Explorations of Game Theory Applied in Cloud Computing

Chubo Liu, Kenli Li, and Keqin Li

1 Background and Motivation

Cloud computing has recently emerged as a paradigm for a cloud provider to host and deliver computing services to enterprises and consumers [1]. Usually, the provided services mainly refer to Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), which are all made available to the general public in a *pay-as-you-go* manner [2, 3]. In most systems, the service provider provides the architecture for users to make reservations/price bidding in advance [4, 5]. When making reservations for a cloud service or making price bidding for resource usage, multiple users and the cloud provider need to reach an agreement on the costs of the provided service and make planning to use the service/resource in the reserved time slots, which could lead to a competition for the usage of limited resources [6]. Therefore, it is important for a user to configure his/her strategies without complete information of other users, such that his/her utility is maximized.

For a cloud provider, the income (i.e., the revenue) is the service charge to users [7]. When providing services to multiple cloud users, a suitable pricing model is a significant factor that should be taken into account. The reason lies in that a proper pricing model is not just for the profit of a cloud provider, but for the appeals to more cloud users in the market to use cloud service. Specifically, if the per request charge is too high, a user may refuse to use the cloud service, and choose another

C. Liu (✉) · K. Li
College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan, China
e-mail: liuchubo@hnu.edu.cn

K. Li
Department of Computer Science, State University of New York, New Paltz, NY, USA

R. Ranjan et al. (eds.), *Handbook of Integration of Cloud Computing, Cyber Physical Systems and Internet of Things*, Scalable Computing and Communications,
https://doi.org/10.1007/978-3-030-43795-4_3

cloud provider or just finish his/her tasks locally. On the contrary, if the charge is too low, the aggregated requests may be more than enough, which could lead to low service quality (long task response time) and thus dissatisfies its cloud users.

A rational user will choose a strategy to use the service/resources that maximizes his/her own net reward, i.e., the utility obtained by choosing the cloud service minus the payment [1]. On the other hand, the utility of a user is not only determined by the importance of his/her tasks (i.e., how much benefit the user can receive by finishing the tasks), but also closely related to the urgency of the task (i.e., how quickly it can be finished). The same task, such as running an online voice recognition algorithm, is able to generate more utility for a cloud user if it can be completed within a shorter period of time in the cloud [1]. However, considering the energy saving and economic reasons, it is irrational for a cloud provider to provide enough computing resources to satisfy all requests in a time slot. Therefore, multiple cloud users have to compete for the cloud service/resources. Since the payment and time efficiency of each user are affected by decisions of other users, it is natural to analyze the behavior of such systems as strategic games [4].

In this chapter, we try to enhance services in cloud computing by considering from multiple users' perspective. Specifically, we try to improve cloud services by simultaneously optimizing multiple users' utilities which involve both time and payment. We use game theory to analyze the situation. We formulate a service reservation model and a price bidding model and regard the relationship of multiple users as a non-cooperative game. We try to obtain a Nash equilibrium to simultaneously maximize multiple users' utilities. To solve the problems, we prove the existence of Nash equilibrium and design two different approaches to obtain a Nash equilibrium for the two problems, respectively. Extensive experiments are also conducted, which verify our analyses and show the efficiencies of our methods.

2 Related Works

In many scenarios, a service provider provides the architecture for users to make reservations in advance [4–6] or bid for resource usage [8–10]. One of the most important aspects that should be taken into account by the provider is its resource allocation model referring users' charging/bidding prices, which is closely related to its profit and the appeals to market users.

Many works have been done on the pricing scheme in the literature [7, 11–15]. In [7], Cao et al. proposed a time dependent pricing scheme, i.e., the charge of a user is dependent on the service time of his/her requirement. However, we may note that the service time is not only affected by the amount of his/her own requirement, but also influenced by other factors such as the processing capacity of servers and the requirements of others. Mohsenian-Rad et al. [11] proposed a dynamic pricing scheme, in which the per price (the cost of one request or one unit of load) of a certain time slot is set as an increasing and smooth function of the aggregated requests in that time slot. That is to say, when the aggregated requests are quite much in a time slot, the users have to pay relatively high costs to complete the same amount of requests, which is an effective way to convince the users to shift their peak-time task requests. In [9], Samimi et al. focused on resource allocation in cloud that considers the benefits for both the users and providers. To address the problem, they proposed a new resource allocation model called combinatorial double auction resource allocation (CDARA), which allocates the resources according to bidding prices. In [8], Zaman and Grosu argued that combinatorial auction-based resource allocation mechanisms are especially efficient over the fixed-price mechanisms. They formulated resource allocation problem in clouds as a combinatorial auction problem and proposed two solving mechanisms, which are extensions of two existing combinatorial auction mechanisms. In [10], the authors also presented a resource allocation model using combinatorial auction mechanisms. Similar studies and models can be found in [16–19]. Similar studies and models can be found in [12–19]. However, these models are only applied to control energy consumption and different from applications in cloud services, since there is no need to consider time efficiency in them. Furthermore, almost all of them consider from the perspective of a cloud provider, which is significantly different from our multiple users' perspective.

Game theory is a field of applied mathematics that describes and analyzes scenarios with interactive decisions [20–22]. It is a formal study of conflicts and cooperation among multiple competitive users [23] and a powerful tool for the design and control of multiagent systems [24]. There has been growing interest in adopting cooperative and non-cooperative game theoretic approaches to modeling many problems [11, 25–27]. In [11], Mohsenian-Rad et al. used game theory to solve an energy consumption scheduling problem. In their work, they proved the existence of the unique Nash equilibrium solution and then proposed an algorithm to obtain it. They also analyzed the convergence of their proposed algorithm. Even though the formats for using game theory in our work, i.e., proving Nash equilibrium solution existence, proposing an algorithm, and analyzing the convergence of the proposed algorithm, are similar to [11], the formulated problem and the analysis process are entirely different. In [28], the authors used cooperative and non-cooperative game theory to analyze load balancing for distributed systems. Different from their proposed non-cooperative algorithm, we solve our problem in a distributed iterative way. In our previous work [29], we used non-cooperative game theory to address the scheduling for simple linear deteriorating jobs. For more works on game theory, the reader is referred to [15, 28, 30–32].

3 Strategy Configurations of Multiple Users Competition for Cloud Service Reservation

3.1 Model Formulation and Analyses

To begin with, we present our system model in the context of a service cloud provider, and establish some important results. In this paper, we are concerned with a market with a service cloud provider and n cloud users, who are competing for the cloud service reservation. We denote the set of users as $\mathcal{N} = \{1, \ldots, n\}$. The arrival of requests from cloud user i $(i \in \mathcal{N})$ is assumed to follow a Poisson process. The cloud provider is modeled by an M/M/m queue, serving a common pool of cloud users with m homogeneous servers. Similar to [33, 34], we assume that the request profile of each user is determined in advance for H future time slots. Each time slot can represent different timing horizons, e.g., one hour of a day.

3.1.1 Request Profile Model

We consider a user request model motivated by [12, 15], where the user i's $(i \in \mathcal{N})$ request profile over the H future time slots is formulated as

$$\boldsymbol{\lambda}_i = \left(\lambda_i^1, \ldots, \lambda_i^H\right)^T, \tag{1}$$

where λ_i^h $(i \in \mathcal{N})$ is the arrival rate of requests from user i in the hth time slot and it is subject to the constraint $\sum_{h=1}^{H} \lambda_i^h = \Lambda_i$, where Λ_i denotes user i's total requests. The arrivals in different time slots of the requests are assumed to follow a Poisson process. The individual strategy set of user i can be expressed as

$$Q_i = \left\{ \boldsymbol{\lambda}_i \middle| \sum_{h=1}^{H} \lambda_i^h = \Lambda_i \text{ and } \lambda_i^h \geq 0, \forall h \in \mathcal{H} \right\}, \tag{2}$$

where $\mathcal{H} = \{1, \ldots, H\}$ is the set of all H future time slots.

3.1.2 Load Billing Model

To efficiently convince the users to shift their peak-time requests and fairly charge the users for their cloud services, we adopt the instantaneous load billing scheme, which is motivated by [12, 15], where the request price (the cost of one request) of a certain time slot is set as an increasing and smooth function of the total requests in that time slot, and the users are charged based on the instantaneous request price. In this paper, we focus on a practical and specific polynomial request price model.

Specifically, the service price for one unit of workload of the hth time slot is given by

$$C\left(\lambda_{\Sigma}^{h}\right) = a\left(\lambda_{\Sigma}^{h}\right)^{2} + b, \tag{3}$$

where a and b are constants with $a, b > 0$, and λ_{Σ}^{h} is the aggregated requests from all users in time slot h, i.e., $\lambda_{\Sigma}^{h} = \sum\limits_{i=1}^{n} \lambda_{i}^{h}$.

3.1.3 Cloud Service Model

The cloud provider is modeled by an M/M/m queue, serving a common pool of multiple cloud users with m homogeneous servers. The processing capacity of each server is presented by its service rate μ_0. We denote μ as the total processing capacity of all m servers and Λ as the aggregated requests from all cloud users, respectively. Then we have $\mu = m\mu_0$, and $\Lambda = \sum\limits_{i=1}^{n} \Lambda_i$.

Let p_i be the probability that there are i service requests (waiting or being processed) and $\rho = \Lambda/\mu$ be the service utilization in the M/M/m queuing system. With reference to [7, 35], we obtain

$$p_i = \begin{cases} \frac{1}{i!}(m\rho)^i p_0, & i < m; \\ \frac{m^m \rho^i}{m!} p_0, & i \geq m; \end{cases} \tag{4}$$

where

$$p_0 = \left\{ \sum_{k=0}^{m-1} \frac{1}{k!}(m\rho)^k + \frac{1}{m!} \frac{(m\rho)^m}{1-\rho} \right\}^{-1}. \tag{5}$$

The average number of service requests (in waiting or in execution) is

$$\bar{N} = \sum_{i=0}^{\infty} k p_i = \frac{p_m}{1-\rho} = m\rho + \frac{\rho}{1-\rho} P_q, \tag{6}$$

where P_q represents the probability that the incoming requests need to wait in queue.

Applying Little's result, we get the average response time as

$$\bar{T} = \frac{\bar{N}}{\Lambda} = \frac{1}{\Lambda}\left(m\rho + \frac{\rho}{1-\rho} P_q \right). \tag{7}$$

In this paper, we assume that all the servers will likely keep busy, because if not so, some servers could be shutdown to reduce mechanical wear and energy cost. For analytical tractability, P_q is assumed to be 1. Therefore, we have

$$\bar{T} = \frac{\bar{N}}{\Lambda} = \frac{1}{\Lambda}\left(m\rho + \frac{\rho}{1-\rho}\right) = \frac{m}{\mu} + \frac{1}{\mu - \Lambda}. \tag{8}$$

Now, we focus on time slot h ($h \in \mathcal{H}$). We get that the average response time in that time slot as

$$\bar{T}^h = \frac{m}{\mu} + \frac{1}{\mu - \lambda_\Sigma^h}, \tag{9}$$

where $\lambda_\Sigma^h = \sum_{i=1}^{n} \lambda_i^h$. In this paper, we assume that $\lambda_i^h < \mu$ ($\forall h \in \mathcal{H}$), i.e., the aggregated requests in time slot h never exceeds the total capacity of all servers.

3.1.4 Architecture Model

In this subsection, we model the architecture of our proposed service mechanism, in which the cloud provider can evaluate proper charge parameters according to the aggregated requests and the cloud users can make proper decisions through the information exchange module. As shown in Fig. 1, each user i ($i \in \mathcal{N}$) is equipped with a utility function (U_i) and the request configuration ($\boldsymbol{\lambda}_i$), i.e., the service reservation strategy over H future time slots. All requests enter a queue to be processed by the cloud computing. Let $\boldsymbol{\lambda}_\Sigma$ be aggregated request vector, then we

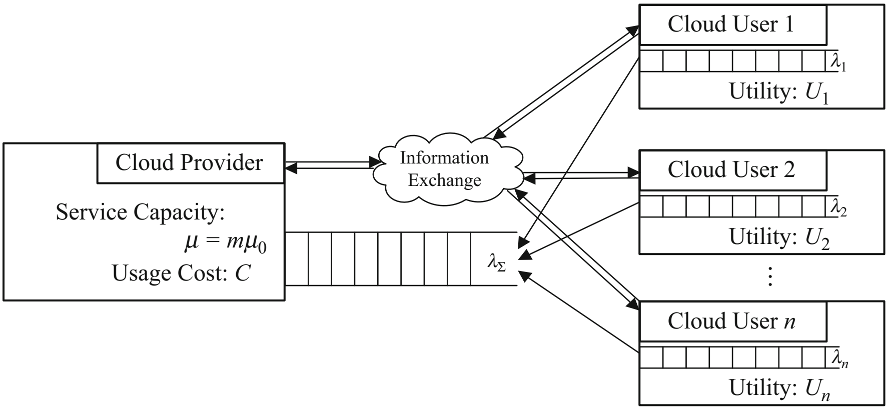

Fig. 1 Architecture overall view

have $\boldsymbol{\lambda}_\Sigma = \sum\limits_{i=1}^{n} \boldsymbol{\lambda}_i$. The cloud provider consists of m homogeneous servers with total processing rate μ, i.e., $\mu = m\mu_0$, where μ_0 is the service rate of each server, and puts some information (e.g., price parameters a and b, current aggregated request vector $\boldsymbol{\lambda}_\Sigma$) into the information exchange module. When multiple users try to make a cloud service reservation, they first get information from the exchange module, then compute proper strategies such that their own utilities are maximized and send the newly strategies to the cloud provider. The procedure is terminated until the set of remaining users, who prefer to make cloud service reservation, and their corresponding strategies are kept fixed.

