}{\left(\sum_{r \in \mathcal{M}} \lambda_{r,t}\right)^2} - C_j\left(\frac{\lambda_{j,t} L_t}{\sum_{r \in \mathcal{M}} \lambda_{r,t}}\right) \\ \text{subject to} \quad & \lambda_{j,t} \ge 0, \quad j \in \mathcal{M}, t \in \mathcal{T}. \end{aligned} \tag{18}$$

3.5 Nash equilibrium analysis

The following section will explain that the ES's game (Equ. (18)) has a unique Nash equilibrium, as shown by the lemma below.

Lemma 3.1. *Assume that the bids profile in Nash equilibrium at time slot t is denoted as $\boldsymbol{\lambda}_t^*$. When the Nash equilibrium is reached, it will satisfy $\lambda_{j,t}^* < \sum_{r \in \mathcal{M}, r \ne j} \lambda_{r,t}^*$ for all ESs.*

Proof. The function $\Pi_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)$ is expressed as follows:

$$\Pi_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right) = \frac{\lambda_{j,t} L_t^2}{\left(\sum_{r \in \mathcal{M}} \lambda_{r,t}\right)^2}. \tag{19}$$

As the formula above suggests, $\Pi_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)$ is the first term in $P_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)$. From Equ. (19), we can calculate the first derivative function as follows

$$\begin{aligned} &\frac{d\Pi_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)}{d\lambda_{j,t}} \\ &= \frac{L_t^2 \cdot \left(\sum_{r\in\mathcal{M}} \lambda_{r,t}\right)^2 - 2\lambda_{j,t} L_t^2 \left(\sum_{r\in\mathcal{M}} \lambda_{r,t}\right)}{\left(\sum_{r\in\mathcal{M}} \lambda_{r,t}\right)^4} \end{aligned} \tag{20}$$

Let

$$\frac{d\Pi_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)}{d\lambda_{j,t}} > 0,$$

we can get

$$\left(\sum_{r\in\mathcal{M}} \lambda_{r,t}\right)^2 - 2\lambda_{j,t} \sum_{r\in\mathcal{M}} \lambda_{r,t} > 0. \tag{21}$$

The Equ. (21) is equivalent to

$$\left(\lambda_{j,t} + \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}\right)^2 - 2\lambda_{j,t}\left(\lambda_{j,t} + \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}\right) > 0. \tag{22}$$

From Equ. (22), we can derive that

$$0 \leq \lambda_{j,t} < \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}.$$

In summary, we can conclude that $P_{j,t}\left(\lambda_{j,t}, \boldsymbol{\lambda}_{-j,t}\right)$ is an increasing function when $0 \leq \lambda_{j,t} < \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}$. And it becomes a decreasing function when $\lambda_{j,t} \geq \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}$. Thus, in order to maximize profit, we should meet the constraint $0 \leq \lambda_{j,t} < \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}$. In the Nash equilibrium, the bid of ES j at time slot t is denoted as $\lambda_{j,t}^*$. Therefore, we can conclude that $\lambda_{j,t}^* < \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}^* \ \ (j \in \mathcal{M})$. □

Similar to [24], the proof for the following theorem given as follows.

Theorem 3.1. *The ES's noncooperative game has a unique Nash equilibrium. Furthermore, the Nash equilibrium is the solution of the following convex optimization problem:*

$$\begin{aligned} \underset{0\leq f_{j,t}<\frac{L_t}{2}}{\text{maximize}} \quad & \sum_{j\in\mathcal{M}} -\Psi_j\left(f_{j,t}\right) \\ \textit{subject to} \quad & \sum_{j\in\mathcal{M}} f_{j,t} = L_t, \end{aligned} \tag{23}$$

where

$$\Psi_j\left(s_{j,t}\right) = \left(\frac{L_t - f_{j,t}}{L_t - 2f_{j,t}}\right) C_j\left(f_{j,t}\right) - \int_0^{f_{j,t}} \frac{L_t C_j\left(\Pi_j\right)}{\left(L_t - 2\Pi_j\right)^2} d\Pi_j. \tag{24}$$

Proof. According to lemma 3.1, we can infer that the load supplied by each ES at time slot t does not exceed $L_t/2$ at the Nash equilibrium. The Lagrange function of the optimization problem in Equ. (23) is denoted as F. Thus, we have

$$F = \sum_{j\in\mathcal{M}} -\Psi_j\left(f_{j,t}\right) + \phi\left(\sum_{j\in\mathcal{M}} f_{j,t} - L_t\right), \tag{25}$$

where ϕ denotes the Lagrange multiplier. We can obtain the following expression through the first-order optimality function.

$$\left(\frac{\partial F}{\partial f_{j,t}^*}\right)\left(f_{j,t} - f_{j,t}^*\right) \leq 0, \quad \forall j \in \mathcal{M}, \tag{26}$$

where $f_{j,t}^*$ is defined as the supply function in equilibrium, while ϕ^* is the Lagrange multiplier in equilibrium.

