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

Zhao Tong, *Member, IEEE*, Xin Deng, Zheng Xiao, *Member, IEEE*, Dan He, Anthony Theodore Chronopoulos, *Life Senior Member, IEEE*, and Schahram Dustdar, *Fellow, 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 servers' resource allocation has been widely studied. The limited computational resources of edge servers (ESs) cannot meet the different demands of terminal entities (TEs). This makes it a challenge to efficiently schedule computational tasks on ESs. In this paper, we consider a MEC resource transaction market with multiple ESs and multiple TEs, which are interdependent and mutually influence each other. This paper aims to investigate the dynamic tasks allocation problem between TEs and ESs and to meet the optimal benefits for both parties in MEC system. However, this many-to-many interaction requires resolving several problems, including task allocation, TEs' selection on ESs and conflicting interests of both parties. A bilateral game framework is applied to tackle the tasks allocation problem by modeling the problem as two noncooperative games: the supplier and customer side games. 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. Our simulation results have demonstrated the superior performance of *DTOA* in increasing the ESs' profit and TEs' payoffs, as well as flattening the peak and off-peak loads.**

***Index Terms*—Bidding mechanism, Internet of Things (IoT), multi-access edge computing (MEC), Nash equilibrium, noncooperative game, task outsourcing.**

Manuscript received 27 February 2023; revised 25 May 2023; accepted 15 July 2023. Date of publication 18 July 2023; date of current version 7 February 2024. This work was supported by the Program of National Natural Science Foundation of China (Grant No. 62072174, 61502165, 61872129), Distinguished Youth Science Foundation of Hunan Province, China, (Grant No. 2023JJ10030), National Natural Science Foundation of Hunan Province, China, (Grant No. 2022JJ40278, 2020JJ5370), Scientific Research Fund of Hunan Provincial Education Department, China, (Grant No. 22A0026). The associate editor coordinating the review of this article and approving it for publication was Z. Zhu. *(Corresponding author: Zheng Xiao.)*

Zhao Tong and Xin Deng are with the College of Information Science and Engineering, Hunan Normal University, Hunan 410081, China (e-mail: tongzhao@hunnu.edu.cn; 202120293782@hunnu.edu.cn).

Zheng Xiao and Dan He are with the College of Computer Science and Electronic Engineering, Hunan University, Hunan 410082, China (e-mail: zxiao@hnu.edu.cn; danhe@hnu.edu.cn).

Anthony Theodore Chronopoulos is with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249 USA, and also with the Department of Computer Engineering Informatics, University of Patras, 26500 Patras, Greece (e-mail: antony.tc@gmail.com).

Schahram Dustdar is with the Distributed Systems Group, Vienna University of Technology, 1040 Vienna, Austria (e-mail: dustdar@dsg.tuwien.ac.at).

Digital Object Identifier 10.1109/TNSM.2023.3296676

I. 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. Fig. 1 describes a MEC communication network in IoT, where edge servers (ESs) are deployed densely near TEs. In the MEC paradigm, ESs are placed at the edge of the network so that computing services can be deployed on them for fast execution [1]. Since with the limited computation capabilities and battery lives, TEs are prone to offload computationally intensive tasks (e.g., program execution) onto ESs (e.g., 4G/5G base stations), even though TEs will be charged for the computing service. For ESs, they are expected to provide computing services in parallel, which would accelerate the speed of task processing and alleviate offloading delays.

However, to take full advantages of the computational resources of ESs, the tasks of TEs need to be appropriately allocated [2]. Task outsourcing has been employed as an efficient paradigm that can accommodate as many on-demand tasks as possible. Its principle is to offload the TE's tasks to the ESs appropriately according to the load of the ES at each time slot. Specifically, TEs can choose different ESs based on their real-time and cost requirements. ESs need to accomplish more tasks while meeting the TEs' time requirements. One of the main challenges of task outsourcing is to consider the interests of both parties, i.e., ESs and TEs. On the one hand, ESs strive to attract **more** TEs to use their computing resources and may raise the prices. On the other hand, a rational TE will figure out an optimal task allocation strategy that can get sufficient computing resources from ESs at a **lower** cost. There is a conflict of interest between ESs and TEs. Therefore, the problem about how to satisfy the interests of ESs and TEs remains to be solved. Game theory can be employed as an effective tool to model the interests of two or more conflicting individuals in a trading market and also to model load balancing in distributed systems [3], [4]. It is

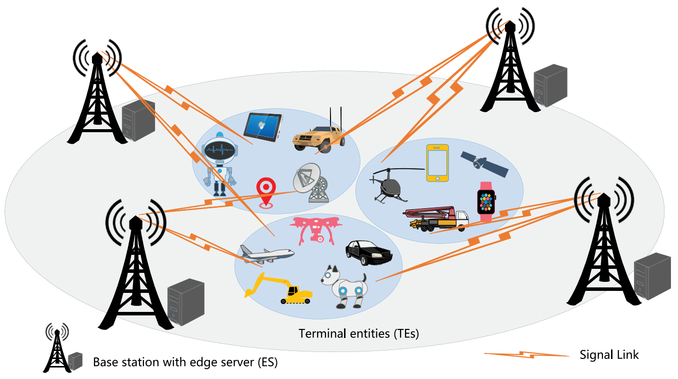

Fig. 1. The MEC network scenario in IoT.

proved that the Nash equilibrium solution offers a Quality of Service guarantee for the TEs [5].

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' wireless resources. They are interdependent and mutually influence each other. The scheduler, as a key element of task outsourcing, is the third party in the model that collects all the information (pricing and demand profiles) from ESs and TEs. The information is fully integrated and the ESs and TEs are precisely matched with each other based on demand, so that the solution of the model can be globally optimal. The interactions of ESs and TEs are illustrated in Fig. 1, below. Several base stations with ESs also compete with each other to win more TEs. The ESs can communicate with the TEs via a 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. In general, the computational resources of an ES are limited, but are relatively sufficient for the requests of a TE. One ES can serve a certain number of TEs. Demand submitted by TEs is usually responded by ESs immediately. Therefore, on this basis, it is assumed that the computational resources of the ESs are sufficient for a limited number (less than 2000) of TEs [6], [7].

In dynamic games, there are many different models that seek Nash equilibrium solutions in different ways. In this paper, considering the complex resource supply and demand relationships and interaction processes between multi-ES (suppliers) and multi-TE (customers), we apply a bilateral game framework between ESs and TEs to model the task outsourcing problem as two non-cooperative games. These two games are interrelated and play out 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.

In a specific MEC scenario, the MEC system has limited resources for computing, thus the computing and communication load pressure of edge base stations is relatively high. Thus, effective resource allocation is particularly important. We propose a new effective scheme to allocate computing resources which makes the resource utilization improved in the entire MEC systems. 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:

- The bilateral game framework is applied to model the market transactions of computing resources between multiple TEs and multiple ESs in the MEC system to maximize the profit of both TEs and ESs in the system.
- It is the first work that applies the supply function bidding mechanism to the MEC application scenario. Each ES first submits its bid to reveal the available capacity "supplied" to the market, and then the TEs determine the optimal demand profile based on the ESs' bids.
- The existence of Nash equilibrium is proved theoretically. Then, the distributed task outsourcing algorithm (***DTOA***) is proposed to solve the optimal demand and bidding strategies such that all devices converge to the Nash equilibrium.
- Simulation results demonstrate the superiority of ***DTOA*** in improving resource utilization. The ***DTOA*** not only achieves the maximization of bilateral interests, but also reduces peak loads by shifting load demand to off-peak periods.