3.1.5 Problem Formulation

Now, let us consider user i's ($i \in \mathcal{N}$) utility in time slot h. A rational cloud user will seek a strategy to maximize its expected net reward by finishing the tasks, i.e., the benefit obtained by choosing the cloud service minus its total payment. Since all cloud users are charged based on the instantaneous load billing and how much tasks they submit, we denote the cloud user i's payment in time slot h by P_i^h, where $P_i^h = C\left(\lambda_\Sigma^h\right)\lambda_i^h$ with $C\left(\lambda_\Sigma^h\right)$ denoting the service price for one unit of workload in time slot h. On the other hand, since a user will be more satisfied with much faster service, we also take the average response time into account. Note that time utility will be deteriorated with the delay of time slots. Hence, in this paper, we assume that the deteriorating rate of time utility is δ ($\delta > 1$). Denote $\bar{T}^h$ the average response time and T^h the time utility of user i in time slot h, respectively. Then we have $T^h = \delta^h \bar{T}^h$. More formally, the utility of user i ($i \in \mathcal{N}$) in time slot h is defined as

$$\begin{aligned} U_i^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right) &= r\lambda_i^h - P_i^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right) - w_i T^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right) \\ &= r\lambda_i^h - P_i^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right) - w_i \delta^h \bar{T}^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right), \end{aligned} \tag{10}$$

where $\boldsymbol{\lambda}_{-i}^h = \left(\lambda_1^h, \ldots, \lambda_{i-1}^h, \lambda_{i+1}^h, \ldots, \lambda_n^h\right)$ denotes the vector of all users' request profile in time slot h except that of user i, r ($r > 0$) is the benefit factor (the reward obtained by one task request), and w_i ($w_i > 0$) is the waiting cost factor, which reflects its urgency. If a user is more concerned with task completion, then the associated waiting factor w_i might be larger.

For simplicity, we use P_i^h and $\bar{T}^h$ to denote $P_i^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right)$ and $\bar{T}^h\left(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h\right)$, respectively. Following the adopted request price model, the total utility obtained by user i ($i \in \mathcal{N}$) over all H future time slots can thus be given by

$$U_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) = \sum_{h=1}^{H} U_i^h(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h) = \sum_{h=1}^{H}\left(r\lambda_i^h - P_i^h - w_i\delta^h\bar{T}^h\right), \tag{11}$$

where $\boldsymbol{\lambda}_{-i} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_{i-1}, \boldsymbol{\lambda}_{i+1}, \ldots, \boldsymbol{\lambda}_n)$ denotes the $(n-1)\ H \times 1$ vector of all users' request profile except that of user i.

We consider the scenario where all users are selfish. Specifically, each user tries to maximize his/her total utility over the H future time slots, i.e., each user i $(i \in \mathcal{N})$ tries to find a solution to the following optimization problem (OPT$_i$):

$$\text{maximize}\ \ U_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}),\ \ \boldsymbol{\lambda}_i \in Q_i. \tag{12}$$

3.2 Game Formulation and Analyses

In this section, we formulate the considered scenario into a non-cooperative game among the multiple cloud users. By employing variational inequality (VI) theory, we analyze the existence of a Nash equilibrium solution set for the formulated game. And then we propose an iterative proximal algorithm to compute a Nash equilibrium. We also analyze the convergence of the proposed algorithm.

3.2.1 Game Formulation

Game theory studies the problems in which players try to maximize their utilities or minimize their disutilities. As described in [28], a non-cooperative game consists of a set of players, a set of strategies, and preferences over the set of strategies. In this paper, each cloud user is regarded as a player, i.e., the set of players is the n cloud users. The strategy set of player i $(i \in \mathcal{N})$ is the request profile set of user i, i.e., Q_i. Then the joint strategy set of all players is given by $Q = Q_1 \times \cdots \times Q_n$.

As mentioned before, all users are considered to be selfish and each user i $(i \in \mathcal{N})$ tries to maximize his/her own utility or minimize his/her disutility while ignoring the others. In view of (12), we can observe that user i's optimization problem is equivalent to

$$\begin{aligned} \text{minimize} \quad & f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) = \sum_{h=1}^{H} \left(P_i^h + w_i \delta^h \bar{T}^h - r\lambda_i^h \right), \\ s.t. \quad & (\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) \in Q. \end{aligned} \tag{13}$$

The above formulated game can be formally defined by the tuple $G = \langle Q, \boldsymbol{f} \rangle$, where $\boldsymbol{f} = (f_1, \ldots, f_n)$. The aim of user i $(i \in \mathcal{N})$, given the other players' strategies $\boldsymbol{\lambda}_{-i}$, is to choose an $\boldsymbol{\lambda}_i \in Q_i$ such that his/her disutility function $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is minimized. That is to say, for each user i $(i \in \mathcal{N})$,

$$\boldsymbol{\lambda}_i^* \in \arg\min_{\boldsymbol{\lambda}_i \in Q_i} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}^*),\ \ \boldsymbol{\lambda}^* \in Q. \tag{14}$$

At the Nash equilibrium, each player cannot further decrease its disutility by choosing a different strategy while the strategies of other players are fixed. The equilibrium strategy profile can be found when each player's strategy is the best response to the strategies of other players.

3.2.2 Billing Parameters Analysis

It is important to investigate the way the cloud provider decides load billing scheme. In our proposed model, the request charge changes according to the total load during different time slots. The cloud provider needs to decide the proper pricing parameters a and b. The reason lies in that if the per request charge (the cost of one task request) is too high, some users may refuse to use the cloud service, and choose to finish his/her tasks locally. On the contrary, if the charge is low, the aggregated requests may be more than enough, which could lead to low service quality (long task response time). In this paper, we assume that each user i $(i \in \mathcal{N})$ has a reservation value v_i. That is to say, cloud user i will prefer to use the cloud service if $U_i\left(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}\right) \geq v_i$ and refuse to use the service otherwise. If the cloud provider wants to appeal all n cloud users to use its service while charging relatively high, then it must guarantee that the obtained utility of each user i $(i \in \mathcal{N})$ is equal to his/her reservation value v_i, i.e., $U_i\left(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}\right) = v_i$ $(\forall i \in \mathcal{N})$, which implies that

$$\sum_{h=1}^{H}\left(r\lambda_i^h - P_i^h - w_i\delta^h\bar{T}^h\right) = v_i, \forall i \in \mathcal{N}. \tag{15}$$

Considering all users together, (15) is equivalent to

$$r\Lambda - P_T - w_\Sigma \sum_{h=1}^{H}\delta^h\bar{T}^h = \sum_{i=1}^{n} v_i, \tag{16}$$

where $\Lambda = \sum\limits_{i=1}^{n}\Lambda_i$, $w_\Sigma = \sum\limits_{i=1}^{n} w_i$, and $P_T = \sum\limits_{i=1}^{n}\sum\limits_{h=1}^{H} P_i^h$.

For the cloud provider, its objective is trying to decide proper pricing parameters a and b such that its net reward, i.e., the charge to all cloud users (P_T) minus its cost (e.g., energy cost and machine maintenance cost), is maximized. In this paper, we denote π as the net profit and γ_h the cost in time slot h. When total capacity μ is determined, γ_h is assumed to be constant. Then the cloud provider's problem is to try to maximize the value π. That is

$$\text{maximize} \quad \pi = P_T(\boldsymbol{\lambda}) - \sum_{h=1}^{H}\gamma_h,$$

$$\text{s.t.} \quad r\Lambda - P_T(\boldsymbol{\lambda}) - w_\Sigma \sum_{h=1}^{H} \delta^h \bar{T}^h = \sum_{i=1}^{n} v_i, \tag{17}$$

$$\Lambda = \sum_{i=1}^{n} \Lambda_i = \sum_{h=1}^{H} \lambda_\Sigma^h, \tag{18}$$

$$\mu > \lambda_\Sigma^h \geq 0, \forall h \in \mathcal{H}. \tag{19}$$

The above optimization problem is equivalent to

$$\begin{aligned} \text{maximize} \quad & \pi = r\Lambda - w_\Sigma \sum_{h=1}^{H} \delta^h \bar{T}^h - \sum_{i=1}^{n} v_i - \sum_{h=1}^{H} \gamma_h, \\ \text{s.t.} \quad & \Lambda = \sum_{i=1}^{n} \Lambda_i = \sum_{h=1}^{H} \lambda_\Sigma^h, \\ & \mu > \lambda_\Sigma^h \geq 0, \forall h \in \mathcal{H}. \end{aligned} \tag{20}$$

Theorem 3.1 *For the cloud provider, the profit is maximized when the billing parameters (a and b) satisfy the constraint* (17) *and*

$$\lambda_\Sigma^h = \mu - \frac{(H\mu - \Lambda)\left(1 - \delta^{1/2}\right)\delta^{(h-1)/2}}{\left(1 - \delta^{H/2}\right)}, \tag{21}$$

where $h \in \mathcal{H}$.

Proof We can maximize π in (20) by using the method of Lagrange multiplier, namely,

$$\frac{\partial \pi}{\partial \lambda_\Sigma^h} = -w_\Sigma \delta^h \frac{\partial \bar{T}^h}{\partial \lambda_\Sigma^h} = -\varphi,$$

where φ is the Lagrange multiplier. That is,

$$\frac{w_\Sigma \delta^h}{\left(\mu - \lambda_\Sigma^h\right)^2} = \varphi,$$

for all $1 \leq h \leq H$, and $\sum_{h=1}^{H} \lambda_\Sigma^h = \Lambda$. After some algebraic calculation, we have

$$\varphi = \frac{w_\Sigma \delta \left(1 - \delta^{H/2}\right)^2}{(H\mu - \Lambda)^2 \left(1 - \delta^{1/2}\right)^2}.$$

Then we can obtain

$$\lambda_{\Sigma}^{h} = \mu - \frac{(H\mu - \Lambda)\left(1 - \delta^{1/2}\right)\delta^{(h-1)/2}}{\left(1 - \delta^{H/2}\right)},$$

and the result follows. □

Note that the obtained result (21) must satisfy the constraint (19), that is to say,

$$\begin{cases} \mu - \frac{(H\mu - \Lambda)\left(1 - \delta^{1/2}\right)\delta^{(H-1)/2}}{\left(1 - \delta^{H/2}\right)} \geq 0, & h = H; \\ \mu - \frac{(H\mu - \Lambda)\left(1 - \delta^{1/2}\right)}{\left(1 - \delta^{H/2}\right)} < \mu, & h = 1; \\ H\mu - \Lambda > 0. \end{cases} \tag{22}$$

We obtain

$$\begin{cases} \mu \leq \frac{c\Lambda}{cH - 1}, \\ H\mu > \Lambda, \end{cases} \tag{23}$$

where

$$c = \frac{\left(1 - \delta^{1/2}\right)\delta^{(H-1)/2}}{1 - \delta^{H/2}}.$$

Then we have

$$\frac{H}{\Lambda} < \mu \leq \frac{\Lambda}{H - 1/c}, \tag{24}$$

where

$$c = \frac{\left(1 - \delta^{1/2}\right)\delta^{(H-1)/2}}{1 - \delta^{H/2}}.$$

As mentioned before, we assume that the aggregated requests do not exceed the capacity of all the servers, i.e., $H\mu > \Lambda$. In addition, if $\mu > \frac{\Lambda}{H - 1/c}$, it is possible to shutdown some servers such that μ satisfies the constraint (24), which can also save energy cost. Therefore, in this paper, we assume that the total processing capacity μ satisfies constraint (24).

From Theorem 3.1, we know that if the cloud provider wants to appeal all the n cloud users to use its service, then proper pricing parameters a and b can be selected to satisfy constraint (17). Specifically, if b (a) is given, and a (b) is higher than the computed value from (17), then there exist some users who refuse to use the cloud service, because their obtained utilities are less than their reservation values.

3.2.3 Nash Equilibrium Analysis

In this subsection, we analyze the existence of Nash equilibrium for the formulated game $G = \langle Q, \boldsymbol{f} \rangle$ and prove the existence problem by employing variational inequality (VI) theory. Then we propose an iterative proximal algorithm (IPA). The convergence of the proposed algorithm is also analyzed. Before address the problem, we show three important properties presented in Theorems 3.2, 3.3, and 3.4, which are helpful to prove the existence of Nash equilibrium for the formulated game.