From Equ. (24), $(\partial F/\partial f_{j,t}^*)$ can be expressed as follows:

$$\frac{\partial F}{\partial f_{j,t}^*} = \phi^* - \left(\frac{L_t - f_{j,t}^*}{L_t - 2f_{j,t}^*}\right) C_j'\left(f_{j,t}^*\right). \tag{27}$$

We assume the first-order optimality condition for the optimization problem in Equ. (18). Thus, we obtain

$$\left(\frac{\partial P_{j,t}}{\partial \lambda_{j,t}}\right)\left(\lambda_{j,t} - \lambda_{j,t}^*\right) \leq 0, \quad \forall j \in \mathcal{M}. \tag{28}$$

From Equ. (17), $\partial P_{j,t}/\partial\lambda_{j,t}$ is calculated as follows:

$$\frac{\partial P_{j,t}}{\partial \lambda_{j,t}} = p_e\left(\boldsymbol{\lambda}_t, L_t\right) - \frac{L_t - f_{j,t}^*}{L_t - 2f_{j,t}^*} C_j'\left(f_{j,t}^*\right). \tag{29}$$

By substituting Equ. (29) into Equ. (28), we can write the optimality condition for Nash equilibrium as follows

$$\left(p_e\left(\boldsymbol{\lambda}_t, L_t\right) - \frac{L_t - f_{j,t}^*}{L_t - 2f_{j,t}^*} C_j'\left(f_{j,t}^*\right)\right)\left(\lambda_{j,t} - \lambda_{j,t}^*\right) \leq 0. \tag{30}$$

From Equ. (26) and Equ. (30), we can see that the Lagrange multiplier is actually the price $p_e(\boldsymbol{\lambda}_t, L_t)$ of the computing service. In addition, the optimality condition Equ. (26) is equivalent to Equ.30. Therefore, the existence and uniqueness of the Nash equilibrium is equivalent to proving the existence and uniqueness of the optimal point of problem Equ. (23). □

In Theorem 3.1, it is proved that the ES's game has a unique Nash equilibrium solution, whose strategies are determined by the aggregate load L_t. Besides, a ES can scale-up and scale-down its resource capacity according to different market demands. Thus, ESs will bid differently for different levels of load.

We next analyze the existence of Nash equilibrium for the customer side game, which is proved by the theorem below.

Theorem 3.2. *The customers' optimization problem is a convex programming problem. In fact, the customer side game Equ. (10) is an n-person game. It has a unique pure strategy Nash equilibrium.*

Proof. From the above discussion it follows that the objective function in Equ. (10) is equal to

$$\begin{aligned} &\sum_{t\in\mathcal{T}} U_i\left(\chi_{i,t} + r_{i,t}\right) \\ &\quad - \sum_{t\in\mathcal{T}} \left(\frac{\left(\chi_{i,t} + r_{i,t}\right)^2 + \sum_{j\in\mathcal{N}, j\neq i}\left(\chi_{j,t} + r_{j,t}\right)}{\sum_{r\in\mathcal{M}} \lambda_{r,t}}\right). \end{aligned} \tag{31}$$

Let $k = 1/(\sum_{r\in\mathcal{M}} \lambda_{r,t}), k > 0$. For simplicity, we denote the right part of Equ. (31) as follows:

$$h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) = \sum_{t\in\mathcal{T}} k\Bigg(\left(\chi_{i,t} + r_{i,t}\right)^2 + \sum_{j\in\mathcal{N}, j\neq i}\left(\chi_{j,t} + r_{j,t}\right)\Bigg).$$

We have

$$\begin{aligned}\nabla_{\boldsymbol{\chi}_i} h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) &= \left[\frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,t}}\right]_{t=1}^{T} \\ &= \left(\frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,1}}, \cdots, \frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,T}}\right) \\ &= 2k\left[(\chi_{i,t} + r_{i,t}) + \sum_{j\in\mathcal{N}, j\neq i} (\chi_{j,t} + r_{j,t})\right]_{t=1}^{T}\end{aligned} \tag{32}$$

and the Hessian matrix is as follows:

$$\nabla^2_{\boldsymbol{\chi}_i} h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) = \begin{pmatrix} 2k & 2k & \cdots & 2k \\ 2k & 2k & \cdots & 2k \\ \vdots & \vdots & \ddots & \vdots \\ 2k & 2k & \cdots & 2k \end{pmatrix}_{N\times T}. \tag{33}$$

This further leads to

$$X^{\mathrm{T}} \nabla^2_{\boldsymbol{\chi}_i} h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) X = 2k\left(X_1 + X_2 + \cdots + X_{N\times T}\right)^2 \geq 0,$$

$$\forall X = (X_1, X_2, \cdots, X_{N\times T})^{\mathrm{T}}.$$

Therefore, the Hessian matrix of $h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})$ is positive semi-definite and $h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})$ is convex. Moreover, since the utility function $U_i(.)$ is continuous and strictly concave in the strategy space, the payoff function Equ. (9) of each TE i $(\forall i \in \mathcal{N})$ is strictly concave. So the objective function in Equ. (31) is concave. Hence, Equ. (10) is a convex optimization problem. Meanwhile, since the constraints of Equ. (10) are inequalities or linear equations, the feasible domain is convex. Thus, the TEs' optimization problem is a convex programming problem. Hence, the TE's game is a strictly concave N-person game. Since the demand profile sets are closed, bounded and convex, the existence of Nash equilibrium can be proved based on [25, Theorem 1]. Analogously to [25, Theorem 3], for a concave N-person game, there exists a unique equilibrium solution. Therefore, the theorem is proved. □

In the Nash equilibrium, for any given ESs' bids, no TE can increase its payoff by a unilateral change on its strategy. In the next section, a task outsourcing algorithm is developed to determine the point for both ES and TE's games.

4 Distributed Task Outsourcing Algorithm

In this section, we propose a distributed task outsourcing algorithm to demonstrate the interaction among TEs and ESs. Our method is referred as ***DTOA***. Let g be the iteration number.