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

II. Related Work

In recent years, significant attention has been devoted to task offloading and resource scheduling in MEC networks [8], [9], [10], [11], [12]. Hu and Li [8] utilized a mixed- integer nonlinear method to obtain a request strategy of shorter response delay. To achieve minimal power consumption on average of time and guarantee stability in request queue in a multi-tiered fog computing system, Gao et al. [9] modeled dynamic offloading using a stochastic network optimization approach. But there were inevitable prediction errors in the results. To ensure the requirement on task processing delay, a partially observable offloading approach was proposed by Xie et al. [10]. This approach meanwhile optimizes energy overhead for the devices. However, relatively high time complexity exists in this scheme compared to our work, however. To balance the response latency and energy consumption of fog devices running multiple applications, Jiang et al. [11] considered an end-to-end runtime scheduler. Simulation results of a real platform demonstrate the effectiveness of the framework. For the problem of resource allocation in vehicular networks, a new software-defined networking architecture was proposed by Goudarzi et al. [12]. It obtains elastic processing power for dynamic route calculation and real-time vehicle monitoring. However, the energy model is weak. The above proposed approaches do not consider either the simultaneous optimization of the benefits of TEs and ESs or the matching problem between TEs and ESs.

In addition to optimization goals such as reducing task latency and energy consumption similar to the above work, much of the work was dedicated to optimizing the profitability of TEs and ESs. Apostolopoulos et al. [13] proposed a novel method for determining the optimal data offloading amount for each user in a multiple MEC server environment, considering risk-seeking and loss-aversion of users. The non-cooperative game between users formulated by the authors achieves a Nash equilibrium. When using the blockchain in the MEC system, to maximize the average profit of all miners, Du et al. [14] utilized the asynchronous advantage actor-critic deep reinforcement learning algorithm for resource allocation. The proposed algorithm has a low complexity. The paper does not consider the profitability of the server, which is part of the economics. Yuan et al. [15] believed that TEs would leave some time for local pre-processing before requesting offloading services. A computational offloading strategy based on a queuing model was proposed to maximize the revenue of the ES. Experiments demonstrated the superiority of this scheduling system in reducing latency as well. In a single-ES and multi-TE MEC scenario, Tao et al. [16] modeled the interaction process between the two parties as a Stackelberg game. The utility of both the ES and the TEs is improved. However, multiple ESs is a more common scenario that is not considered. Many works have explored the computational offloading problem in multi-server and multi-user scenarios [17], [18]. In [19], Lee et al. proposed a distributed learning resource management mechanism for multiple ESs and multiple TEs. Federated learning was considered to achieve information security sharing. They employed a Stackelberg game to study the economic benefits under this mechanism. The benefits of all ESs and TEs are maximized when the game is balanced. Most of the above works are direct matching of ESs and TEs. However, this may only accomplish the local optimum.

Due to the imbalance between the ESs' computing resources and the TEs' demands, it is difficult to schedule appropriate computing tasks to ESs. Thus, the matching problem between multiple ESs and multiple TEs becomes a key issue. To solve the problem of data center work allocation in cloud computing systems, Kishor et al. [20] proposed a game-theoretic solution that achieves latency and energy efficient load balancing. Zhang et al. [21] modeled the matching relationship between ESs and TEs as a commodity trading by applying a multi-round sealed sequential combinational auction mechanism. In [22], the authors studied task offloading in vehicular MEC environments and modeled 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. In the offloading scenario with multiple users and multiple computing nodes, Wu et al. [23] proposed a decentralized offloading strategy based on energy-efficient bilateral matching. This method outperforms the benchmark scheme in terms of user fairness. A three-tier IoT fog network was proposed in [24], 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.

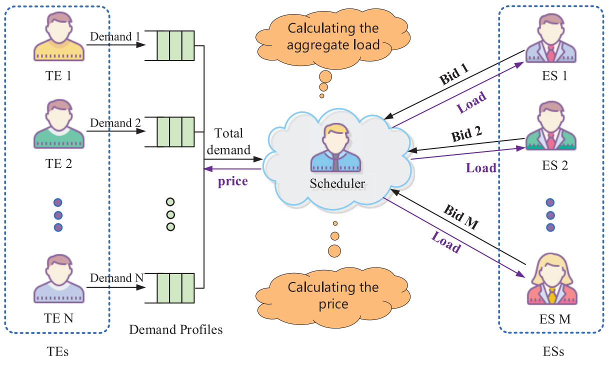

Fig. 2. Diagram of a resources transaction market.

Furthermore, authors in [25], [26] adopted a price-based mechanism to design efficient resource allocation in a MEC network. For example, [25] 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 the cost. The work in [26] 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 works above, we observe that the existing resource allocation and matching problems in MEC generally involve edge nodes and clients using resources from an edge node. Against this backdrop, our paper tries to balance the objectives of both ESs and TEs. In this paper, we also adopt a price-based supply bidding mechanism to solve the resource allocation problem.

III. System Model

A. 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 and TEs act as customers who purchase resources from ESs. It is assumed that ESs have sufficient computational resources to handle the TEs' submission demands. The scheduler serves as a third-party agency outsourcing TEs' tasks to ESs. For ease of reading, the main symbols used in this paper are listed in Table II.

TEs submit their demand profiles to the scheduler via a communication network. Additionally, ESs compete with each