Theorem 3.2 *For each cloud user i ($i \in \mathcal{N}$), the set Q_i is convex and compact, and each disutility function $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is continuously differentiable in $\boldsymbol{\lambda}_i$. For each fixed tuple $\boldsymbol{\lambda}_{-i}$, the disutility function $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is convex in $\boldsymbol{\lambda}_i$ over the set Q_i.*

Proof It is obvious that the statements in the first part of above theorem hold. We only need to prove the convexity of $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ in $\boldsymbol{\lambda}_i$ for every fixed $\boldsymbol{\lambda}_{-i}$. This can be achieved by proving that the Hessian matrix of $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is positive semidefinite [12, 36]. Since $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) = \sum_{h=1}^{H} \left(P_i^h + w_i \delta^h \bar{T}^h - r\lambda_i^h \right)$, we have

$$\nabla_{\boldsymbol{\lambda}_i} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) = \left[\frac{\partial f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})}{\partial \lambda_i^h} \right]_{h=1}^{H} = \left(\frac{\partial f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})}{\partial \lambda_i^1}, \ldots, \frac{\partial f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})}{\partial \lambda_i^H} \right).$$

and the Hessian matrix is expressed as

$$\begin{aligned} &\nabla^2_{\boldsymbol{\lambda}_i} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) \\ &= \operatorname{diag}\left\{ \left[\frac{\partial^2 f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})}{\partial (\lambda_i^h)^2} \right]_{h=1}^{H} \right\} \\ &= \operatorname{diag}\left\{ \left[2a\left(2\lambda_\Sigma^h + \lambda_i^h\right) + \frac{2w_i\delta^h}{\left(\mu - \lambda_\Sigma^h\right)^3} \right]_{h=1}^{H} \right\}. \end{aligned} \tag{25}$$

Obviously, the diagonal matrix in (25) has all diagonal elements being positive. Thus, the Hessian matrix of $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is positive semidefinite and the result follows. The theorem is proven. □

Theorem 3.3 *The Nash equilibrium of the formulated game G is equivalent to the solution of the variational inequality (VI) problem, denoted by* VI$(Q, \mathbf{F})$*, where $Q = Q_1 \times \cdots \times Q_n$ and*

$$\mathbf{F}(\boldsymbol{\lambda}) = \left(\mathbf{F}_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})\right)_{i=1}^{n}, \tag{26}$$

with

$$\mathbf{F}_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}) = \nabla_{\boldsymbol{\lambda}_i} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i}). \tag{27}$$

Proof According to Prop. 4.1 in [37], we know that the above claim follows if two conditions are satisfied. First, for each user i ($i \in \mathcal{N}$), the strategy set Q_i is closed and convex. Second, for every fixed $\boldsymbol{\lambda}_{-i}$, the disutility function $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ is continuously differentiable and convex in $\boldsymbol{\lambda}_i \in Q_i$. By Theorem 3.2, it is easy to know that both the mentioned two conditions are satisfied in the formulated game G. Thus, the result follows. □

Theorem 3.4 *If both matrices $\mathcal{M}_1$ and $\mathcal{M}_2$ are semidefinite, then the matrix $\mathcal{M}_3 = \mathcal{M}_1 + \mathcal{M}_2$ is also semidefinite.*

Proof As mentioned above, both matrices $\mathcal{M}_1$ and $\mathcal{M}_2$ are semidefinite. Then we have $\forall \boldsymbol{x}$

$$\boldsymbol{x}^T \mathcal{M}_1 \boldsymbol{x} \geq 0 \text{ and } \boldsymbol{x}^T \mathcal{M}_2 \boldsymbol{x} \geq 0.$$

We obtain $\forall \boldsymbol{x}$,

$$\boldsymbol{x}^T \mathcal{M}_3 \boldsymbol{x} = \boldsymbol{x}^T \mathcal{M}_1 \boldsymbol{x} + \boldsymbol{x}^T \mathcal{M}_2 \boldsymbol{x} \geq 0.$$

Thus, we can conclude that $\mathcal{M}_3$ is semidefinite and the result follows. □

Recall that the objective of this subsection is to study the existence of Nash equilibrium for the formulated game $G = \langle Q, \boldsymbol{f} \rangle$ in (54). In the next theorem, we prove that if several conditions are satisfied, the existence of such Nash equilibrium is guaranteed.

Theorem 3.5 *If* $\max_{i=1,\ldots,n}(w_i) \leq 1/n$*, there exists a Nash equilibrium solution set for the formulated game $G = \langle Q, \boldsymbol{f} \rangle$.*

Proof Based on Theorem 3.3, the proof of this theorem follows if we can show that the formulated variational inequality problem VI(Q, $\mathbf{F}$) in Theorem 3.3 possesses a solution set. According to Th. 4.1 in [37], the VI(Q, $\mathbf{F}$) admits a solution set if the mapping $\mathbf{F}(\boldsymbol{\lambda})$ is monotone over Q, since the feasible set Q is compact and convex.

To prove the monotonicity of $\mathbf{F}(\boldsymbol{\lambda})$, it suffices to show that for any $\boldsymbol{\lambda}$ and $\boldsymbol{s}$ in Q,

$$(\boldsymbol{\lambda} - \boldsymbol{s})^T \left(\mathbf{F}(\boldsymbol{\lambda}) - \mathbf{F}(\boldsymbol{s})\right) \geq 0,$$

namely,

$$\sum_{h=1}^{H} \sum_{i=1}^{n} \left(\lambda_i^h - s_i^h\right) \left(\nabla_{\lambda_i^h} f_i(\boldsymbol{\lambda}) - \nabla_{s_i^h} f_i(\boldsymbol{s})\right) \geq 0. \tag{28}$$

Let $\boldsymbol{\lambda}^h = \left(\lambda_1^h, \ldots, \lambda_n^h\right)^T$ and $\boldsymbol{s}^h = \left(s_1^h, \ldots, s_n^h\right)^T$, then we can write (28) as

$$\sum_{h=1}^{H}\left(\boldsymbol{\lambda}^h - \boldsymbol{s}^h\right)\left(\nabla_{\boldsymbol{\lambda}^h} f^h\left(\boldsymbol{\lambda}^h\right) - \nabla_{\boldsymbol{s}^h} f^h\left(\boldsymbol{s}^h\right)\right) \geq 0, \tag{29}$$

where

$$f^h\left(\boldsymbol{\lambda}^h\right) = \sum_{i=1}^{n}\left(P_i^h + w_i \delta^h \bar{T}^h - r\lambda_i^h\right),$$

and

$$\nabla_{\boldsymbol{\lambda}^h} f^h\left(\boldsymbol{\lambda}^h\right) = \left(\nabla_{\lambda_1^h} f^h\left(\boldsymbol{\lambda}^h\right), \ldots, \nabla_{\lambda_n^h} f^h\left(\boldsymbol{\lambda}^h\right)\right)^T.$$

We can observe that if

$$\left(\boldsymbol{\lambda}^h - \boldsymbol{s}^h\right)\left(\boldsymbol{g}^h\left(\boldsymbol{\lambda}^h\right) - \boldsymbol{g}^h\left(\boldsymbol{s}^h\right)\right) \geq 0, \ \forall h \in \mathcal{H}, \tag{30}$$

where $\boldsymbol{g}^h\left(\boldsymbol{\lambda}^h\right) = \nabla_{\boldsymbol{\lambda}^h} f^h(\boldsymbol{\lambda}^h)$, then equation (29) holds.

Recall the definition of a monotone mapping, we can find that (30) holds if the mapping $\boldsymbol{g}^h\left(\boldsymbol{\lambda}^h\right)$ is monotone. With reference to [37], the condition in (30) is equivalent to proving the Jacobian matrix of $\boldsymbol{g}^h\left(\boldsymbol{\lambda}^h\right)$, denoted by $\mathbf{G}\left(\boldsymbol{\lambda}^h\right)$, is positive semidefinite.

After some algebraic manipulation, we can write the (i, j)th element of $\mathbf{G}\left(\boldsymbol{\lambda}^h\right)$ as

$$\left[\mathbf{G}\left(\boldsymbol{\lambda}^h\right)\right]_{i,j} = \begin{cases} 2a\left(2\lambda_\Sigma^h + \lambda_i^h\right) + \frac{2w_i\delta^h}{\left(\mu-\lambda_\Sigma^h\right)^3}, & \text{if } i = j; \\ 2a\left(\lambda_\Sigma^h + \lambda_i^h\right) + \frac{2w_i\delta^h}{\left(\mu-\lambda_\Sigma^h\right)^3}, & \text{if } i \neq j. \end{cases}$$

Since the matrix $\mathbf{G}\left(\boldsymbol{\lambda}^h\right)$ may not be symmetric, we can prove its positive semidefiniteness by showing that the symmetric matrix

$$\begin{aligned}\mathbf{G}\left(\boldsymbol{\lambda}^h\right) + \mathbf{G}\left(\boldsymbol{\lambda}^h\right)^T = \\ 2a\underbrace{\left(\boldsymbol{\lambda}^h \mathbf{1}_{n\times 1}^T + \mathbf{1}_{n\times 1}\left(\boldsymbol{\lambda}^h\right)^T + 2\lambda_\Sigma^h \mathbf{1}_{n\times n} + 2\lambda_\Sigma^h \mathbf{E}_n\right)}_{\mathcal{M}_1} \\ + 2a\sigma\underbrace{\left(\boldsymbol{w}\mathbf{1}_{n\times 1}^T + \mathbf{1}_{n\times 1}\boldsymbol{w}^T\right)}_{\mathcal{M}_2}\end{aligned}$$

is positive semidefinite [38], where

$$\sigma = \frac{\delta^h}{a\left(\mu - \lambda_\Sigma^h\right)^3},$$

$\boldsymbol{w} = (w_1, \ldots, w_n)^T$, $\mathbf{1}_{r\times s}$ is a $r \times s$ matrix with every element of 1, and $\mathbf{E}_n$ is an identity matrix. This is equivalent to showing that the smallest eigenvalue of this matrix is non-negative.

With referring to [12, 38], we obtain the two non-zero eigenvalues of $\mathcal{M}_1$ as follows:

$$\eta^1_{\mathcal{M}_1} = (n+3)\lambda_\Sigma^h + \sqrt{n\sum_{i=1}^{n}\left(\lambda_i^h + \lambda_\Sigma^h\right)^2},$$

$$\eta^2_{\mathcal{M}_1} = (n+3)\lambda_\Sigma^h - \sqrt{n\sum_{i=1}^{n}\left(\lambda_i^h + \lambda_\Sigma^h\right)^2}.$$

Let

$$A\left(\boldsymbol{\lambda}^h\right) = (n+3)\lambda_\Sigma^h,$$

and

$$B\left(\boldsymbol{\lambda}^h\right) = \sqrt{n\sum_{i=1}^{n}\left(\lambda_i^h + \lambda_\Sigma^h\right)^2},$$

and $\eta_{\min}$ be the minimal eigenvalue of matrix $\mathcal{M}_1$. Then, we have $\eta_{\min} = \min\left\{A\left(\boldsymbol{\lambda}^h\right) - B\left(\boldsymbol{\lambda}^h\right), 2\lambda_\Sigma^h\right\}$. Furthermore, we can derive that

$$\begin{aligned}\left(A\left(\boldsymbol{\lambda}^h\right)\right)^2 - \left(B\left(\boldsymbol{\lambda}^h\right)\right)^2 &= (4n+9)\left(\lambda_\Sigma^h\right)^2 - n\sum_{i=1}^{n}\left(\lambda_i^h\right)^2\\ &\geq n\left(\left(\sum_{i=1}^{n}\lambda_i^h\right)^2 - \sum_{i=1}^{n}\left(\lambda_i^h\right)^2\right) \geq 0.\end{aligned}$$

Hence, we can obtain $\eta_{\min} \geq 0$ and conclude that $\mathcal{M}_1$ is semidefinite. Similar to the semidefinite proof of $\mathcal{M}_1$, we can also obtain that if $\max_{i=1,\ldots,n}(w_i) \leq 1/n$, then $\mathcal{M}_2$ is semidefinite. By Theorem 3.4, we can conclude that the matrix $\mathbf{G}\left(\boldsymbol{\lambda}^h\right)$ is semidefinite, and the result follows. □

3.2.4 An Iterative Proximal Algorithm

Once we have established that the Nash equilibria of the formulated game $\mathbf{G} = \langle Q, \boldsymbol{f} \rangle$ exists, we are interested in obtaining a suitable algorithm to compute one of these equilibria with minimum information exchange between the multiple users and the cloud provider.