Notations:

Let $\chi^g_{i,t}$ denote the demand profiles of TE i in iteration g at time slot t and vector $\boldsymbol{\chi}^g_i$ denote the demand profile of TE i for all time slots. The matrix $\boldsymbol{\chi} = (\boldsymbol{\chi}_1, \ldots, \boldsymbol{\chi}_t, \ldots, \boldsymbol{\chi}_T)^T$ denotes the demand profiles of all TEs in iteration g for all time slots. Let matrix $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_t, \ldots, \boldsymbol{\lambda}_T)^T$ denote the bids of all ESs for all time slots. L^g_t denotes the aggregate loads in iteration g at time slot t. $p^g_e(\boldsymbol{\lambda}^g_t, L^g_t)$ denotes the computing service price in iteration g at time slot t.

As shown in Fig. 4, the interaction between ESs and TEs can be modeled as a two-stage game. They interact with each other to determine optimal bids and demand profiles. The detailed process is depicted in Algorithm 1 and 2.

- The ESs try to maximize their profits by determining their own bids according to optimization function Equ. (18).
- The TEs will then adjust their demand profiles following optimization function Equ. (10).

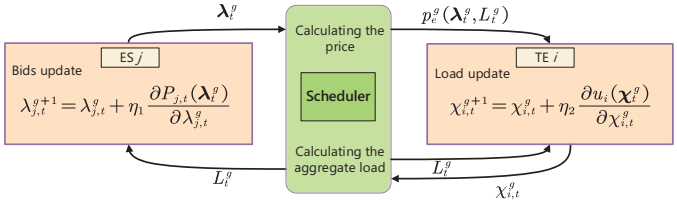

Fig. 4: Interactions between the ESs, TEs and scheduler.

Algorithm 1 TE's game

Initialization: $g = 0$.
Randomly initialize TEs' demand profiles.
repeat
  for (each time slot $t \in \mathcal{T}$) **do**
    Receive L^g_t from the scheduler.
    Update the bid $\boldsymbol{\lambda}^g_t$ by Algorithm 2.
    Receive the updated $p^g_e(\boldsymbol{\lambda}^g_t, L^g_t)$ from the scheduler.
    for (each TE $i \in \mathcal{N}$) **do**
      $\chi^{g+1}_{i,t} = \left[\chi^g_{i,t} + \eta_2 \frac{\partial u_i(\boldsymbol{\chi}^g_t)}{\partial \chi^g_{i,t}}\right]^+$.
    end for
  end for
  $g := g + 1$.
until $\left\|\boldsymbol{\chi}^g - \boldsymbol{\chi}^{g-1}\right\| < \epsilon$.

Algorithm 2 ES's game

Input: Total load at time slot t: $L_t, t \in \mathcal{T}$ and t.
Output: Bids of all ESs at time slot t: $\boldsymbol{\lambda}_t$.
Initialization: Randomly initialize ESs' bid profiles for the first time.
Receive L_t from the scheduler.
for (each ES $j \in \mathcal{M}$) **do**
  $\lambda^{g+1}_{j,t} = \left[\lambda^g_{j,t} + \eta_1 \frac{\partial P_{j,t}(\boldsymbol{\lambda}^g_t)}{\partial \lambda^g_{j,t}}\right]^+$.
end for
return $\boldsymbol{\lambda}_t$.

The ***DTOA*** can be described as follows. Firstly, the scheduler randomly initializes the TEs' demand profiles and ESs' bid profiles. Secondly, the TE i $(i \in \mathcal{N})$ sends the shiftable demand profile $\boldsymbol{\chi}^g_i$ to the broker and receives L^g_t from it. Then, the ESs will receive a signal to update their bids based on the following iterative equation:

$$\lambda^{g+1}_{j,t} = \left[\lambda^g_{j,t} + \eta_1 \frac{\partial P_{j,t}(\boldsymbol{\lambda}^g_t)}{\partial \lambda^g_{j,t}}\right]^+, \quad \forall t \in \mathcal{T}. \tag{34}$$

where η_1 is the step size. $[\cdot]^+$ in Equ. (34) is the projection onto the feasible set defined by the constraints $\lambda_{j,t} \geq 0$.

It is noticed that the ES j ($j \in \mathcal{M}$) does not know other ESs' bids. In this aspect, the ***DTOA*** can also preserve the privacy of participants. Thirdly, the computing service price $p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$ is updated by the scheduler according to Equ. (15). The TEs will further be informed to update their shiftable demand profiles using a gradient boosting method:

$$\chi_{i,t}^{g+1} = \left[\chi_{i,t}^g + \eta_2 \frac{\partial u_i(\boldsymbol{\chi}_t^g)}{\partial \chi_{i,t}^g}\right]^+, \quad \forall t \in \mathcal{T}. \tag{35}$$

η_2 is the step size. $[\cdot]^+$ in Equ. (35) is the projection onto the feasible set defined by the constraints $\sum_{t\in\mathcal{T}} \chi_{i,t} = Q_i^{\text{total}}$ and $\chi_{i,t} \geq 0$. It is worth remarking that Equ. (10) needs the updated price $p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$ and L_t^g to determine $\left(\partial u_i(\boldsymbol{\chi}_t^g)/\partial \chi_{i,t}^g\right)$. Besides, since $\left(\partial u_i(\boldsymbol{\chi}_t^g)/\partial \chi_{i,t}^g\right)$ only depends on its own demand profile and the price and there is no need to know the demand profile of other TEs. Thus, this fact protects the privacy of the TEs. Finally, the stopping criterion of the algorithm is checked by the scheduler. If the relative change of shiftable demand profiles during two consecutive iterations is lower than the value ϵ, the iterations can be stopped. Otherwise, the TEs will continue computing their demand profiles based on the newly updated price and bids.