TABLE I
OVERVIEW OF RECENT RESEARCHES

Ref.	Technique	Objectives	Architecture	Advantage	defect
[8]	Quasi-convex technique	Energy, delay	Ultra-dense EC	Low energy consumption, Optimal response rate	Not practical
[9]	Stochastic optimization	Time-average power consumptions	Fog	Exploring predictive offloading in fog computing, reduceing latency	Exist prediction errors
[10]	Markov Decision Process, deep recurrent Q-network	Energy	Fog	Ensuring task processing latency	High time complexity
[11]	Energy-aware cloud fog offloading	Energy, run-time network bandwidth, QoS	Fog	Saving considerable energy, adapt to dynamic changes in the network	Ignoring privacy assessments
[12]	Multi-agent RL-based approach	Reallocation overhead, delay	EC	Novel software-defined edge framework, Low time complexity	Weak energy model
[13]	non-cooperative game, convex optimization	probability	MEC	Considering risk-seeking and loss-aversion	No privacy evaluation
[14]	Blockchain, A3C deep reinforcement learning	Rational total profit of all miners	MEC	High convergence speed, Low time complexity	Ignoring the benefits of the MEC server
[15]	Game theory, queue	Network-wide revenue	EC	Considering local preprocessing	Only one ES
[16]	Stackelberg game, differential evolution algorithm	Profit of ES, utility of TEs	MEC	Proving the existence of Nash equilibrium, limite computing resources	No analyse of complexity, one ES
[17]	Potential game	Energy, latency	MEC	Minimizing the overhead of each TE	Ignoring the TEs' latency tolerance
[18]	Game theory	Multi-TE offload in multichannel wireless interference environments	MCC	Suitable for large-scale TEs	Missing comparison algorithms
[19]	Stackelberg game, federal Learning	Utility of ESs and TEs	EC	Ensuring privacy, considering multi-ESs and distributed learning regime	Inconsistent training accuracy
[20]	Nash bargaining game	Response time, energy	CC	Guaranteeing fairness to the end-users, comparing with other approaches	Ignoring the dynamic properties of the system
[21]	Auction theory	QoS, ES and TE matching	MEC	Multi-ES and Multi-TE	Ignoring the cost of ESs
[22]	Matching theory	Delay	Vehicular MEC	Considering mobility constraints, acceptable convergence speed	No privacy evaluation
[23]	Bilateral matching algorithm	Revenue, energy	Dispersed computing	Improving user fairness, saving user energy costs	Not real (private devices are reluctant to calculate for free)
[24]	Stackelberg game, matching theory	Utility of all participants	Fog	Reaching optimal utilities for all devices	Weak energy model, ignoring complex analysis
[25]	Stackelberg game	Revenue, cost, delay	MEC	Considering limited edge computing resources	Ignoring edge cost model
[26]	Stackelberg game, Newton–Raphson method	Revenue, QoS	MEC	Proving the uniqueness of Nash equilibrium	Not real (a single ES)

other for acquiring more TEs, and submit bids based on strategies of opponents and their own resource capacities. In fact, information about bids is private. ESs cannot obtain bids from others. The third-party scheduler needs to collect the initial bids of all ESs and achieve information sharing. Then, the scheduler calculates service price and the combined load based on the bids of ESs and the total demand of TEs. The decision of these two entities has an impact on each other. After receiving the real-time price signal, 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.

TABLE II
DEFINITIONS OF MATHEMATICAL NOTATIONS

Notation	Definition
$C_{j,t}(.)$	The cost function of the ES j at time slot t
$R_{j,t}(.)$	the revenue function of ES j at the t
L_t	TEs' total load demand at time slot t
u_i	the payoff of TE i
$f_{j,t}$	The supply function of the ES j at time slot t
$p_e(t)$	The computing service price at time slot t
$\chi_{i,1}, \cdots, \chi_{i,T}$	The shiftable demand profile of TE i $(i \in \mathcal{N})$
$p_1, \cdots, p_K$	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 time slot granularity is consistent with [27] and [28] of smart grid. The setting is reasonable since edge computing has already been more closely integrated with the smart grid. We divide one day into a set of T $(T = 24)$ time slots, denoted as $\mathcal{T} = \{1, \ldots, T\}$. The set of ESs and TEs are represented as $\mathcal{M} = \{1, \ldots, M\}$ 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 worth of investigating. We next present the model of both sides in the MEC resources transaction market.

B. 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. The proof of the truthfulness of bid price is part of auctions. While they are very important, we assume that the players are truthful. Such research will be done in future work. 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 Eq. (1) into Eq. (2), we can get

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

$f_{j,t}$ is denoted as the task load that ES j is 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

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

$C_{j,t}$ is the generation cost function of the ES j $(j \in \mathcal{M})$ at the time slot t $(t \in \mathcal{T})$. We assume that the cost function of the edge server is a monotonically increasing convex function. Initially, as the task load increases, the cost of the edge server increases at a progressively faster rate due to equipment maintenance. As the task load continues to increase, the cost increases at a slower rate. The quadratic function is consistent with the assumptions of this paper. Therefore, we take the form of the quadratic cost function of the MEC server in [29]. Polynomial cost functions are utilized to formulate the generation cost which is convex and monotonically increasing. This type of cost function is in the form of $C_{j,t}(x) = a_m x^m + \cdots + a_0, x = f_{j,t}$, where m is the polynomial function degree, with coefficients $a_m, \ldots, a_0$. Polynomial cost functions are in computational efficiency, because we can approximate them by Taylor polynomials or fit polynomials using regression model [30]. The ES j's cost function is defined as a quadratic function:

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

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

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

C. 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. $U_i(.)$ is used to show TE i's satisfaction from consuming computing resources. When modeling the utility function of TEs, we consider linear, logarithmic, exponential and quadratic functions. According to the law of market economy, the marginal benefit is decreasing, much the same as the cost function of smart grid. In [31], [32], the quadratic function, which is often used to model the decreasing utility, is chosen. Without loss of generality, we draw on the utility functions in these two papers. Quadratic utility is used to express the utility of the customer, and the class of utility functions are as follows.

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

where $x = (\chi_{i,t} + r_{i,t})$, $w_{i,t}$ and $\alpha_{i,t}, i \in \mathcal{N}$ are coefficients that reflect the dynamic changes of TE *i*'s demand in different time slots. The levels of satisfaction with the consumption of computing resources are dramatically different for different TEs and the same TE in different time slots. Therefore, these two coefficients can be used to distinguish the TEs' satisfaction with the consumption of computing resources.

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

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

where $\boldsymbol{\lambda}_t$ represent the bid profiles of all ESs at time slot *t* and $\boldsymbol{\lambda}_t = \{\boldsymbol{\lambda}_{1,t}, \ldots, \boldsymbol{\lambda}_{M,t}\}$.

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.,

$$Payoff_i = Utility_i - Payout_i. \quad (8)$$

Let u_i denote the payoff of TE *i*. L_t denotes the aggregate load demand of the ESs at time slot *t* and $L_t = \sum_{j\in\mathcal{N}}(\chi_{j,t} + r_{j,t})$. By substituting Eq. (6) and Eq. (7) into Eq. (8), we can obtain

$$u_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) = \sum_{t\in\mathcal{T}} \Big(U_i(\chi_{i,t} + r_{i,t}) - (\chi_{i,t} + r_{i,t}) p_e(\boldsymbol{\lambda}_t, L_t)\Big), \quad (9)$$

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 Eq. (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. Considering TE *i*, the optimization problem can be formulated as follows when other TEs' profiles are fixed:

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

D. 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.

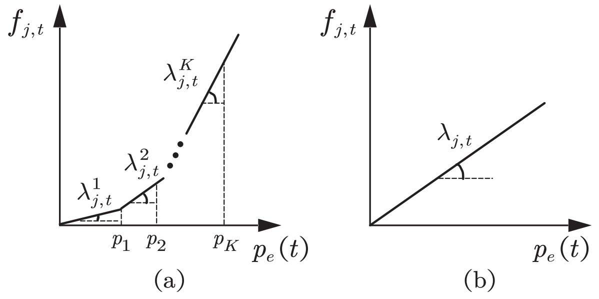

Fig. 3. (a) Piece-wise linear. (b) Affine supply functions.