Note that we can further rewrite the optimization problem (54) as follows:

$$\begin{aligned} &\text{minimize} \quad f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) = \sum_{h=1}^{H} \left(P_i^h + w_i \delta^h \bar{T}^h - r\lambda_i^h \right), \\ &\text{s.t.} \quad \boldsymbol{\lambda}_i \in Q_i, \end{aligned} \tag{31}$$

where $\boldsymbol{\lambda}_\Sigma$ denotes the aggregated request profile of all users over the H future time slots, i.e., $\boldsymbol{\lambda}_\Sigma = \sum_{i=1}^{n} \boldsymbol{\lambda}_i$. From (31), we can see that the calculation of the disutility function of each individual user only requires the knowledge of the aggregated request profile of all users ($\boldsymbol{\lambda}_\Sigma$) rather than that the specific individual request profile of all other users ($\boldsymbol{\lambda}_{-i}$), which can bring about two advantages. On the one hand, it can reduce communication traffic between users and the cloud provider. On the other hand, it can also keep privacy for each individual user to certain extent, which is seriously considered by many cloud users.

Since all users are considered to be selfish and try to minimize their own disutilities while ignoring the others. It is natural to consider an iterative algorithm where, at every iteration k, each individual user i ($\forall i \in \mathcal{N}$) updates his/her strategy to minimize his/her own disutility function $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma)$. However, following Th. 4.2 in [37], it is not difficult to show that their convergence cannot be guaranteed in our case if the users are allowed to simultaneously update their strategies according to (31).

To overcome this issue, we consider an iterative proximal algorithm (IPA), which is based on the proximal decomposition Alg. 4.2 [37]. The proposed algorithm is guaranteed to converge to a Nash equilibrium under some additional constraints on the parameters of the algorithm. With reference to [37], consider the regularized game in which each user i ($i \in \mathcal{N}$) tries to solve the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) + \frac{\tau}{2} \left\| \boldsymbol{\lambda}_i - \bar{\boldsymbol{\lambda}}_i \right\|^2, \\ &\text{s.t.} \quad \boldsymbol{\lambda}_i, \bar{\boldsymbol{\lambda}}_i \in Q_i. \end{aligned} \tag{32}$$

That is to say, when given the aggregated requests, we must find a strategy vector $\boldsymbol{\lambda}_i^*$ for user i ($i \in \mathcal{N}$) such that

$$\boldsymbol{\lambda}_i^* \in \underset{\boldsymbol{\lambda}_i \in Q_i}{\arg\min} \left\{ f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) + \frac{\tau}{2} \left\| \boldsymbol{\lambda}_i - \bar{\boldsymbol{\lambda}}_i \right\|^2 \right\}, \tag{33}$$

Algorithm 1 Iterative Proximal Algorithm (IPA)

```
Input:
    Strategy set of all users: $\mathcal{Q}$, $\epsilon$.
Output:
    Request configuration: $\boldsymbol{\lambda}$.
1:  Initialization: Each cloud user $i$ $(i \in \mathcal{N})$ randomly choose a $\boldsymbol{\lambda}_i^{(0)} \in \mathcal{Q}_i$ and set $\bar{\boldsymbol{\lambda}}_i \leftarrow \mathbf{0}$. Set
    $S_c \leftarrow \mathcal{N}$, $S_l \leftarrow \emptyset$, and $k \leftarrow 0$.
2:  while $(S_c \neq S_l)$ do
3:      Set $S_l \leftarrow S_c$.
4:      while $(\|\boldsymbol{\lambda}^{(k)} - \boldsymbol{\lambda}^{(k-1)}\| > \epsilon)$ do
5:          for (each cloud user $i \in S_c$) do
6:              Receive $\boldsymbol{\lambda}_\Sigma^{(k)}$ from the cloud provider and compute $\boldsymbol{\lambda}_i^{(k)}$ as follows (by Algorithm 2):
7:
                $$\boldsymbol{\lambda}_i^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\lambda}_i \in \mathcal{Q}_i} \left\{ f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma^{(k)}) + \frac{\tau}{2}\left\|\boldsymbol{\lambda}_i - \bar{\boldsymbol{\lambda}}_i\right\|^2 \right\}.$$
8:              Send the updated strategy to the cloud provider.
9:          end for
10:         if (Nash equilibrium is reached) then
11:             Each user $i$ $(i \in S_c)$ updates his/her centroid $\bar{\boldsymbol{\lambda}}_i \leftarrow \boldsymbol{\lambda}_i^{(k)}$.
12:         end if
13:         Set $k \leftarrow k + 1$.
14:     end while
15:     for (each user $i \in S_c$) do
16:         if $U_i\left(\boldsymbol{\lambda}_i^{(k)}, \boldsymbol{\lambda}_\Sigma^{(k)}\right) < v_i$ then
17:             Set $\boldsymbol{\lambda}_i^{(k)} \leftarrow \mathbf{0}$, and $S_c \leftarrow S_c - \{i\}$.
18:     end for
19: end while
20: return $\boldsymbol{\lambda}^{(k)}$.
```

where $\tau(\tau > 0)$ is a regularization parameter and may guarantee the convergence of the best-response algorithm Cor. 4.1 in [37] if it is large enough. The idea is formalized in Algorithm 1.

Theorem 3.6 *There exists a constant τ_0 such that if $\tau > \tau_0$, then any sequence $\left\{\boldsymbol{\lambda}_i^{(k)}\right\}_{k=1}^{\infty}$ $(i \in S_c)$ generated by the IPA algorithm converges to a Nash equilibrium.*

Proof We may note that Algorithm 1 converges if the inner while loop (Steps 4–14) can be terminated. Therefore, if we can prove that Steps 4–14 converges, the result follows. In practice, Steps 4–14 in Algorithm 1 is a developed instance of the proximal decomposition algorithm, which is presented in Alg. 4.2 [37] for the variational inequality problem. Next, we rewrite the convergence conditions exploiting the equivalence between game theory and variational inequality (Ch. 4.2 in [37]). Given $f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_{-i})$ defined as in Eq. (54), Algorithm 1 convergences if the following two conditions are satisfied. (1) The Jacobian matrix of $\mathbf{F}$ is positive semidefinite (Th. 4.3 [37]). We denote the Jacobian by $\mathbf{JF}(\boldsymbol{\lambda}) = \left(\mathrm{J}_{\boldsymbol{\lambda}_j}\mathbf{F}_i(\boldsymbol{\lambda})\right)_{i,j=1}^{n}$, where $\mathrm{J}_{\boldsymbol{\lambda}_j}\mathbf{F}_i(\boldsymbol{\lambda}) = \left(\nabla_{\boldsymbol{\lambda}_j} f_i(\boldsymbol{\lambda})\right)_{j=1}^{n}$, which is the partial Jacobian matrix of $\mathbf{F}_i$ with respect to λ_j vector. (2) The $n \times n$ matrix $\boldsymbol{\Upsilon}_{\mathbf{F},\tau} = \boldsymbol{\Upsilon}_{\mathbf{F}} + \tau\mathbf{E}_n$ is a P-matrix (Cor. 4.1 [37]), where

$$[\boldsymbol{\Upsilon}_{\mathbf{F}}]_{ij} = \begin{cases} \alpha_i^{\min}, & \text{if } i = j; \\ -\beta_{ij}^{\max}, & \text{if } i \neq j; \end{cases}$$

with

$$\alpha_i^{\min} = \inf_{\boldsymbol{\lambda} \in Q} \eta_{\min} \left(\mathbf{J}_{\boldsymbol{\lambda}_i} \mathbf{F}_i \left(\boldsymbol{\lambda} \right) \right),$$

and

$$\beta_{ij}^{\max} = \sup_{\boldsymbol{\lambda} \in Q} \eta_{\min} \left(\mathbf{J}_{\boldsymbol{\lambda}_j} \mathbf{F}_i \left(\boldsymbol{\lambda} \right) \right),$$

and $\eta_{\min}(\mathbf{A})$ denoting the smallest eigenvalue of $\mathbf{A}$. After some algebraic manipulation, we can write the block elements of $\mathbf{JF}(\boldsymbol{\lambda})$ as

$$\begin{aligned} \mathrm{J}_{\boldsymbol{\lambda}_i} \mathbf{F}_i(\boldsymbol{\lambda}) &= \nabla^2_{\boldsymbol{\lambda}_i} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) \\ &= \operatorname{diag} \left\{ \left[2a \left(2\lambda_\Sigma^h + \lambda_i^h \right) + \frac{2 w_i \delta^h}{\left(\mu - \lambda_\Sigma^h \right)^3} \right]_{h=1}^H \right\}, \end{aligned}$$

and

$$\begin{aligned} \mathrm{J}_{\boldsymbol{\lambda}_j} \mathbf{F}_i(\boldsymbol{\lambda}) &= \nabla^2_{\boldsymbol{\lambda}_i \boldsymbol{\lambda}_j} f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) \\ &= \operatorname{diag} \left\{ \left[2a \left(\lambda_\Sigma^h + \lambda_i^h \right) + \frac{2 w_i \delta^h}{\left(\mu - \lambda_\Sigma^h \right)^3} \right]_{h=1}^H \right\}, \end{aligned}$$

for $i \neq j$ $(i, j \in \mathcal{N})$.

Next, we show that the above conditions (1) and (2) hold, respectively. By Theorem 3.2, we know that the vector function $\mathbf{F}(\boldsymbol{\lambda})$ is monotone on Q, which implies that $\mathbf{JF}(\boldsymbol{\lambda})$ is semidefinite. On the other hand, considering $\mathrm{J}_{\boldsymbol{\lambda}_i} \mathbf{F}_i(\boldsymbol{\lambda})$, we have $\alpha_i^{\min} > 0$.

Let

$$L^h(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h) = 2a \left(\lambda_\Sigma^h + \lambda_i^h \right) + \frac{2 w_i \delta^h}{\left(\mu - \lambda_\Sigma^h \right)^3}.$$

Then, we have $\frac{\partial L^h}{\partial \lambda_i^h} > 0$. As mentioned before, λ_Σ^h $(\forall h \in \mathcal{H})$ does not exceed the total processing capacity of all servers μ. We assume that $\lambda_\Sigma^h \leq (1 - \varepsilon)\mu$, where ε is a small positive constant. Then we can conclude that

$$L^h(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h) \leq 4a\,(1-\varepsilon)\,\mu + \frac{2w_{\max}\delta^h}{(\varepsilon\mu)^3},$$

where $w_{\max} = \max_{i=1,\ldots,n}\{w_i\}$.

Hence, if

$$\tau_0 \geq (n-1)\left(4a\,(1-\varepsilon)\,\mu + \frac{2w_{\max}\delta^H}{(\varepsilon\mu)^3}\right),$$

then

$$\beta_{ij}^{\max} = \sup_{\boldsymbol{\lambda}\in Q} \left\| \mathrm{J}_{\boldsymbol{\lambda}_j}\mathbf{F}_i\,(\boldsymbol{\lambda}) \right\| \leq \tau_0.$$

Then, it follows from Prop 4.3 in [37] that, if τ is chosen as in Theorem 3.6, the matrix $\boldsymbol{\Upsilon}_{\mathbf{F},\tau}$ is a P-matrix, and the result follows. □

Next, we focus on the calculation for the optimization problem in (33). Let

$$L_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) = f_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma) + \frac{\tau}{2}\left\|\boldsymbol{\lambda}_i - \bar{\boldsymbol{\lambda}}_i\right\|^2. \tag{34}$$

Then, we have to minimize $L_i(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma)$. Note that the variable in (34) is only $\boldsymbol{\lambda}_i$, therefore, we can rewrite (34) as

$$L_i(\boldsymbol{\lambda}_i, \boldsymbol{\kappa}_\Sigma) = f_i(\boldsymbol{\lambda}_i, \boldsymbol{\kappa}_\Sigma) + \frac{\tau}{2}\left\|\boldsymbol{\lambda}_i - \bar{\boldsymbol{\lambda}}_i\right\|^2, \tag{35}$$

where $\boldsymbol{\kappa}_\Sigma = \boldsymbol{\lambda}_\Sigma - \boldsymbol{\lambda}_i$. We denote R_i the constraint of user i, i.e.,

$$R_i = \lambda_i^1 + \lambda_i^2 + \ldots + \lambda_i^H = \Lambda_i,$$

and try to minimize $L_i(\boldsymbol{\lambda}_i, \boldsymbol{\kappa}_\Sigma)$ by using the method of Lagrange multiplier, namely,

$$\frac{\partial L_i}{\partial \lambda_i^h} = \phi\frac{\partial R_i}{\partial \lambda_i^h} = \phi,$$

for all $1 \leq h \leq H$, where ϕ is a Lagrange multiplier. Notice that

$$\frac{\partial P_i^h}{\partial \lambda_i^h} = a\left(2(\lambda_i^h + \kappa_\Sigma^h)\lambda_i^h + \left(\lambda_i^h + \kappa_\Sigma^h\right)^2\right) + b,$$

and

$$\frac{\partial \bar{T}^h}{\partial \lambda_i^h} = \frac{1}{\left(\mu - \kappa_\Sigma^h - \lambda_i^h\right)^2}.$$