The optimization problems Equ. (18) and Equ. (10) will converge to the optimal point by the projected gradient method. In the end, the algorithm will converge. In the equilibrium, the ESs are playing their equilibrium strategies according to TEs' tasks strategies, and the TEs also choose their equilibrium strategies based on ESs' submitted bids. Thus when the Nash equilibrium is reached, none of the ESs and TEs improve their profit.

5 Performance Evaluation

5.1 Simulation experiment

In this section, we present a simulation experiment to validate our theoretical analysis. We assume a MEC resource exchange market has 10 ESs and 1000 TEs, which are willing to participate in the ***DTOA*** scheme. There are 24 time slots. The relevant parameters of the model are shown in Table 2. The base demand $r_{i,t}$ of each TE at each time slot is randomly selected from [9660, 37065]. The shiftable demand refers to real-time, non-shiftable tasks, which reflects the changes in the total demand of all TEs at different time slots. Since most loads are running in real-time pattern, it is plausible to assume relatively low shiftable loads for TEs. The shiftable demand $\chi_{i,t}$ of each TE is assumed to be chosen randomly from 10% to 12% of its base demand. And the total demand is the sum of the base demand and shiftable demand. Considering the generation cost function $c(f_{j,t}) = a_{j,2}f_{j,t}^2 + a_{j,1}f_{j,t} + a_{j,0}$ for each ES j ($j \in \mathcal{M}$), we assume that $a_{j,2}$ is randomly generated in the interval [4.76e-6, 4.76e-5], $a_{j,1} = 0.001$ and $a_{j,0} = 0.001$. The initial values of η_1 and η_2 are set as 0.05 and 0.01 respectively. In order to find the optimal solution, the step size of next iteration will be a little less than the previous one, namely η_1=η_1*0.985 and η_2=η_2*0.98. The initial bids $\lambda_{j,t}$ of ESs are all set as 20000. The $\alpha_{i,t}$ is set as 0.5 and $\omega_{i,t}$ is randomly selected from interval [0.8, 1.0]. Also, the ϵ is set equal to 0.3.

TABLE 2: System Parameters

System parameters	Value(Fixed)-[Varied range]
Base demand $r_{i,t}$	[9660, 37065]
Shiftable demand $\chi_{i,t}$	[10%,12%]*Base demand
$a_{j,2}$	[4.76e-6, 4.76e-5]
$a_{j,1}$	(0.001)
$a_{j,0}$	(0.001)
step size η_1	(0.05), η_1=η_1*0.985
step size η_2	(0.001), η_2=η_2*0.98
ES's bid $\lambda_{j,t}$	(20000)
$\alpha_{i,t}$	(0.5)
$\omega_{i,t}$	[0.8,1.0]
ϵ	(0.3)

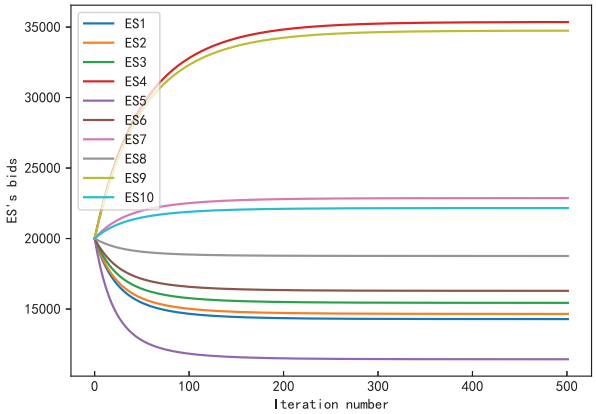

Fig. 5: Convergence of ES 1-10's bids at time slot 5.

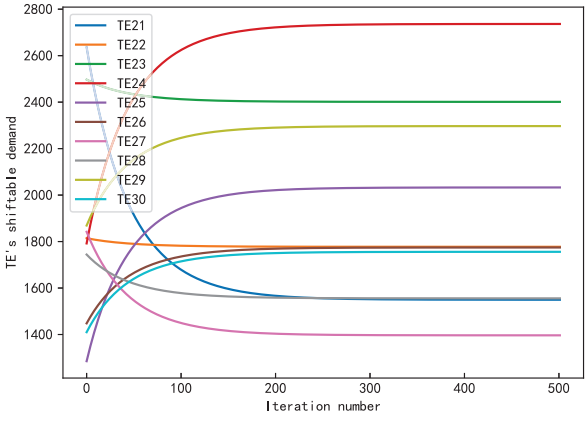

Fig. 6: Convergence of shiftable loads for TE 21-30 at time slot 5.

5.2 Algorithm Convergence

The performance of our proposed ***DTOA*** is evaluated in terms of its convergence. Fig. 5 and Fig. 6 show the convergence of ESs' bids and TEs' shiftable loads at time slot 5. From Fig. 6, these ten TEs (TEs 21-30) are randomly selected from 1000 TEs. The speed of convergence to the equilibrium point depends on the step sizes and the stopping criterion ϵ. As the number of iterations increases, the bids and the shiftable load demands start from the initial values and they gradually converge to stable values. In our experiment, the algorithm converges after around 248 iterations. Hence, the proposed ***DTOA*** is efficient and verifies the theoretical proof presented above.