We assume that the supply function $f_{j,t}$ is chosen from the family of increasing and convex price-wise linear functions of $p_e(t)$ [33]. 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 Fig. 3(a). $\lambda_{j,t}^k \ge 0$ represents the slope of the function between the break points p_{k-1} and p_k. Fig. 3(b) shows the affine supply function.

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

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

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. 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 demands of all TEs are the same as the load supplied by all ESs. Although the fluctuation in TEs' demands will drive changes in ESs' bid profiles, the demands and supplies remain balanced. The equivalence further builds up the connection between the supplier game and the customer game. Hence, it can be expressed as

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

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

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

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

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

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

In [34], the affine supply function $f_{j,t}(p_e(t), \boldsymbol{\lambda}_{j,t}) = \lambda_{j,t}^1 p_e(t)$ 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 [35]. As for Eq. (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 Eq. (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 $p_e(\boldsymbol{\lambda}_t, L_t)$. As suggested by Eq. (11), supply function $f_{j,t}$ for ES *j* can be expressed as

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

Similar to the computing service, the supply function can be represented by $f_{j,t}(\boldsymbol{\lambda}_t, L_t)$. 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 Eq. (5) and Eq. (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:

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

E. Nash Equilibrium Analysis

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

Lemma 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 $\lambda_{j,t}^* < \sum_{r\in\mathcal{M}, r\neq j} \lambda_{r,t}^*$ for all ESs.

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

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

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

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

Let

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

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 Eq. (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 Eq. (22), we can derive that

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

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

The proof for Theorem 1 is given as Appendix A.

Theorem 1: The ES's noncooperative game has a unique Nash equilibrium. Furthermore, the Nash equilibrium is the solution of the following convex optimization problem:

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

where

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

In Theorem 1, it is proved that the ES's game has a unique Nash equilibrium solution, whose strategies are determined

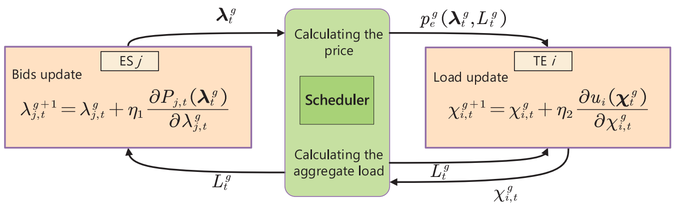
$\boldsymbol{\lambda}_t^g$
$p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$
Calculating the price
Scheduler
Calculating the aggregate load
ES j
Bids update
$\lambda_{j,t}^{g+1} = \lambda_{j,t}^g + \eta_1 \frac{\partial P_{j,t}(\boldsymbol{\lambda}_t^g)}{\partial \lambda_{j,t}^g}$
TE i
Load update
$\chi_{i,t}^{g+1} = \chi_{i,t}^g + \eta_2 \frac{\partial u_i(\boldsymbol{\chi}_t^g)}{\partial \chi_{i,t}^g}$
L_t^g
L_t^g
$\chi_{i,t}^g$

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

by the aggregate load L_t. Besides, an 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, i.e., Theorem 2. The proof of Theorem 2 can be found in Appendix B.

Theorem 2: The customers' optimization problem is a convex programming problem. In fact, the customer side game Eq. (10) is a n-person game. It has a unique pure strategy Nash equilibrium.

In the Nash equilibrium, for any given ESs' bid, no TE can increase its payoff by a unilateral change on its strategy. The Nash equilibrium can not be broken due to dynamic topology of TEs because agents are used to deal with the interactions of the TEs (and the ESs).

In the next section, a task outsourcing algorithm is developed to determine the point for both ES and TE's games.

Algorithm 1 TE's Game

```
1:  Initialization: $g = 0$.
2:  Randomly initialize TEs' demand profiles.
3:  repeat
4:      for (each time slot $t \in \mathcal{T}$) do
5:          Receive $L_t^g$ from the scheduler;
6:          Update the bid $\boldsymbol{\lambda}_t^g$ by Algorithm 2;
7:          Receive the updated $p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$ from the scheduler;
8:          for (each TE $i \in \mathcal{N}$) do
9:              $\chi_{i,t}^{g+1} = \left[\chi_{i,t}^g + \eta_2 \frac{\partial u_i(\boldsymbol{\chi}_t^g)}{\partial \chi_{i,t}^g}\right]^+$;
10:         end for
11:     end for
12:     $g := g + 1$;
13: 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: $\boldsymbol{\lambda}_t$.
1:  Initialization: Randomly initialize ESs' bid profiles for the first time.
2:  Receive $L_t$ from the scheduler.
3:  for (each ES $j \in \mathcal{M}$) do
4:      $\lambda_{j,t}^{g+1} = \left[\lambda_{j,t}^g + \eta_1 \frac{\partial P_{j,t}(\boldsymbol{\lambda}_t^g)}{\partial \lambda_{j,t}^g}\right]^+$.
5:  end for
6:  return $\boldsymbol{\lambda}_t$.
```

IV. 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 $\chi_{i,t}^g$ denote the demand profiles of TE i in iteration g at time slot t and vector $\boldsymbol{\chi}_i^g$ 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_t^g denotes the aggregate loads in iteration g at time slot t. $p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$ 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 Algorithms 1 and 2.

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

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 $\boldsymbol{\chi}_i^g$ to the broker and receives L_t^g from it. Then, the ESs will receive a signal to update their bids based on the following iterative equation:

$$\lambda_{j,t}^{g+1} = \left[\lambda_{j,t}^g + \eta_1 \frac{\partial P_{j,t}(\boldsymbol{\lambda}_t^g)}{\partial \lambda_{j,t}^g}\right]^+, \quad \forall t \in \mathcal{T}. \tag{25}$$

where η_1 is the step size. $[\cdot]^+$ in Eq. (25) 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 Eq. (15). The TEs will further be informed to update their shiftable demand profiles using a gradient boosting method:

$$\chi_{i,t}^{g+1} = \left[\chi_{i,t}^g + \eta_2 \frac{\partial u_i(\boldsymbol{\chi}_t^g)}{\partial \chi_{i,t}^g}\right]^+, \quad \forall t \in \mathcal{T}. \tag{26}$$

η_2 is the step size. $[\,\cdot\,]^+$ in Eq. (26) 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 Eq. (10) needs the updated price $p_e^g(\boldsymbol{\lambda}_t^g, L_t^g)$ and L_t^g to determine $(\partial u_i(\boldsymbol{\chi}_t^g)/\partial \chi_{i,t}^g)$. Besides, since $(\partial u_i(\boldsymbol{\chi}_t^g)/\partial \chi_{i,t}^g)$ 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

TABLE III
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)

be stopped. Otherwise, the TEs will continue computing their demand profiles based on the newly updated price and bids.

The optimization problems Eq. (18) and Eq. (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 will improve their profit.