We obtain

$$\begin{aligned}\frac{\partial L_i}{\partial \lambda_i^h} &= \frac{\partial P_i^h}{\partial \lambda_i^h} + w_i\delta^h \frac{\partial \bar{T}^h}{\partial \lambda_i^h} - r + \tau\left(\lambda_i^h - \bar{\lambda}_i^h\right)\\ &= a\left(2(\lambda_i^h+\kappa_\Sigma^h)\lambda_i^h+\left(\lambda_i^h+\kappa_\Sigma^h\right)^2\right)+b+\frac{w_i\delta^h}{\left(\mu-\kappa_\Sigma^h-\lambda_i^h\right)^2}-r+\tau\left(\lambda_i^h-\bar{\lambda}_i^h\right)=\phi.\end{aligned} \tag{36}$$

Denote $Y_i^h(\lambda_i^h, \kappa_\Sigma^h)$ as the first order of $L_i(\boldsymbol{\lambda}_i, \boldsymbol{\kappa}_\Sigma)$ on λ_i^h. Then, we have

$$\begin{aligned}Y_i^h(\lambda_i^h, \kappa_\Sigma^h) = a\left(2(\lambda_i^h + \kappa_\Sigma^h)\lambda_i^h + \left(\lambda_i^h + \kappa_\Sigma^h\right)^2\right)\\ + b + \frac{w_i\delta^h}{\left(\mu - \kappa_\Sigma^h - \lambda_i^h\right)^2} - r + \tau\left(\lambda_i^h - \bar{\lambda}_i^h\right).\end{aligned} \tag{37}$$

Since the first order of $Y_i^h(\lambda_i^h, \kappa_\Sigma^h)$ is

$$\frac{\partial Y_i^h}{\partial \lambda_i^h} = \frac{\partial^2 L_i}{\partial\left(\lambda_i^h\right)^2} = 2a\left(3\lambda_i^h + 2\kappa_\Sigma^h\right) + \frac{2w_i\delta^h}{\left(\mu - \kappa_\Sigma^h - \lambda_i^h\right)^3} + \tau > 0, \tag{38}$$

we can conclude that $Y_i^h(\lambda_i^h, \kappa_\Sigma^h)$ is an increasing positive function on λ_i^h. Based on above derivations, we propose an algorithm to calculate $\boldsymbol{\lambda}_i$ ($i \in \mathcal{N}$), which is motivated by [35].

Given $\varepsilon, \mu, a, b, r, \tau, \boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma$, and Λ_i, our optimal request configuration algorithm to find $\boldsymbol{\lambda}_i$ is given in Algorithm 2. The algorithm uses another subalgorithm Calculateλ_i^h described in Algorithm 3, which, given ε, μ, a, b, r, τ, κ_Σ^h, and ϕ, finds λ_i^h satisfies (36).

The key observation is that the left-hand side of (36), i.e., (37), is an increasing function of λ_i^h (see (38)). Therefore, given ϕ, we can find λ_i^h by using the binary search method in certain interval $[lb, ub]$ (Steps 2–9 in Algorithm 3). We set lb simply as 0. For ub, as mentioned in Theorem 3.6,

$$\lambda_i^h \leq (1 - \varepsilon)\mu,$$

where ε is a relative small positive constant. Therefore, in this paper, ub is set in Step 1 based on the above discussion. The value of ϕ can also be found by using the binary search method (Steps 10–20 in Algorithm 2). The search interval $[lb, ub]$ for ϕ is determined as follows. We set lb simply as 0. As for ub, we notice that the left-hand side of (36) is an increasing function of λ_i^h. Then, we set an increment variable inc, which is initialized as a relative small positive constant and repeatedly doubled (Step 7). The value of inc is added to ϕ to increase ϕ until the sum of λ_i^h

Algorithm 2 Calculate$\boldsymbol{\lambda}_i(\varepsilon, \mu, a, b, r, \tau, \boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma, \Lambda_i)$

```
Input: $\varepsilon, \mu, a, b, r, \tau, \boldsymbol{\lambda}_i, \boldsymbol{\lambda}_\Sigma, \Lambda_i$
Output: $\boldsymbol{\lambda}_i$.
 1: Initialization: Let $inc$ be a relative small positive constant. Set $\boldsymbol{\kappa}_\Sigma \leftarrow \boldsymbol{\lambda}_\Sigma - \boldsymbol{\lambda}_i$, $\boldsymbol{\lambda}_i \leftarrow \mathbf{0}$, and
    $\phi \leftarrow 0$.
 2: while ($\lambda_i^1 + \lambda_i^2 + \ldots + \lambda_i^H < \Lambda_i$) do
 3:     Set $mid \leftarrow \phi + inc$, and $\phi \leftarrow mid$.
 4:     for (each time slot $h \in \mathcal{H}$) do
 5:         $\lambda_i^h \leftarrow$ Calculate$\lambda_i^h(\varepsilon, \mu, a, b, r, \tau, \kappa_\Sigma^h, \phi)$.
 6:     end for
 7:     Set $inc \leftarrow 2 \times inc$.
 8: end while
 9: Set $lb \leftarrow 0$ and $ub \leftarrow \phi$.
10: while ($ub - lb > \epsilon$) do
11:     Set $mid \leftarrow (ub + lb)/2$, and $\phi \leftarrow mid$.
12:     for (each time slot $h \in \mathcal{H}$) do
13:         $\lambda_i^h \leftarrow$ Calculate$\lambda_i^h(\varepsilon, \mu, a, b, r, \tau, \kappa_\Sigma^h, \phi)$.
14:         if ($\lambda_i^1 + \lambda_i^2 + \ldots + \lambda_i^H < \Lambda_i$) then
15:             Set $lb \leftarrow mid$.
16:         else
17:             Set $ub \leftarrow mid$.
18:         end if
19:     end for
20: end while
21: Set $\phi \leftarrow (ub + lb)/2$.
22: for (each time slot $h \in \mathcal{H}$) do
23:     $\lambda_i^h \leftarrow$ Calculate$\lambda_i^h(\varepsilon, \mu, a, b, r, \tau, \kappa_\Sigma^h, \phi)$.
24: end for
25: return $\boldsymbol{\lambda}_i$.
```

Algorithm 3 Calculate$\lambda_i^h(\varepsilon, \mu, a, b, r, \tau, \kappa_\Sigma^h, \phi)$

```
Input: $\varepsilon, \mu, a, b, r, \tau, \kappa_\Sigma^h, \phi$.
Output: $\lambda_i^h$.
 1: Initialization: Set $ub \leftarrow (1 - \varepsilon)\mu - \kappa_\Sigma^h$, and $lb \leftarrow 0$.
 2: while ($ub - lb > \epsilon$) do
 3:     Set $mid \leftarrow (ub + lb)/2$, and $\lambda_i^h \leftarrow mid$.
 4:     if ($Y_i^h(\lambda_i^h, \kappa_\Sigma^h) < \phi$) then
 5:         Set $lb \leftarrow mid$.
 6:     else
 7:         Set $ub \leftarrow mid$.
 8:     end if
 9: end while
10: Set $\lambda_i^h \leftarrow (ub + lb)/2$.
11: return $\lambda_i^h$.
```

($h \in \mathcal{H}$) found by Calculateλ_i^h is at least Λ_i (Steps 2–8). Once $[lb, ub]$ is decided, ϕ can be searched based on the fact that $Y_i^h(\lambda_i^h, \boldsymbol{\lambda}_{-i}^h)$ is an increasing function of λ_i^h. After ϕ is determined (Step 21), $\boldsymbol{\lambda}_i$ can be computed (Steps 22–24).

Finally, we can describe the proposed iterative proximal algorithm as follows. At the beginning, each cloud user i ($i \in \mathcal{N}$) sends his/her weight value (w_i) and total task request (Λ_i) to the cloud provider. Then the cloud provider computes τ as in Theorem 3.6 according to the aggregated information and chooses proper parameters a and b such that constraint (17) is satisfied. After this, the cloud provider puts the computed load billing parameters a and b into public information exchange module. Then, at each iteration k, the cloud provider broadcasts a synchronization signal and the current aggregated request profile $\boldsymbol{\lambda}_{\Sigma}^{(k)}$. Within iteration k, each user receives the aggregated profile $\boldsymbol{\lambda}_{\Sigma}^{(k)}$ and computes his/her strategy by solving its own optimization problem in (32), and then sends the newly updated strategy to the cloud provider. Lastly, as indicated in Steps 10–12 of Algorithm 1, the cloud provider checks whether the Nash equilibrium has been achieved and if so, it broadcasts a signal to inform all users to update their centroid $\bar{\boldsymbol{\lambda}}_i$. It also checks whether all cloud users' strategies are unchanged and if so, it informs all users to choose whether they still prefer to the cloud service due to their reserved values. This process continues until the set of the remaining cloud users and their corresponding strategies are kept fixed. In this paper, we assume that the strategies of all cloud users are unchanged if $\left\|\boldsymbol{\lambda}^{(k)} - \boldsymbol{\lambda}^{(k-1)}\right\| \leq \epsilon$, where $\boldsymbol{\lambda}^{(k)} = \left(\boldsymbol{\lambda}_i^{(k)}\right)_{i=1}^{n}$ with $\boldsymbol{\lambda}_i^{(k)} = \left(\left(\lambda_i^h\right)^{(k)}\right)_{h=1}^{H}$. The parameter ϵ is a pre-determined relatively small constant. We also denote S_c as the current set of remaining cloud users. Note that the individual strategies are not revealed among the users in any case, and only the aggregated request profile $\boldsymbol{\lambda}^{(k)}$, which is determined at the cloud provider adding the individual H-time slots ahead request profile, is communicated between the cloud provider and multiple cloud users.

3.3 Performance Evaluation of IPA

In this section, we provide some numerical results to validate our theoretical analyses and illustrate the performance of the IPA algorithm.

In the following simulation results, we consider the scenario consisting of maximal 50 cloud users. Each time slot is set as one hour of a day and H is set as 24. As shown in Table 1, the aggregated request (Λ) is varied from 50 to 500 with increment 50. The number of cloud users (n) is varied from 5 to 50 with increment 5. Each cloud user i ($i \in \mathcal{N}$) chooses a weight value from 0 to $1/n$ to balance his/her time utility and net profit. For simplicity, the reservation value v_i for each user i ($i \in \mathcal{N}$) and billing parameter b are set to zero. Market benefit factor r is set to 50, deteriorating rate on time utility δ is equal to 1.2, and ε is set as 0.01. The total capacity of all servers μ is selected to satisfy constraint (24) and another billing parameter a is computed according to (17). In our simulation, the initial strategy configuration, i.e., before using IPA algorithm, is randomly generated from $\mathcal{Q}$.

Figure 2 presents the utility results for five different cloud users versus the number of iterations of the proposed IPA algorithm. Specifically, Fig. 2 presents the utility results of 5 randomly selected cloud users (users 1, 9, 23, 38, and 46) with a scenario consisting of 50 cloud users. We can observe that the utilities of all the users seem to increase and finally reach a relative stable state with the increase of iteration number. The reason behind lies in that the request strategies of all the users

Table 1 System parameters

System parameters	Value (Fixed)–[Varied range] (increment)
Aggregated task requests (Λ)	(500)–[100, 500] (50)
Number of cloud users (n)	(50)–[5, 50] (5)
Weight value (w_i)	[0, 1/n]
Reservation value (v_i)	0
Other parameters (ε, b, r, δ)	(0.01, 0, 50, 1.2)

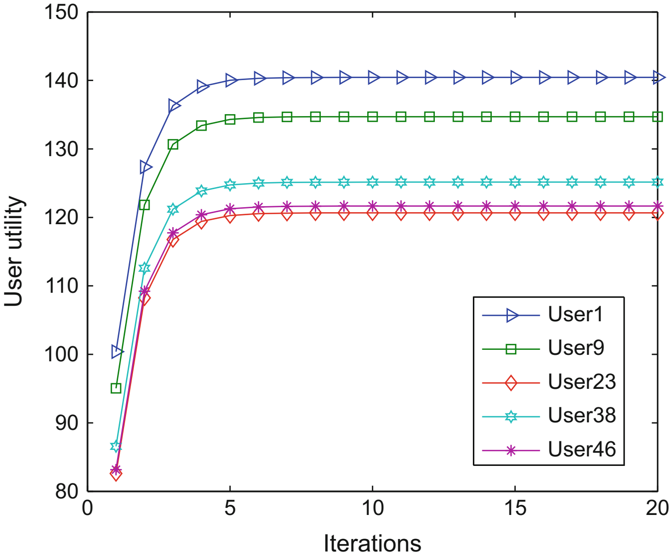

Fig. 2 Convergence process

keep unchanged, i.e., reach a Nash equilibrium solution after several iterations. This trend also reflects the convergence process of our proposed IPA algorithm. It can be seen that the developed algorithm converges to a Nash equilibrium very quickly. Specifically, the utility of each user has already achieved a relatively stable state after about 8 iterations, which verifies the validness of Theorem 3.6, as well as displays the high efficiency of the developed algorithm.