To demonstrate the computational complexity of the algorithm, we evaluate the running time of the algorithm for different number of TEs and ESs. As shown in Fig. 7, the running time of the algorithm increases linearly with the number of TEs N and it is almost independent of M. This is because that by increasing the number of TEs and ESs, the number of updates for TEs and ESs will increase proportional to N and M, respectively. The update process for TEs takes more time comparing with the updates for ESs since the TEs need to consider load shifting during T time slots (the projected gradient), which make the update process more complex. From Fig. 7, the running time of the algorithm is acceptable even for large number of TEs. So it can be concluded that the algorithm is efficient and can be implemented in scenarios with large number of TEs.

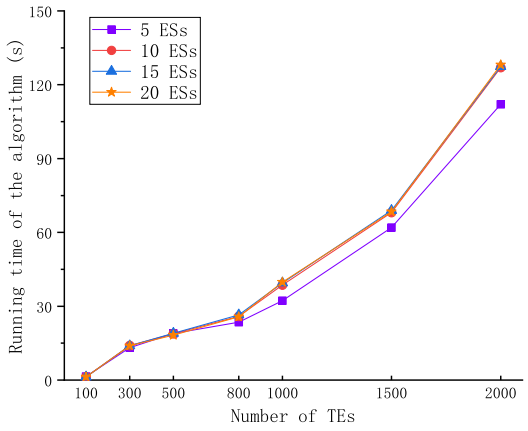

Fig. 7: Running time of algorithm for different number of TEs and ESs

5.3 Economic effects of algorithm

By participating in the ***DTOA*** scheme, the payout (see Equ. (7)) represents TE's expenditure on purchasing computing resources. Fig. 8 shows the daily total payout for TE 1 to TE 30 before algorithm and after algorithm. Compared with before algorithm, the total payout of each TE after algorithm is reduced. We can see that TEs can save around 5% of they payout by participating in ***DTOA*** scheme. The vertical axis of Fig. 8 shows the payouts of TEs are so huge, and even a 5% savings reduces a large expenditure. Furthermore, as shown in Fig. 9, the daily total payoff of each TE after using the algorithm has increased than before algorithm. The payoff (see Equ. (9)) is the utility of the calculation tasks minus the payout. Although the green bar is only a little more than the red bar chart, the payoff has also increased a lot because its magnitude is large and arrives 10^9.

Fig. 10 displays the total profit of ESs 1-10 before and after algorithm. The total profit of ES is the sum of the profit of all time slots. From Fig. 10, we can see that the total profit of each ES increases after applying ***DTOA*** because the aggregate load profile becomes smoother; and hence, the ESs' generation cost decreases. Besides, the suppliers aim to submit optimal bids that maximize their profits in each time slot. The results of the algorithm are in line with expectations, which shows the supplier side's individual rationality of our proposed method.

5.4 Peak-reducing effect of algorithm

For simulations, the initial state of TEs' demand is assumed to be load profile before algorithm. The aggregate load profile becomes smoother after the ***DTOA***. The dashed circle shows the fluctuation of the demand including valley filling and peak clipping. Normally, the peak load demand is 26200, while the peak load demand decreases to 23800 in the case of the ***DTOA***. Therefore, the peak load demands are shifted from peak to off-peak time slots. Furthermore, the load demand for each TE is shifted to time slots with higher $w_{j,t}$, which brings a higher payoff to the TEs. This demonstrates that the proposed ***DTOA*** performs satisfactorily in reducing the peak load demand.

Fig. 12 shows the PAR (peak-to-average ratio) index with and without task scheduling in 24 time slots and for different number of TEs. Before algorithm, since there are high peak load and low average load, PAR index is high. By applying ***DTOA***, the peak clipping and valley filling are achieved and the PAR index is low even for high number of TEs. This demonstrates that the proposed task scheduling method can shift the shiftable loads from peak periods to off-peak periods effectively.

5.5 Influence of parameters on iteration numbers

In this section, we discuss the influence of some parameters on the convergence speed of the algorithm. The convergence speed of the algorithm is reflected in the round of algorithm updates (iteration numbers). The smaller the iteration numbers, the faster the algorithm converges. The bigger the iteration numbers, the slower the algorithm converges.

Fig. 13 shows the influence of the parameter ϵ on iteration numbers. ϵ is the stopping criterion of the algorithm. As can be seen, the smaller the parameter ϵ, the more iterations and the slower the algorithm convergence. This fact shows that the stricter of the stopping criterion, the more times the algorithm needs to be updated.

In Fig. 14, the influence of the parameter η_2 on iteration numbers are shown. Since the parameter η_2 will change in every round, as shown in Table 2, we set different initial value of parameter η_2 to show its impact on iteration numbers. As can be seen, when other parameters are fixed, with the initial value of parameter η_2 becomes larger, the number of iterations also increases. In summary, the speed