Next, we perform the complexity analysis of Algorithms 1 and 2. The lines 1 and 2 in Algorithm 1 are the basic numerical computations, executing the operations in parallel. line 3 starts the iteration. Lines 4 to 11 are the first level of loops, which are executed *T* times. Lines 5 to 7 are mainly simple numerical computations, and the second layer of iterations begins at line 8, performing *N* times. Thus, the time complexity of Algorithm 1 is $\mathcal{O}(T \times N)$. In Algorithm 2, line 1 performs the initialization process and line 2 is the base value calculation. Lines 3 to 5 perform iterations to obtain the update strategy of ES. The loop is executed *M* times. Thus the time complexity of Algorithm 2 is $\mathcal{O}(M)$. Since *T* is set as 24, both Algorithms 1 and 2 possess near linear complexity.

V. PERFORMANCE EVALUATION

A. Simulation Experiment

In this section, we present a simulation experiment to validate our theoretical analysis. For the experimental parameters in this paper, we refer to papers on smart grid for some parameters and make some adjustments [31], [33], [37]. 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 III. 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 and 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

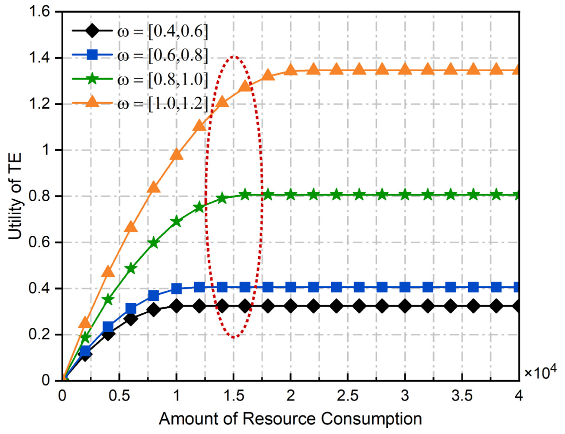

Fig. 5. Influence of parameter $\omega_{i,t}$ on TEs' utility.

$c(f_{j,t}) = a_{j,2}f_{j,t}^2 + a_{j,1}f_{j,t} + a_{j,0}$ for each ES j $(j \in \mathcal{M})$, we assume that $a_{j,2}$ is 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.

Next, the effect of the parameter $\omega_{i,t}$ on the utility of TEs and the choice of its value are discussed. The shiftable demand term $\chi_{i,t}$ of each TE is chosen from the interval [10%, 12%]. As seen from Fig. 5, as the value of the horizontal coordinate base demand $r_{i,t}$ increases, when the range of $\omega_{i,t}$ is from 0.4 to 0.8, the utility function of the TE converges quickly, but the utility value is low. The utility is higher than those of $\omega_{i,t}$ intervals [0.4, 0.6] and [0.6, 0.8], as the range of $\omega_{i,t}$ is from 0.8 to 1.0. The corresponding curve can also converge at a fast rate. When the range of $\omega_{i,t}$ is from 1.0 to 1.2, the function converges slow, and the utility value of TEs is much higher than those of the $\omega_{i,t}$ intervals [0.4, 0.6], [0.6, 0.8] and [0.8, 1.0]. This interval or a larger interval can be chosen when the utility for TEs is better. Therefore, by combining the convergence speed and the payoff of TEs, we chose the parameter $\omega_{i,t}$ from the interval [0.8,1.0] for the experiments.

B. Algorithm Convergence

The performance of our proposed ***DTOA*** is evaluated in terms of its convergence. Fig. 6 and Fig. 7 show the convergence of ESs' bids and TEs' shiftable loads at time slot 5. From Fig. 7, 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

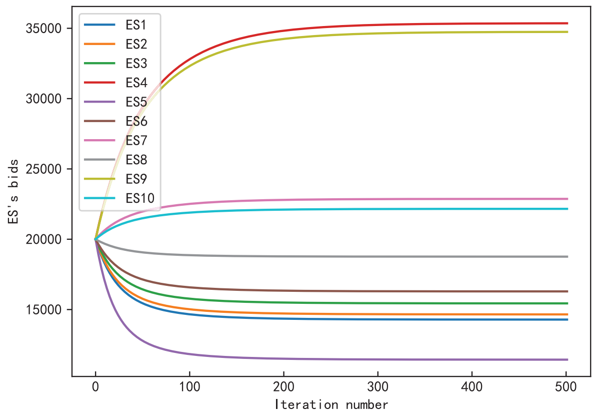

Fig. 6. Convergence of ES 1-10's bids at time slot 5.

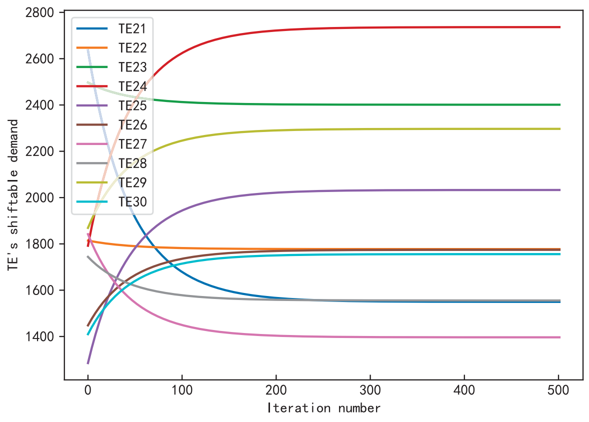

Fig. 7. Convergence of shiftable loads for TE 21-30 at time slot 5.

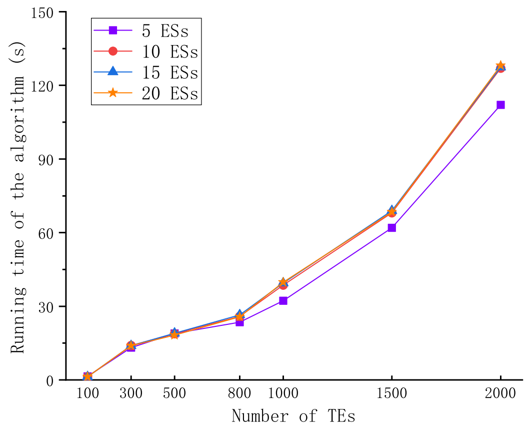

Fig. 8. Running time of algorithm for different number of TEs and ESs.

different number of TEs and ESs. As shown in Fig. 8, 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. 8, 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.

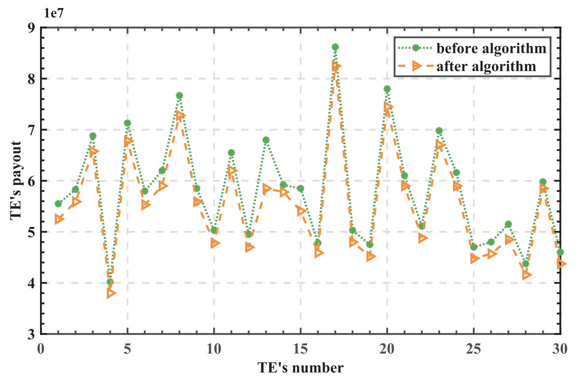

Fig. 9. Daily total payout for TE 1 to TE 30.