In Fig. 3, we compare the aggregated request profile of all cloud users with the situation before and after IPA algorithm. Specifically, Fig. 3 shows the aggregated requests in different time slots. The situation before IPA algorithm corresponds to a feasible strategy profile randomly generated in the initialization stage, while the situation after IPA algorithm corresponds to the result obtained by using our proposed IPA algorithm. Obviously, the proposed service reservation scheme encourages the cloud users to shift their task requests in peak time slots to non-peak time slots, resulting in a more balanced load shape and lower total load. We can also observe that the aggregated requests in different time slots are almost the same. To

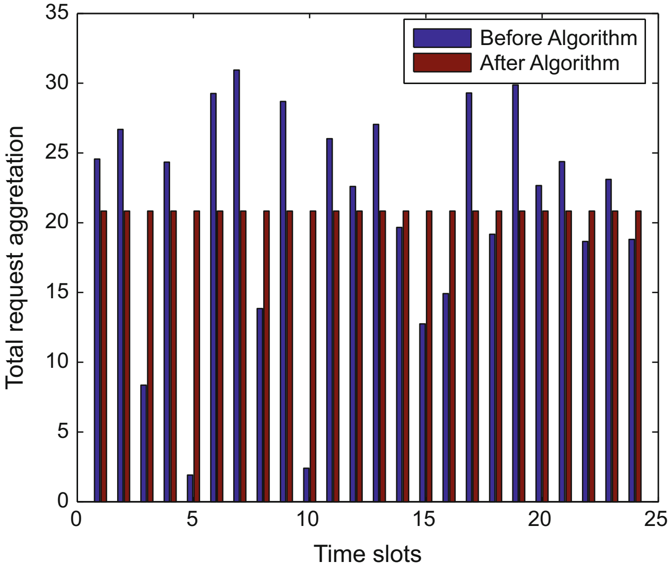

Fig. 3 Aggregation load

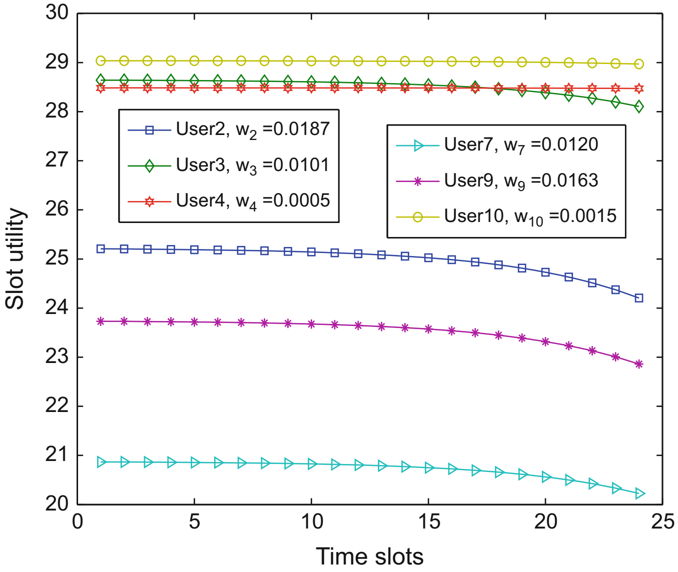

Fig. 4 Specific slot utility

demonstrate this phenomenon, we further investigate the specific utilities of some users and their corresponding strategies in different time slots, which are presented in Figs. 4 and 5.

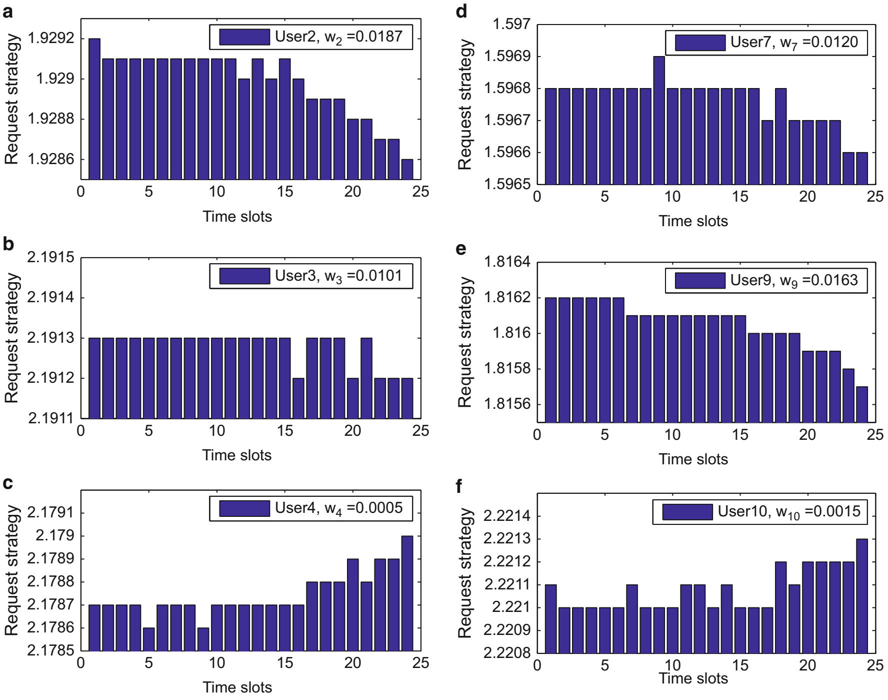

Fig. 5 Specific slot shifting

In Figs. 4 and 5, we plot the utility shape and the request profile of some cloud users for the developed IPA algorithm for a scenario of 10 users. Figure 4 presents the utility shape under the developed algorithm over future 24 time slots. We randomly select 6 users (users 2, 3, 4, 7, 9, and 10). It can be seen that the utilities in different time slots of all users tend to decrease at different degrees. Specifically, the slot utilities of the users with higher weights have a clearly downward trend and tend to decrease sharply in later time slots (users 2, 3, 7, 9). On the other hand, the slot utilities of the users with lower weights decline slightly (users 4, 10). Figure 5 exhibits the corresponding request strategies of the users shown in Fig. 4. We can observe that the slot utilities of the users with higher weights tend to decrease (users 2, 3, 7, 9) while those of the users with lower weights tend to increase (users 4, 10). Furthermore, the aggregated requests increase or decrease sharply in later time slots. The reason behind lies in the fact that in our proposed model, we take into the average response time into account and the deteriorating factor of the value grows exponentially, which also demonstrates the downward trends shown in Fig. 4. On the other hand, the weights are chosen randomly, there could be a balance between the increment and the decrement of the utilities. Hence, the aggregated requests in different time slots make little differences (Fig. 3).

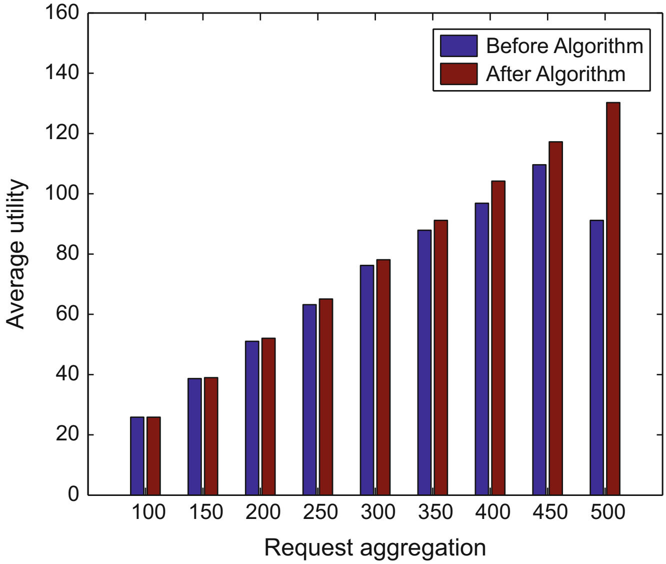

Fig. 6 Average utility vs. aggregation

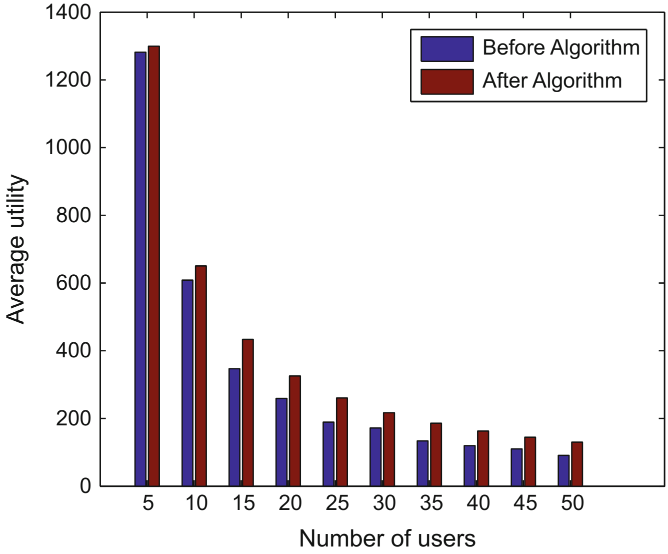

Fig. 7 Average utility vs. number of users

Figures 6 and 7 present the average utility versus the increase of request aggregation and the number of users, respectively. Figure 6 illustrates the average utility results with the linear increment of request aggregation. We can observe that the average utility also linearly increases with the increase of request aggregation. No matter what the request aggregation is, the average utility obtained after our proposed IPA algorithm is better than that of the initial strategy profile. Moreover, the differences between the results before IPA algorithm and those after the algorithm are also increases. That is to say, our proposed IPA algorithm makes significant sense when the aggregated requests are somewhat large. Figure 7 shows the impacts of number of users. It can be seen that both of the results after IPA algorithm and before algorithm are inversely proportional to the number of uses. The reason behind lies in that the variation of number of users makes little impact on the average utility value when the request aggregation is fixed. Moreover, similar to the results presented in Fig. 6, the average utility obtained after IPA algorithm is always better than that of the initial strategy profile.

4 A Framework of Price Bidding Configurations for Resource Usage in Cloud Computing

4.1 System Model and Problem Formulation

To begin with, we present our system model in the context of a service cloud provider with multiple cloud users, and establish some important results. In this paper, we are concerned with a market with a service cloud provider and n cloud users, who are competing for using the computing resources provided by the cloud provider. We denote the set of users as $\mathcal{N} = \{1, \ldots, n\}$. Each cloud user wants to bid for using some servers for several future time slots. The arrival requests from cloud user i ($i \in \mathcal{N}$) is assumed to follow a Poisson process. The cloud provider consists of multiple zones. In each zone, there are many homogeneous servers. In this paper, we focus on the price bidding for resource usage in a same zone and assume that the number of homogeneous servers in the zone is m. The cloud provider tries to allocate cloud user i ($i \in \mathcal{N}$) with m_i servers without violating the constraint $\sum_{i \in \mathcal{N}} m_i \leq m$. The allocated m_i servers for cloud user i ($i \in \mathcal{N}$) are modeled by an M/M/m queue, only serving the requests from user i for t_i future time slots. We summarize all the notations used in this sub-section in the notation table.

4.1.1 Bidding Strategy Model

As mentioned above, the n cloud users compete for using the m servers by bidding different strategies. Specifically, each cloud user responds by bidding with a per server usage price p_i (i.e., the payment to use one server in a time slot) and the number of time slots t_i to use cloud service. Hence, the bid of cloud user i ($i \in \mathcal{N}$) is an ordered pair $b_i = \langle p_i, t_i \rangle$.

Table 2 Notations

Notation	Description
n	Number of cloud users
m	Number of servers in a zone in the cloud
$\mathcal{N}$	Set of the n cloud users
$\mathcal{M}$	Set of the m servers in the zone in the cloud
p_i	Bidding price of cloud user i
$\underline{p}$	Minimal bidding price for a server in one time slot
$\bar{p}_i$	Maximal possible bidding price of cloud user i
$\mathcal{P}_i$	The set of price bidding strategies of cloud user i
t_i	Reserved time slots of cloud user i
b_i	Bidding strategy of cloud user i
$\boldsymbol{b}$	Bidding strategy of all cloud users
$\boldsymbol{b}_{-i}$	Bidding strategy profile of all users except that of user i
λ_i^t	Request arrival rate of cloud user i in t-th time slot
$\boldsymbol{\lambda}_i^{t_i}$	User i's request profile over the t_i future time slots
m_i	Allocated number of servers for cloud user i
$\boldsymbol{m}$	Allocated server vector for all cloud users
μ_i	Processing rate of a server for requests from user i
$\bar{T}_i^t$	Average response time of cloud user i in t-th time slot
$\Xi_{\mathcal{S}}$	Aggregated payment from users in $\mathcal{S}$ for using a server
P_i^t	Payment of cloud users i in t-th time slot
P_T	Total payment from all cloud users
r_i	Benefit obtained by user i by finishing one task request
u_i^t	Utility of cloud user i in t-th time slot
u_i	Total utility of cloud user i over t_i future time slots
$\boldsymbol{u}$	Utility vector of all cloud users

We assume that cloud user i ($i \in \mathcal{N}$) bids a price $p_i \in \mathcal{P}_i$, where $\mathcal{P}_i = \left[\underline{p}, \bar{p}_i\right]$, with $\bar{p}_i$ denoting user i's maximal possible bidding price. $\underline{p}$ is a conservative bidding price, which is determined by the cloud provider. If $\underline{p}$ is greater than $\bar{p}_i$, then $\mathcal{P}_i$ is empty and the cloud user i ($i \in \mathcal{N}$) refuses to use cloud service. As mentioned above, each cloud user i ($i \in \mathcal{N}$) bids for using some servers for t_i future time slots. In our work, we assume that the reserved time slots t_i is a constant once determined by the cloud user i. We define user i's ($i \in \mathcal{N}$) request profile over the t_i future time slots as follow:

$$\boldsymbol{\lambda}_i^{t_i} = \left(\lambda_i^1, \ldots, \lambda_i^{t_i}\right)^T, \tag{39}$$

where λ_i^t ($t \in \mathcal{T}_i$) with $\mathcal{T}_i = \{1, \ldots, t_i\}$, is the arrival rate of requests from cloud user i in the t-th time slot. The arrival of the requests in different time slots of are assumed to follow a Poisson process.