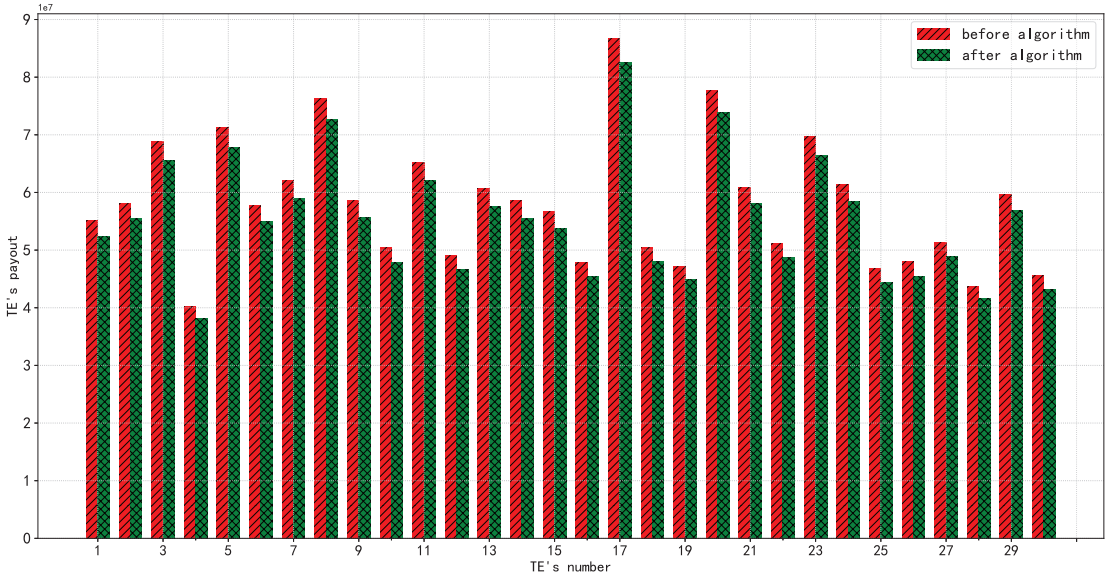

Fig. 8: Daily total payout for TE 1 to TE 30.

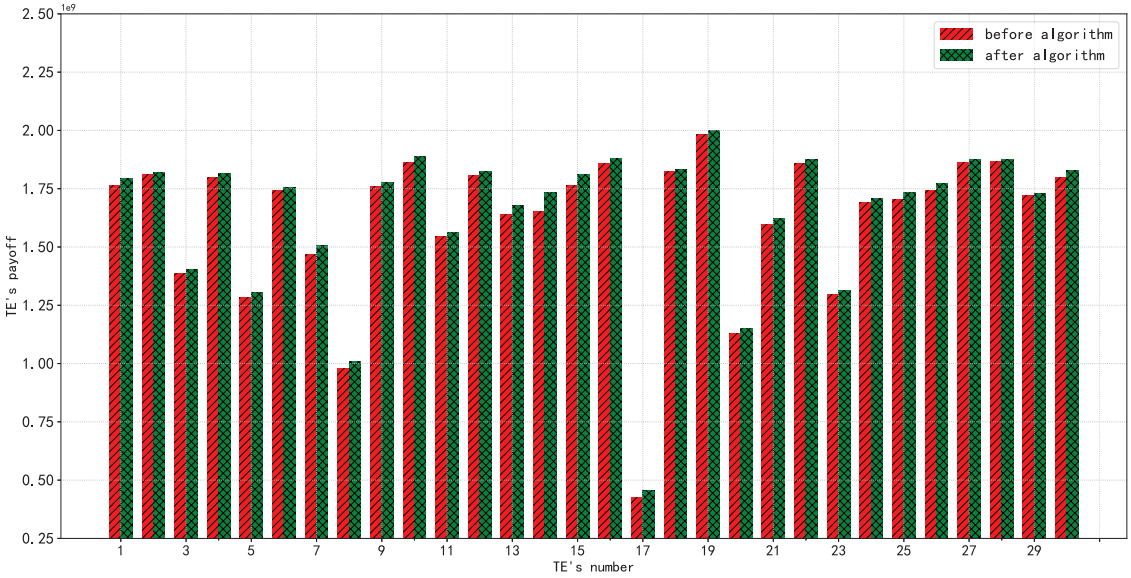

Fig. 9: Daily total payoff for TE 1 to TE 30.

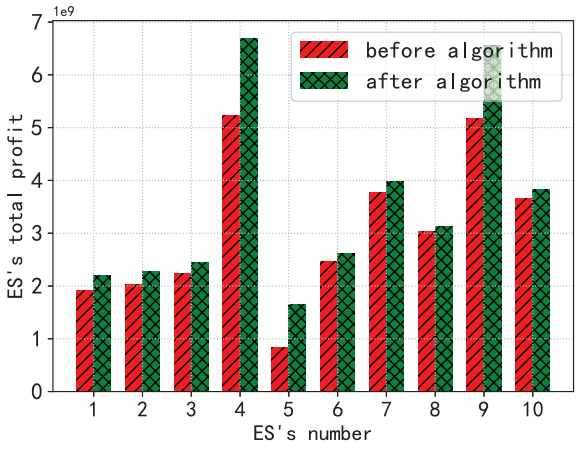

Fig. 10: Daily total profit for ES 1 to ES 10.

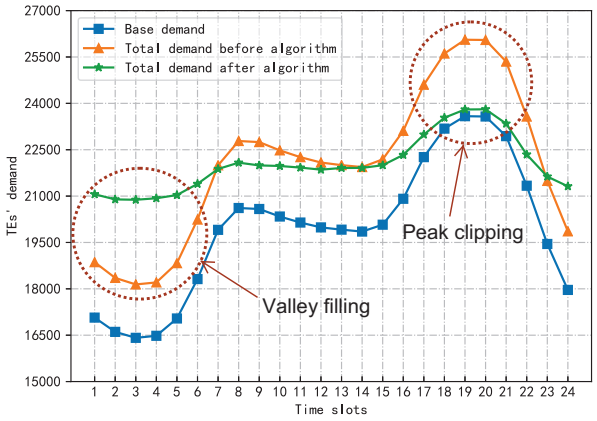

Fig. 11: Base and total demand before and after algorithm. The peak shaving is achieved by using ***DTOA*** (the dashed circle).