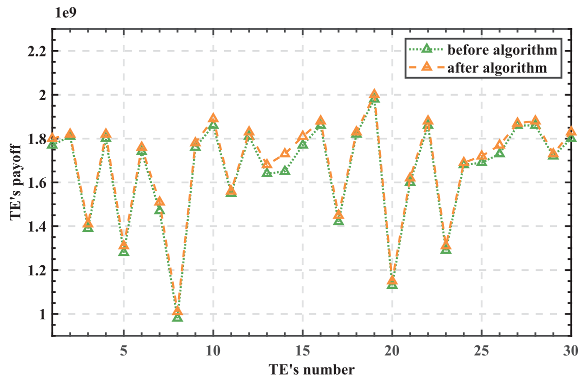

Fig. 10. Daily total payoff for TE 1 to TE 30.

C. Economic Effects of Algorithm

By participating in the ***DTOA*** scheme, the payout (see Eq. (7)) represents TE's expenditure on purchasing computing resources. For simulations, the original effects of TE's payoff, TE's expense, ES's profit, etc. in the initial state are represented by the yellow line of "before algorithm". The experimental effects after the intervention of the proposed ***DTOA*** are represented by the "after algorithm". In order to enhance the feasibility, we averaged all the data in multiple groups to make the results generalizable. Fig. 9 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. 9 shows the payouts of TEs are so huge, and even a 5% savings reduces a large expenditure. Furthermore, as shown in Fig. 10, the daily total payoff of each TE after using the algorithm has increased than before algorithm. The payoff (see Eq. (9)) is the utility of the calculation tasks minus the payout. Although the yellow line is only a little more than the green one, the improvement of the effect is also obvious due to its magnitude is large and arrives 10^9.

Fig. 11 displays the total profit of ESs 1-10 before and after algorithm. The total profit of ES is the sum of the profit of

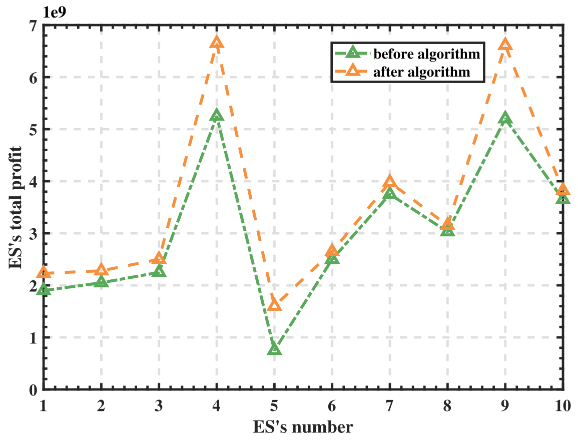

Fig. 11. Daily total profit for ES 1 to ES 10.

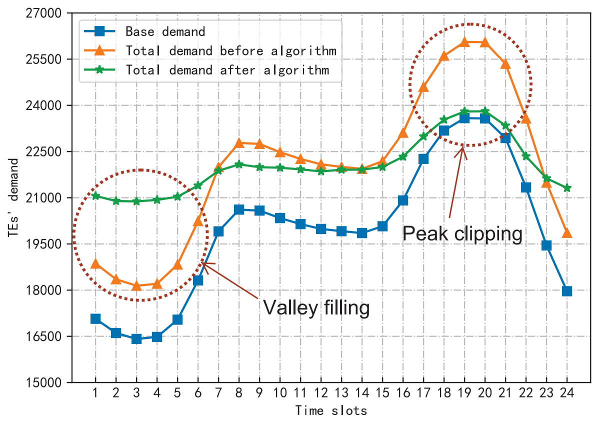

Fig. 12. Base and total demand before and after the algorithm. The peak shaving is achieved by using ***DTOA*** (the dashed circle).

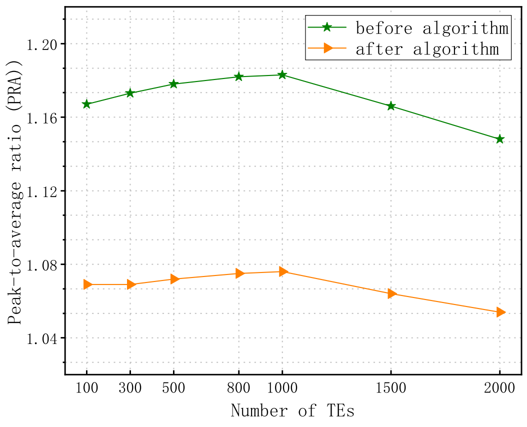

Fig. 13. Peak-to-average ratio with and without task scheduling.

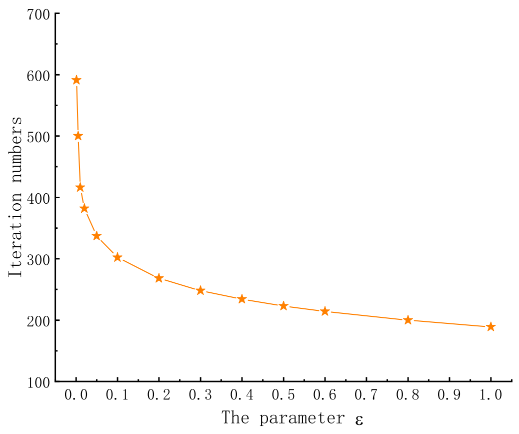

Fig. 14. The influence of the parameter ϵ on iteration numbers.

all time slots. From Fig. 11, 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.

D. Peak-Reducing Effect of Algorithm

For simulations, the initial state of TEs' demand is assumed to be load profile before algorithm. As shown in Fig. 12, 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***. The peak load demands are shifted from peak to off-peak time slots. This is because cost is a quadratic function of the demand. Even in the low-price of ESs, a small increasing in demand will result in high cost. Therefore, instead of only exploring the process ability of the low-price ESs, ***DTOA*** not only considers the diversity of the hourly task prices, but also tries to balance the peak and off-peak loads. 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. 13 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.

E. 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. 14 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.

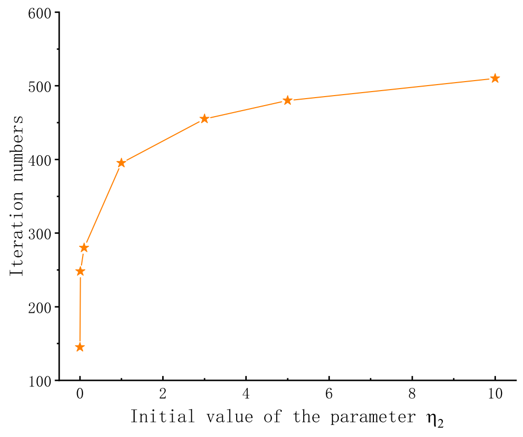

Fig. 15. The influence of the parameter η_2 on iteration numbers.

In Fig. 15, the influence of the parameter η_2 on iteration numbers is shown. Since the parameter η_2 will change in every round, as shown in Table III, 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 of the algorithm convergence is related to the setting of some parameters.

VI. 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. In future research, we will focus on the computing offloading of ESs in a three-tier IoT MEC networks. Furthermore, as the development of 5G technology, communication resources are becoming more abundant, which poses a challenge to the computational capacity of ESs. We will consider the problem of limited computational resources and design reasonable offloading algorithms to allocate tasks to the available computational resources and dynamically adjust task allocation to accommodate real-time changes in the computational resources by predicting and monitoring resources on ESs.