4.1.2 Server Allocation Model

We consider a server allocation model motivated by [39, 40], where the allocated number of servers is proportional fairness. That is to say, the allocated share of servers is the ratio between the cloud user's product value of his/her bidding price with reserved time slots and the summation of all product values from all cloud users. Then, each cloud user i $(i \in \mathcal{N})$ is allocated a portion of servers as

$$m_i\left(b_i, \boldsymbol{b}_{-i}\right) = \left\lfloor \frac{p_i t_i}{\sum_{j \in \mathcal{N}} p_j t_j} \cdot m \right\rfloor, \tag{40}$$

where $\boldsymbol{b}_{-i} = (b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n)$ denotes the vector of all users' bidding profile except that of user i, and $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x. We design a server allocation model as Eq. (40) for two considerations. On one hand, if the reserved time slots to use cloud service t_i is large, the cloud provider can charge less for one server in a unit of time to appeal more cloud users, i.e., the bidding price p_i can be smaller. In addition, for the cloud user i $(i \in \mathcal{N})$, he/she may be allocated more servers, which can improve his/her service time utility. On the other hand, if the bidding price p_i is large, this means that the cloud user i $(i \in \mathcal{N})$ wants to pay more for per server usage in a unit of time to allocate more servers, which can also improve his/her service time utility. This is also beneficial to the cloud provider due to the higher charge for each server. Therefore, we design a server allocation model as Eq. (40), which is proportional to the product of p_i and t_i.

4.1.3 Cloud Service Model

As mentioned in the beginning, the allocated m_i servers for cloud user i $(i \in \mathcal{N})$ are modeled as an M/M/m queue, only serving the requests from cloud user i for t_i future time slots. The processing capacity of each server for requests from cloud user i $(i \in \mathcal{N})$ is presented by its service rate μ_i. The requests from cloud user i $(i \in \mathcal{N})$ in t-th $(t \in \mathcal{T}_i)$ time slot are assumed to follow a Poisson process with average arrival rate λ_i^t.

Let π_{ik}^t be the probability that there are k service requests (waiting or being processed) in the t-th time slot and $\rho_i^t = \lambda_i^t \big/ (m_i \mu_i)$ be the corresponding service utilization in the M/M/m queuing system. With reference to [7], we obtain

$$\pi_{ik}^t = \begin{cases} \frac{1}{k!}\left(m_i \rho_i^t\right)^k \pi_{i0}^t, & k < m_i; \\ \frac{m_i^{m_i} \left(\rho_i^t\right)^k}{m_i!} \pi_{i0}^t, & k \geq m_i; \end{cases} \tag{41}$$

where

$$\pi_{i0}^{t} = \left\{ \sum_{l=0}^{m_i-1} \frac{1}{l!} \left(m_i \rho_i^t\right)^l + \frac{1}{m_i!} \cdot \frac{\left(m_i \rho_i^t\right)^{m_i}}{1 - \rho_i^t} \right\}^{-1}. \tag{42}$$

The average number of service requests (in waiting or in execution) in t-th time slot is

$$\bar{N}_i^t = \sum_{k=0}^{\infty} k \pi_{ik}^t = \frac{\pi_{im_i}^t}{1 - \rho_i^t} = m_i \rho_i^t + \frac{\rho_i^t}{1 - \rho_i^t} \Pi_i^t, \tag{43}$$

where Π_i^t represents the probability that the incoming requests from cloud user i ($i \in \mathcal{N}$) need to wait in queue in the t-th time slot.

Applying Little's result, we get the average response time in the t-th time slot as

$$\bar{T}_i^t = \frac{\bar{N}_i^t}{\lambda_i^t} = \frac{1}{\lambda_i^t} \left(m_i \rho_i^t + \frac{\rho_i^t}{1 - \rho_i^t} \Pi_i^t \right). \tag{44}$$

In this work, we assume that the allocated servers for each cloud user will likely keep busy, because if no so, a user can bid lower price to obtain less servers such that the computing resources can be fully utilized. For analytical tractability, Π_i^t is assumed to be 1. Therefore, we have

$$\bar{T}_i^t = \frac{\bar{N}_i^t}{\lambda_i^t} = \frac{1}{\lambda_i^t} \left(m_i \rho_i^t + \frac{\rho_i^t}{1 - \rho_i^t} \right) = \frac{1}{\mu_i} + \frac{1}{m_i \mu_i - \lambda_i^t}. \tag{45}$$

Note that the request arrival rate from a user should never exceed the total processing capacity of the allocated servers. In our work, we assume that the remaining processing capacity for serving user i ($i \in \mathcal{N}$) is at least $\sigma \mu_i$, where σ is a relative small positive constant. That is, if $\lambda_i^t > (m_i - \sigma)\mu_i$, cloud user i ($i \in \mathcal{N}$) should reduce his/her request arrival rate to $(m_i - \sigma)\mu_i$. Otherwise, server crash would be occurred. Hence, we have

$$\bar{T}_i^t = \frac{1}{\mu_i} + \frac{1}{m_i \mu_i - \chi_i^t}, \tag{46}$$

where χ_i^t is the minimum value of λ_i^t and $(m_i - \sigma)\,\mu_i$, i.e., $\chi_i^t = \min\left\{\lambda_i^t, (m_i - \sigma)\,\mu_i\right\}$.

4.1.4 Architecture Model

In this subsection, we model the architecture of our proposed framework to price bids for resource usage in cloud computing. The multiple users can make appropriate bidding decisions through the information exchange module. As shown in Fig. 8, each cloud user i ($i \in \mathcal{N}$) is equipped with a utility function (u_i), the

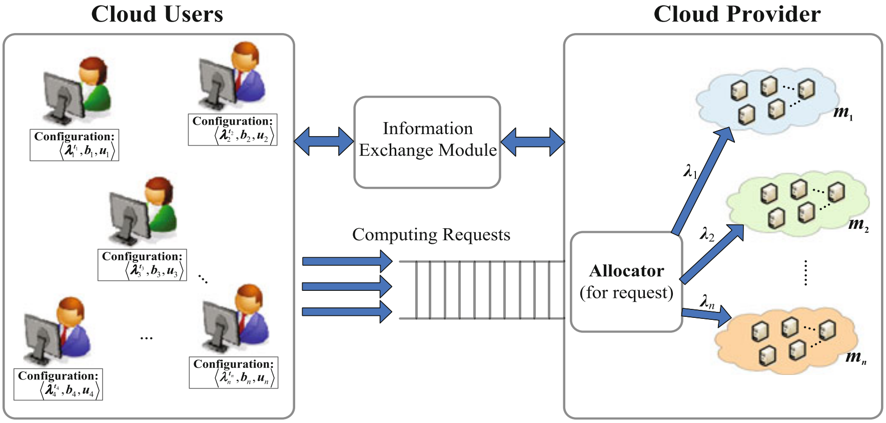
Cloud Users
Cloud Provider
Configuration: $\langle \boldsymbol{\lambda}_1^{t_1}, \boldsymbol{b}_1, \boldsymbol{u}_1 \rangle$
Configuration: $\langle \boldsymbol{\lambda}_2^{t_2}, \boldsymbol{b}_2, \boldsymbol{u}_2 \rangle$
Configuration: $\langle \boldsymbol{\lambda}_3^{t_3}, \boldsymbol{b}_3, \boldsymbol{u}_3 \rangle$
Configuration: $\langle \boldsymbol{\lambda}_4^{t_4}, \boldsymbol{b}_4, \boldsymbol{u}_4 \rangle$
Configuration: $\langle \boldsymbol{\lambda}_n^{t_n}, \boldsymbol{b}_n, \boldsymbol{u}_n \rangle$
Information Exchange Module
Computing Requests
Allocator (for request)
λ_1
λ_2
λ_n
$\boldsymbol{m}_1$
$\boldsymbol{m}_2$
$\boldsymbol{m}_n$

Fig. 8 Architecture model

request arrival rate over reserved time slots ($\boldsymbol{\lambda}_i^{t_i}$), and the bidding configuration (b_i), i.e., the payment strategy for one server in a unit of time and the reserved time slots. Let $\Xi_{\mathcal{N}}$ be the aggregated payment from all cloud users for using a server, then we have $\Xi_{\mathcal{N}} = \sum_{i=1}^{n} p_i t_i$. Denote $\boldsymbol{m} = (m_i)_{i \in \mathcal{N}}$ as the server allocation vector, $\boldsymbol{b} = (b_i)_{i \in \mathcal{N}}$ as the corresponding bids, and $\boldsymbol{u} = (u_i)_{i \in \mathcal{N}}$ as the utility functions of all cloud users. The cloud provider consists of m homogeneous servers and communicates some information (e.g., conservative bidding price $\underline{p}$, current aggregated payment from all cloud users for using a server $\Xi_{\mathcal{N}}$) with multiple users through the information exchange module. When multiple users try to make price bidding strategies for resource usage in the cloud provider, they first get information from the information exchange module, then configure proper bidding strategies ($\boldsymbol{b}$) such that their own utilities ($\boldsymbol{u}$) are maximized. After this, they send the updated strategies to the cloud provider. The procedure is terminated when the set of remaining cloud users, who prefer to use the cloud service, and their corresponding bidding strategies are kept fixed.

4.1.5 Problem Formulation

Now, let us consider user i's ($i \in \mathcal{N}$) utility in time slot t ($t \in \mathcal{T}_i$). A rational cloud user will seek a bidding strategy to maximize his/her expected net reward by finishing the requests, i.e., the benefit obtained by choosing the cloud service minus his/her payment. Since all cloud users are charged based on their bidding prices and allocated number of servers, we denote the cloud user i's payment in time slot t by $P_i^t(b_i, \boldsymbol{b}_{-i})$, where $P_i^t(b_i, \boldsymbol{b}_{-i}) = p_i m_i(b_i, \boldsymbol{b}_{-i})$ with $\boldsymbol{b}_{-i} = (b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n)$ denoting the vector of all users' bidding profile except that of user i. Denote $P_T(b_i, \boldsymbol{b}_{-i})$ as the aggregated payment from all cloud users, i.e., the revenue of the cloud provider. Then, we have

$$P_T\left(b_i, \boldsymbol{b}_{-i}\right) = \sum_{i=1}^{n} \sum_{t=1}^{t_i} P_i^t\left(b_i, \boldsymbol{b}_{-i}\right) = \sum_{i=1}^{n} \left(p_i m_i\left(b_i, \boldsymbol{b}_{-i}\right) t_i\right). \tag{47}$$

On the other hand, since a user will be more satisfied with much faster service, we also take the average response time into account. From Eq. (46), we know that the average response time of user i ($i \in \mathcal{N}$) is impacted by m_i and χ_i^t, where $\chi_i^t = \min\left\{\lambda_i^t, (m_i - \sigma)\,\mu_i\right\}$. The former is varied by $(b_i, \boldsymbol{b}_{-i})$, and the latter is determined by λ_i^t and m_i. Hence, we denote the average response time of user i as $\bar{T}_i^t\left(b_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$. More formally, the utility of user i ($i \in \mathcal{N}$) in time slot t is defined as

$$u_i^t\left(b_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = r_i \chi_i^t - \delta_i P_i^t\left(b_i, \boldsymbol{b}_{-i}\right) - w_i \bar{T}_i^t\left(b_i, \boldsymbol{b}_{-i}, \lambda_i^t\right), \tag{48}$$

where χ_i^t is the minimum value of λ_i^t and $(m_i\left(b_i, \boldsymbol{b}_{-i}\right) - \sigma)\,\mu_i$, i.e., $\chi_i^t = \min\left\{\lambda_i^t, (m_i\left(b_i, \boldsymbol{b}_{-i}\right) - \sigma)\,\mu_i\right\}$ with σ denoting a relative small positive constant, r_i ($r_i > 0$) is the benefit factor (the reward obtained by finishing one task request) of user i, δ_i ($\delta_i > 0$) is the payment cost factor, and w_i ($w_i > 0$) is the waiting cost factor, which reflects its urgency. If a user i ($i \in \mathcal{N}$) is more concerned with service time utility, then the associated waiting factor w_i might be larger. Otherwise, w_i might be smaller, which implies that the user i is more concerned with profit.