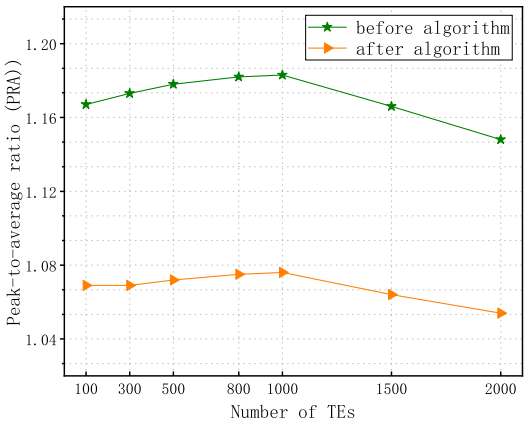

Fig. 12: Peak-to-average ratio with and without task scheduling

of the algorithm convergence is related to the setting of some parameters.

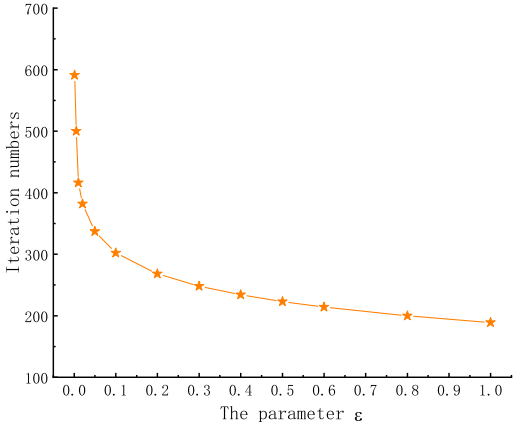

Fig. 13: The influence of the parameter ϵ on iteration numbers.

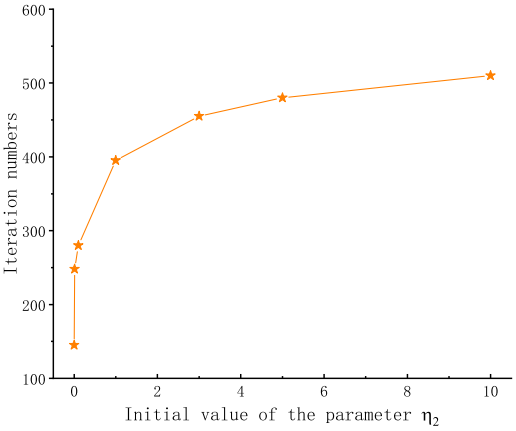

Fig. 14: The influence of the parameter η_2 on iteration numbers.

6 Conclusion

In this paper, we analyze a practical resources transaction market in a MEC network, where multiple different ESs offering the optional computing service to TEs. Since the resources of each ES are limited, the dynamic demand of its TEs may not be met during spikes in demands. To overcome the bottleneck of resource limitation, task outsourcing has been regarded as an effective paradigm by accommodating as many on-demand tasks as possible. We focus on the task outsourcing problem among multiple ESs and multiple TEs. A bidding mechanism is utilized to describe the serving relationship between ESs and TEs, where the two parties are assigned as sellers and buyers. The computing resources of ESs are regarded as commodities.

Simulations results demonstrate that the algorithm increases the ESs' profit and reduces the peak load by shifting the load demand to off-peak periods. Meanwhile, the TEs' payoff are also increased by participating in game process. As for future research, we will focus on the computing offloading of ESs in a three-tier IoT MEC networks.

Acknowledgments

This work is supported by National Natural Science Foundation of China (No.61872129).

References

[1] S. Deng, Z. Xiang, J. Taheri, K. A. Mohammad, and S. Dustdar, "Optimal application deployment in resource constrained distributed edges," *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, pp. 1–1, 2020.

[2] C. Avasalcai, C. Tsigkanos, and S. Dustdar, "Decentralized resource auctioning for latency-sensitive edge computing," in *2019 IEEE International Conference on Edge Computing (EDGE)*, 2019.

[3] N. Sharghivand, F. Derakhshan, and L. Mashayekhy, "Qos-aware matching of edge computing services to internet of things," in *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, 2019.

[4] X. Lyu, H. Tian, L. Jiang, A. Vinel, S. Maharjan, S. Gjessing, and Y. Zhang, "Selective offloading in mobile edge computing for the green internet of things," *IEEE Network*, vol. 32, no. 1, pp. 54–60, 2018.

[5] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing," in *2015 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2015, pp. 1–6.

[6] D. Grosu and A. T. Chronopoulos, "Noncooperative load balancing in distributed systems," *Journal of Parallel & Distributed Computing*, vol. 65, no. 9, pp. 1022–1034, 2005.

[7] Y. Jiang, "A survey of task allocation and load balancing in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 585–599, 2016.

[8] S. Yan, M. Peng, and X. Cao, "A game theory approach for joint access selection and resource allocation in uav assisted iot communication networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1663–1674, 2018.

[9] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2016.

[10] C. Li, W. Chen, J. Tang, and Y. Luo, "Radio and computing resource allocation with energy harvesting devices in mobile edge computing environment," *Computer Communications*, 2019.

[11] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7944–7956, 2019.

[12] Y. Gao, Y. Cui, X. Wang, and Z. Liu, "Optimal resource allocation for scalable mobile edge computing," *IEEE Communications Letters*, 2019.