Appendix A
Proof of Theory 1

Proof: According to lemma 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 Eq. (23) is denoted as F. Thus, we have

$$F = \sum_{j\in\mathcal{M}} -\Psi_j\left(f_{j,t}\right) + \phi\left(\sum_{j\in\mathcal{M}} f_{j,t} - L_t\right), \tag{27}$$

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

$$\left(\frac{\partial F}{\partial f^*_{j,t}}\right)\left(f_{j,t} - f^*_{j,t}\right) \le 0, \quad \forall j \in \mathcal{M}, \tag{28}$$

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

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

$$\frac{\partial F}{\partial f^*_{j,t}} = \phi^* - \left(\frac{L_t - f^*_{j,t}}{L_t - 2f^*_{j,t}}\right) C'_j\left(f^*_{j,t}\right). \tag{29}$$

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

$$\left(\frac{\partial P_{j,t}}{\partial \lambda_{j,t}}\right)\left(\lambda_{j,t} - \lambda^*_{j,t}\right) \le 0, \quad \forall j \in \mathcal{M}. \tag{30}$$

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

$$\frac{\partial P_{j,t}}{\partial \lambda_{j,t}} = p_e(\boldsymbol{\lambda}_t, L_t) - \frac{L_t - f^*_{j,t}}{L_t - 2f^*_{j,t}} C'_j\left(f^*_{j,t}\right). \tag{31}$$

By substituting Eq. (31) into Eq. (30), we can write the optimality condition for Nash equilibrium as follows

$$\left(p_e(\boldsymbol{\lambda}_t, L_t) - \frac{L_t - f^*_{j,t}}{L_t - 2f^*_{j,t}} C'_j\left(f^*_{j,t}\right)\right)\left(\lambda_{j,t} - \lambda^*_{j,t}\right) \le 0. \tag{32}$$

From Eq. (28) and Eq. (32), 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 Eq. (28) is equivalent to Eq.32. Therefore, the existence and uniqueness of the Nash equilibrium is equivalent to proving the existence and uniqueness of the optimal point of problem Eq. (23). ■

Appendix B
Proof of Theory 2

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

$$\sum_{t\in\mathcal{T}} U_i\left(\chi_{i,t} + r_{i,t}\right) - \sum_{t\in\mathcal{T}}\left(\frac{\left(\chi_{i,t}+r_{i,t}\right)^2 + \sum_{j\in\mathcal{N},j\ne i}\left(\chi_{j,t}+r_{j,t}\right)}{\sum_{r\in\mathcal{M}}\lambda_{r,t}}\right). \tag{33}$$

Let $k = 1/(\sum_{r\in\mathcal{M}}\lambda_{r,t}), k > 0$. For simplicity, we denote the right part of Eq. (33) as follows:

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

We have

$$\begin{aligned}\nabla_{\boldsymbol{\chi}_i} h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) &= \left[\frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,t}}\right]_{t=1}^{T} \\ &= \left(\frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,1}}, \ldots, \frac{\partial h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i})}{\partial \chi_{i,T}}\right) \\ &= 2k\left[(\chi_{i,t} + r_{i,t}) + \sum_{j \in \mathcal{N}, j \neq i} (\chi_{j,t} + r_{j,t})\right]_{t=1}^{T} \end{aligned} \tag{34}$$

and the Hessian matrix is as follows:

$$\nabla^2_{\boldsymbol{\chi}_i} h_i(\boldsymbol{\chi}_i, \boldsymbol{\chi}_{-i}) = \begin{pmatrix} 2k & 2k & \cdots & 2k \\ 2k & 2k & \cdots & 2k \\ \vdots & \vdots & \ddots & \vdots \\ 2k & 2k & \cdots & 2k \end{pmatrix}_{N \times T}. \tag{35}$$

This further leads to

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

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 Eq. (9) of each TE i $(\forall i \in \mathcal{N})$ is strictly concave. So the objective function in Eq. (33) is concave. Hence, Eq. (10) is a convex optimization problem. Meanwhile, since the constraints of Eq. (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 [38, Th. 1]. Analogously to [38, Th. 3], for a concave N-person game, there exists a unique equilibrium solution. Therefore, the theorem is proved. ■

References

[1] S. Deng et al., "Optimal application deployment in resource constrained distributed edges," *IEEE Trans. Mobile Comput.*, vol. 20, no. 5, pp. 1907–1923, May 2021.

[2] C. Avasalcai, C. Tsigkanos, and S. Dustdar, "Decentralized resource auctioning for latency-sensitive edge computing," in *Proc. IEEE Int. Conf. EDGE Comput.*, 2019, pp. 72–76.

[3] D. Grosu and A. T. Chronopoulos, "Noncooperative load balancing in distributed systems," *J. Parallel Distrib. Comput.*, vol. 65, no. 9, pp. 1022–1034, 2005.

[4] J. Xue, Q. Hu, Y. An, and L. Wang, "Joint task offloading and resource allocation in vehicle-assisted multi-access edge computing," *Comput. Commun.*, vol. 177, pp. 77–85, Sep. 2021.

[5] S. Penmatsa and A. T. Chronopoulos, "Game-theoretic static load balancing for distributed systems," *J. Parallel Distrib. Comput.*, vol. 71, no. 4, pp. 537–555, 2011.

[6] Z. Hu et al., "An efficient Online computation offloading approach for large-scale mobile edge computing via deep reinforcement learning," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 669–683, Mar./Apr. 2021.

[7] X. Chen, M. Li, H. Zhong, Y. Ma, and C.-H. Hsu, "DNNOff: Offloading DNN-based intelligent IoT applications in mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2820–2829, Apr. 2022.

[8] S. Hu and G. Li, "Dynamic request scheduling optimization in mobile edge computing for IoT applications," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1426–1437, Feb. 2020.

[9] X. Gao, X. Huang, S. Bian, Z. Shao, and Y. Yang, "PORA: Predictive offloading and resource allocation in dynamic fog computing systems," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 72–87, Jan. 2020.

[10] R. Xie, Q. Tang, C. Liang, F. R. Yu, and T. Huang, "Dynamic computation offloading in IoT fog systems with imperfect channel state information: A POMDP approach," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 345–356, Jan. 2021.

[11] Y.-L. Jiang, Y.-S. Chen, S.-W. Yang, and C.-H. Wu, "Energy-efficient task offloading for time-sensitive applications in fog computing," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2930–2941, Sep. 2019.

[12] S. Goudarzi, M. H. Anisi, H. Ahmadi, and L. Musavian, "Dynamic resource allocation model for distribution operations using SDN," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 976–988, Jan. 2021.

[13] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, "Risk-aware data offloading in multi-server multi-access edge computing environment," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1405–1418, Jun. 2020.

[14] J. Du et al., "Resource pricing and allocation in MEC enabled blockchain systems: An A3C deep reinforcement learning approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 1, pp. 33–44, Jan./Feb. 2021.