[13] C. Avasalcai and S. Dustdar, "Latency-aware distributed resource provisioning for deploying iot applications at the edge of the network," in *Future of Information and Communication Conference*. Springer, 2019, pp. 377–391.

[14] H. Zhang, F. Guo, H. Ji, and C. Zhu, "Combinational auction-based service provider selection in mobile edge computing networks," *IEEE Access*, vol. 5, pp. 13 455–13 464, 2017.

[15] B. Gu and Z. Zhou, "Task offloading in vehicular mobile edge computing: A matching-theoretic framework," *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 100–106, 2019.

[16] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, "Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1204–1215, 2017.

[17] M. Liu and Y. Liu, "Price-based distributed offloading for mobile-edge computing with computation capacity constraints," *IEEE Wireless Communications Letters*, vol. 7, no. 3, pp. 420–423, 2017.

[18] Q. Tang, R. Xie, T. Huang, and Y. Liu, "Jointly caching and computation resource allocation for mobile edge networks," *IET Networks*, vol. 8, no. 5, pp. 329–338, 2019.

[19] M. M. Jalali and A. Kazemi, "Demand side management in a smart grid with multiple electricity suppliers," *Energy*, vol. 81, pp. 766–776, 2015.

[20] P. Samadi, A.-H. Mohsenian-Rad, R. Schober, V. W. Wong, and J. Jatskevich, "Optimal real-time pricing algorithm based on utility maximization for smart grid," in *2010 First IEEE International Conference on Smart Grid Communications*. IEEE, 2010, pp. 415–420.

[21] F. Kamyab, M. Amini, S. Sheykhha, M. Hasanpour, and M. M. Jalali, "Demand response program in smart grid using supply function bidding mechanism," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1277–1284, 2015.

[22] R. Baldick, R. Grant, and E. Kahn, "Theory and application of linear supply function equilibrium in electricity markets," *Journal of regulatory economics*, vol. 25, no. 2, pp. 143–167, 2004.

[23] H. Chen, K. Wong, C. Chung, and D. Nguyen, "A coevolutionary approach to analyzing supply function equilibrium model," *IEEE Transactions on Power Systems*, vol. 21, no. 3, pp. 1019–1028, 2006.

[24] R. Johari and J. N. Tsitsiklis, "Parameterized supply function bidding: equilibrium and welfare," *Mathematics of Operations Research*, 2006.

[25] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica*, vol. 33, no. 3, pp. 520–534, 1965.

Zheng Xiao received the Ph.D. in Computer Science from Fudan University, China, in 2009, and B.S. in Communication Engineering from Hunan University in 2003. He is currently an associate professor in College of Information Science and Engineering of Hunan University. His major research interests include distributed artificial intelligence, high performance computing, parallel and distributed systems, intelligent information processing and collaborative optimization. He has published over 30 journal articles and conference papers. He is currently an associate editor of IEEE Access. He is a member of IEEE and CCF.

Dan He received the B.S. degree in computer science and technology from Jiangxi Normal University, Nanchang, China, in 2018. She is currently working toward the M.S. degree at the College of Information Science and Engineering, Hunan University, Changsha, China. Her research interests focus on high performance computing, modeling and resource scheduling in cloud computing systems, big data pricing and game theory.

Yu Chen is currently working toward the M.S. degree at the Collage of Information Science and Engineering, Hunan University, China. His research interests focus on machine learning and natural language processing.

Anthony Theodore Chronopoulos obtained a Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign in 1987. He is a full Professor at the Department of Computer Science, University of Texas, San Antonio, USA and a visiting professor, Department of Computer Engineering & Informatics, University of Patras, Greece. He is the author of 83 journal and 73 peer-reviewed conference proceedings publications in the areas of Distributed and Parallel Computing, Grid and Cloud Computing, Scientific Computing, Computer Networks, Computational Intelligence. He is a Fellow of the Institution of Engineering and Technology (FIET), ACM Senior member, *IEEE* Senior member.

Schahram Dustdar is Full Professor of Computer Science (Informatics) with a focus on Internet Technologies heading the Distributed Systems Group at the TU Wien. He is Chairman of the Informatics Section of the Academia Europaea (since December 9, 2016). He is elevated to IEEE Fellow (since January 2016). From 2004-2010 he was Honorary Professor of Information Systems at the Department of Computing Science at the University of Groningen (RuG), The Netherlands. From December 2016 until January 2017 he was a Visiting Professor at the University of Sevilla, Spain and from January until June 2017 he was a Visiting Professor at UC Berkeley, USA. He is a member of the IEEE Conference Activities Committee (CAC) (since 2016), of the Section Committee of Informatics of the Academia Europaea (since 2015), a member of the Academia Europaea: The Academy of Europe, Informatics Section (since 2013). He is recipient of the ACM Distinguished Scientist award (2009) and the IBM Faculty Award (2012). He is an Associate Editor of IEEE Transactions on Services Computing, ACM Transactions on the Web, and ACM Transactions on Internet Technology and on the editorial board of IEEE Internet Computing. He is the Editor-in-Chief of Computing (an SCI-ranked journal of Springer).

Jiayi Du received his Ph.D., M.S. and B.S. in computer science from Hunan University, China, in 2015, 2010 and 2004. He is currently an assistant professor in Central South University of Forest and Technology, China. His research interest includes modeling and scheduling for parallel and distributed computing systems, embedded system computing, cloud computing, parallel system reliability, and parallel algorithms.