[15] Y. Yuan, C. Yi, B. Chen, Y. Shi, and J. Cai, "A computation offloading game for jointly managing local pre-processing time-length and priority selection in edge computing," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 9868–9883, Sep. 2022, doi: 10.1109/TVT.2022.3177432.

[16] M. Tao, K. Ota, M. Dong, and H. Yuan, "Stackelberg game-based pricing and offloading in mobile edge computing," *IEEE Wireless Commun. Lett.*, vol. 11, no. 5, pp. 883–887, May 2022.

[17] L. Yang, H. Zhang, X. Li, H. Ji, and V. C. M. Leung, "A distributed computation offloading strategy in small-cell networks integrated with mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2762–2773, Dec. 2018.

[18] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[19] J. Lee, D. Kim, and D. Niyato, "Market analysis of distributed learning resource management for Internet of Things: A game-theoretic approach," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8430–8439, Sep. 2020.

[20] A. Kishor, R. Niyogi, A. Chronopoulos, and A. Zomaya, "Latency and energy-aware load balancing in cloud data centers: A bargaining game based approach," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 927–941, Jan.-Mar. 2023, doi: 10.1109/TCC.2021.3121481.

[21] 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. 13455–13464, 2017.

[22] B. Gu and Z. Zhou, "Task offloading in vehicular mobile edge computing: A matching-theoretic framework," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 100–106, Sep. 2019.

[23] H. Wu et al., "Resolving multitask competition for constrained resources in dispersed computing: A bilateral matching game," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 16972–16983, Dec. 2021.

[24] 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 Things J.*, vol. 4, no. 5, pp. 1204–1215, Oct. 2017.

[25] M. Liu and Y. Liu, "Price-based distributed offloading for mobile-edge computing with computation capacity constraints," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 420–423, Jun. 2018.

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

[27] F. V. Cerna, M. Pourakbari-Kasmaei, R. G. Barros, E. Naderi, M. Lehtonen, and J. Contreras, "Optimal operating scheme of neighborhood energy storage communities to improve power grid performance in smart cities," *Appl. Energy*, vol. 331, Feb. 2023, Art. no. 120411.

[28] M. Rahimiyan and L. Baringo, "Strategic bidding for a virtual power plant in the day-ahead and real-time markets: A price-taker robust optimization approach," *IEEE Trans. Power Syst.*, vol. 31, no. 4, pp. 2676–2687, Jul. 2016.

[29] T. Wang, Y. Lu, J. Wang, H.-N. Dai, X. Zheng, and W. Jia, "EIHDP: Edge-intelligent hierarchical dynamic pricing based on cloud-edge-client collaboration for IoT systems," *IEEE Trans. Comput.*, vol. 70, no. 8, pp. 1285–1298, Aug. 2021.

[30] R. A. Berk, *Regression Analysis: A Constructive Critique*, vol. 11. Thousand Oaks, CA, USA: SAGE, 2004.

[31] P. Samadi, H. Mohsenian-Rad, R. Schober, and V. W. Wong, "Advanced demand side management for the future smart grid using mechanism design," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1170–1180, Sep. 2012.

[32] R. Deng, Z. Yang, J. Chen, N. R. Asr, and M.-Y. Chow, "Residential energy consumption scheduling: A coupled-constraint game approach," *IEEE Trans. Smart Grid*, vol. 5, no. 3, pp. 1340–1350, May 2014.

[33] F. Kamyab, M. Amini, S. Sheykhha, M. Hasanpour, and M. M. Jalali, "Demand response program in smart grid using supply function bidding mechanism," *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1277–1284, May 2016.

[34] R. Baldick, R. Grant, and E. Kahn, "Theory and application of linear supply function equilibrium in electricity markets," *J. Regulat. Econ.*, vol. 25, no. 2, pp. 143–167, 2004.

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

[36] R. Johari and J. N. Tsitsiklis, "Parameterized supply function bidding: Equilibrium and welfare," *Math. Oper. Res.*, vol. 59, no. 5, pp. 1079–1089, 2006.

[37] 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 *Proc. 1st IEEE Int. Conf. Smart Grid Commun.*, 2010, pp. 415–420.

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

Zhao Tong (Member, IEEE) received the Ph.D. degree in computer science from Hunan University, Changsha, China, in 2014. He was a Visiting Scholar with Georgia State University from 2017 to 2018. He is currently an Associate Professor with the College of Information Science and Engineering, Hunan Normal University, the young backbone teacher of Hunan Province, China. He has published more than 25 research papers in international conferences and journals, such as IEEE-TPDS, *Information Sciences*, FGCS, NCA, and JPDC, PDCAT, etc. His research interests include parallel and distributed computing systems, resource management, big data, and machine learning algorithm. He is a Senior Member of the China Computer Federation.

Xin Deng received the B.S. degree in computer science and technology from Hengyang Normal University, Hengyang, China, in 2020. She is currently pursuing the M.S. degree with the College of Information Science and Engineering, Hunan Normal University, Changsha, China. Her research interests focus on distributed parallel computing, modeling and resource pricing and allocation in mobile-edge computing systems, and game theory.

Zheng Xiao (Member, IEEE) received the B.S. degree in communication engineering from Hunan University in 2003, and the Ph.D. degree in computer science from Fudan University, China, in 2009. He is currently an Associate Professor with the College of Information Science and Engineering, Hunan University. He has published over 30 journal articles and conference papers. His major research interests include distributed artificial intelligence, high performance computing, parallel and distributed systems, intelligent information processing, and collaborative optimization. He is currently an Associate Editor of IEEE ACCESS. He is a member of CCF.

Dan He received the B.S. degree in computer science and technology from Jiangxi Normal University, Nanchang, China, in 2018. She is currently pursuing the M.S. degree with 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.

Anthony Theodore Chronopoulos (Life Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign in 1987. He is a Full Professor with the Department of Computer Science, University of Texas at San Antonio, San Antonio, USA, and a Visiting Professor with the Department of Computer Engineering and 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, and computational intelligence. He is a Fellow of the Institution of Engineering and Technology and a Senior Member of ACM.

Schahram Dustdar (Fellow, IEEE) is a Full Professor of Computer Science (Informatics) with a focus on Internet Technologies heading the Distributed Systems Group with TU Wien. From 2004 to 2010, he was a Honorary Professor of Information Systems with the Department of Computing Science, University of Groningen (RuG), The Netherlands. From December 2016 to January 2017, he was a Visiting Professor with the University of Sevilla, Spain, and from January to June 2017, he was a Visiting Professor with UC Berkeley, USA. He is the recipient of the ACM Distinguished Scientist Award in 2009 and the IBM Faculty Award in 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). He is Co-Editor-in-Chief of *ACM Transactions on Internet of Things*. He has been the Chairman of the Informatics Section of the Academia Europaea since 9 December 2016. He has been a member of the IEEE Conference Activities Committee since 2016, the Section Committee of Informatics of the Academia Europaea since 2015, and the Academia Europaea: The Academy of Europe, Informatics Section since 2013.