Since the reserved server usage time t_i is a constant and known to cloud user i ($i \in \mathcal{N}$), we use $u_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ instead of $u_i^t\left(b_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$. For further simplicity, we use P_i^t and $\bar{T}_i^t$ to denote $P_i^t\left(b_i, \boldsymbol{b}_{-i}\right)$ and $T_i^t\left(b_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$, respectively. Following the adopted bidding model, the total utility obtained by user i ($i \in \mathcal{N}$) over all t_i time slots can thus be given by

$$\begin{aligned} u_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) &= \sum_{t=1}^{t_i} u_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) \\ &= \sum_{t=1}^{t_i} \left(r_i \chi_i^t - P_i^t - w_i \bar{T}_i^t\right). \end{aligned} \tag{49}$$

In our work, we assume that each user i ($i \in \mathcal{N}$) has a reservation value v_i. That is to say, cloud user i will prefer to use the cloud service if $u_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) \geq v_i$ and refuse to use the cloud service otherwise.

We consider the scenario where all users are selfish. Specifically, each cloud user tries to maximize his/her total utility over the t_i future time slots, i.e., each cloud user i ($i \in \mathcal{N}$) tries to find a solution to the following optimization problem (OPT$_i$):

$$\text{maximize } u_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right), \; p_i \in \mathcal{P}_i. \tag{50}$$

Remark 4.1 In finding the solution to (OPT_i), the bidding strategies of all other users are kept fixed. In addition, the number of reserved time slots once determined by a user is constant. So the variable in (OPT_i) is the bidding price of cloud user i, i.e., p_i.

4.2 Game Formulation and Analyses

In this section, we formulated the considered scenario into a non-cooperative game among the multiple cloud users. By relaxing the condition that the allocated number of servers for each user can be fractional, we analyze the existence of a Nash equilibrium solution set for the formulated game. We also propose an iterative algorithm to compute a Nash equilibrium and then analyze its convergence. Finally, we revise the obtained Nash equilibrium solution and propose an algorithm to characterize the whole process of the framework.

4.2.1 Game Formulation

Game theory studies the problems in which players try to maximize their utilities or minimize their disutilities. As described in [5], a non-cooperative game consists of a set of players, a set of strategies, and preferences over the set of strategies. In this paper, each cloud user is regarded as a player, i.e., the set of players is the n cloud users. The strategy set of player i ($i \in \mathcal{N}$) is the price bidding set of user i, i.e., $\mathcal{P}_i$. Then the joint strategy set of all players is given by $\mathcal{P} = \mathcal{P}_1 \times \cdots \times \mathcal{P}_n$.

As mentioned before, all users are considered to be selfish and each user i ($i \in \mathcal{N}$) tries to maximize his/her own utility or minimize his/her disutility while ignoring those of the others. Denote

$$\psi_i^t \left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \delta_i P_i^t + w_i T_i^t - r_i \chi_{it}. \tag{51}$$

In view of (49), we can observe that user i's optimization problem (OPT_i) is equivalent to

$$\begin{aligned} \text{minimize} \quad & f_i \left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) = \sum_{t=1}^{t_i} \psi_i^t \left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right), \\ \text{s.t.} \quad & \left(p_i, \boldsymbol{p}_{-i}\right) \in \mathcal{P}. \end{aligned} \tag{52}$$

The above formulated game can be formally defined by the tuple $G = \langle \mathcal{P}, \boldsymbol{f} \rangle$, where $\boldsymbol{f} = (f_1, \ldots, f_n)$. The aim of cloud user i ($i \in \mathcal{N}$), given the other players' bidding strategies $\boldsymbol{b}_{-i}$, is to choose a bidding price $p_i \in \mathcal{P}_i$ such that his/her disutility function $f_i \left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ is minimized.

Definition 4.1 (Nash equilibrium) A *Nash equilibrium* of the formulated game $G = \langle \mathcal{P}, \boldsymbol{f} \rangle$ defined above is a price bidding profile $\boldsymbol{p}^*$ such that for every player i $(i \in \mathcal{N})$:

$$p_i^* \in \underset{p_i \in \mathcal{P}_i}{\arg\min}\, f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right), \ \boldsymbol{p}^* \in \mathcal{P}. \tag{53}$$

At the Nash equilibrium, each player cannot further decrease its disutility by choosing a different price bidding strategy while the strategies of other players are fixed. The equilibrium strategy profile can be found when each player's strategy is the best response to the strategies of other players.

4.2.2 Nash Equilibrium Existence Analysis

In this subsection, we analyze the existence of Nash equilibrium for the formulated game $G = \langle \mathcal{P}, \boldsymbol{f} \rangle$ by relaxing one condition that the allocated number of servers for each user can be fractional. Before addressing the equilibrium existence analysis, we show two properties presented in Theorems 4.1 and 4.2, which are helpful to prove the existence of Nash equilibrium for the formulated game.

Theorem 4.1 *Given a fixed $\boldsymbol{b}_{-i}$ and assuming that $r_i \geq w_i \big/ \left(\sigma^2 \mu_i^2\right)$ $(i \in \mathcal{N})$, then each of the functions $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ $(t_i \in \mathcal{T}_i)$ is convex in $p_i \in \mathcal{P}_i$.*

Proof Obviously, $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ $(t \in \mathcal{T}_i)$ is a real continuous function defined on $\mathcal{P}_i$. The proof of this theorem follows if we can show that $\forall p_{(1)}, p_{(2)} \in \mathcal{P}_i$,

$$\psi_i^t\left(\theta p_{(1)} + (1-\theta)\, p_{(2)}, \boldsymbol{b}_{-i}, \lambda_i^t\right) \leq \theta \psi_i^t\left(p_{(1)}, \boldsymbol{b}_{-i}, \lambda_i^t\right) + (1-\theta)\, \psi_i^t\left(p_{(2)}, \boldsymbol{b}_{-i}, \lambda_i^t\right),$$

where $0 < \theta < 1$.

Notice that, $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ is a piecewise function and the breakpoint satisfies $(m_i - \sigma)\, \mu_i = \lambda_i^t$. Then, we obtain the breakpoint as

$$p_i^t = \frac{m_i\, \boldsymbol{\Xi}_{\Xi_{\mathcal{N}\setminus\{i\}}}}{(m - m_i)\, t_i} = \frac{\left(\lambda_i^t + \sigma \mu_i\right) \Xi_{\mathcal{N}\setminus\{i\}}}{\left((m-\sigma)\, \mu_i - \lambda_i^t\right) t_i},$$

where $\Xi_{\mathcal{N}\setminus\{i\}}$ denotes the aggregated payment from all cloud users in $\mathcal{N}$ except of user i, i.e., $\Xi_{\mathcal{N}\setminus\{i\}} = \sum_{j \in \mathcal{N}, j \neq i} p_i t_i$. Next, we discuss the convexity of the function $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$.

Since

$$\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \delta_i P_i^t + w_i \bar{T}_i^t - r_i \chi_i^t,$$

where $\chi_i^t = \min\left\{(m_i - \sigma)\, \mu_i, \lambda_i^t\right\}$, we have

$$\frac{\partial \psi_i^t}{\partial p_i}\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \delta_i \frac{\partial P_i^t}{\partial p_i} + w_i \frac{\partial \bar{T}_i^t}{\partial p_i} - r_i \frac{\partial \chi_i^t}{\partial p_i}.$$

On the other hand, since $\frac{\partial \bar{T}_i^t}{\partial p_i} = 0$ for $p_i \in \left[\underline{p}, p_i^t\right)$ and $\frac{\partial \chi_i^t}{\partial p_i} = 0$ for $p_i \in \left(p_i^t, \bar{p}_i\right]$, we obtain

$$\frac{\partial}{\partial p_i}\varphi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \begin{cases} \delta_i \frac{\partial P_i^t}{\partial p_i} - r_i \frac{\partial \chi_i^t}{\partial p_i}, & p_i < p_i^t; \\ \delta_i \frac{\partial P_i^t}{\partial p_i} + w_i \frac{\partial \bar{T}_i^t}{\partial p_i}, & p_i > p_i^t. \end{cases}$$

Namely,

$$\frac{\partial}{\partial p_i}\varphi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \begin{cases} \delta_i \left(\frac{m p_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2} + m_i\right) - \frac{m r_i \mu_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2}, & p_i < p_i^t; \\ \delta_i \left(\frac{m p_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2} + m_i\right) - \frac{m w_i \mu_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{(m_i \mu_i - \lambda_i^t)^2 \Xi_{\mathcal{N}}^2}, & p_i > p_i^t, \end{cases}$$

where

$$\Xi_{\mathcal{N}} = \Xi_{\mathcal{N}\setminus\{i\}} + p_i t_i = \sum\nolimits_{j \in \mathcal{N}} p_j t_j.$$

We can further obtain

$$\frac{\partial^2}{\partial p_i^2}\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) = \begin{cases} \frac{2m t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2}\left(\frac{(r_i \mu_i - p_i) t_i}{\Xi_{\mathcal{N}}} + 1\right), & p_i < p_i^t; \\ \frac{2m t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2}\left(1 - \frac{p_i t_i}{\Xi_{\mathcal{N}}}\right) + & \\ \quad \frac{2m w_i \mu_i t_i^2 \Xi_{\mathcal{N}\setminus\{i\}}}{(m_i \mu_i - \lambda_i^t)^2 \Xi_{\mathcal{N}}^3}\left(\frac{\mu_i \Xi_{\mathcal{N}\setminus\{i\}}}{(m_i \mu_i - \lambda_i^t) \Xi_{\mathcal{N}}} + 1\right), & p_i > p_i^t. \end{cases}$$

Obviously,

$$\frac{\partial^2}{\partial p_i^2}\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) > 0,$$

for all $p_i \in \left[\underline{p}, p_i^t\right)$ and $p_i \in \left(p_i^t, \bar{p}_i\right]$. Therefore, $\forall p_{(1)}, p_{(2)} \in \left[\underline{p}, p_i^t\right)$ or $\forall p_{(1)}, p_{(2)} \in \left(p_i^t, \bar{p}_i\right]$,

$$\begin{aligned} &\psi_i^t\left(\theta p_{(1)} + (1-\theta)\, p_{(2)}, \boldsymbol{b}_{-i}, \lambda_i^t\right) \\ &\quad \leq \theta \psi_i^t\left(p_{(1)}, \boldsymbol{b}_{-i}, \lambda_i^t\right) + (1-\theta)\, \psi_i^t\left(p_{(2)}, \boldsymbol{b}_{-i}, \lambda_i^t\right), \end{aligned}$$

where $0 < \theta < 1$.

Next, we focus on the situation where $p_{(1)} \in \left[\underline{p}, p_i^t\right)$ and $p_{(2)} \in \left(p_i^t, \bar{p}_i\right]$. Since $\psi_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right)$ is convex on $\left[\underline{p}, p_i^t\right)$ and $\left(p_i^t, \bar{p}_i\right]$, respectively. We only need to

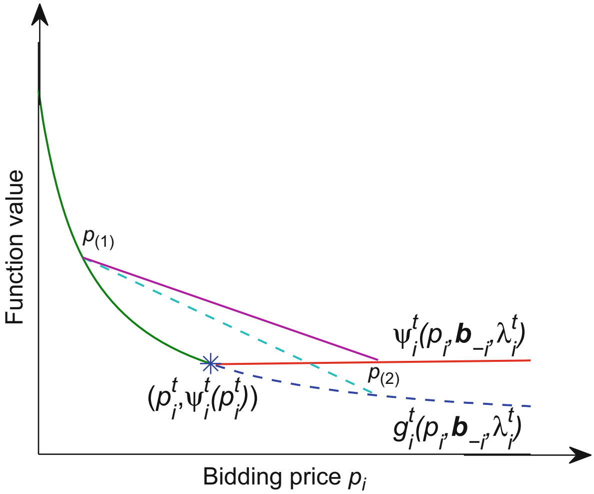

Fig. 9 An illustration

prove that the value of $\psi_i^t\left(p_i^t, \boldsymbol{b}_i, \lambda_i^t\right)$ is less than that of in the linear function value connected by the point in $p_{(1)}$ and the point in $p_{(2)}$, i.e.,

$$\psi_i^t\left(p_i^t, \boldsymbol{b}_i, \lambda_i^t\right) \leq \theta_i^t \psi_i^t\left(p_{(1)}, \boldsymbol{b}_i, \lambda_i^t\right) + \left(1 - \theta_i^t\right) \psi_i^t\left(p_{(2)}, \boldsymbol{b}_i, \lambda_i^t\right),$$

where $\theta_i^t = \frac{p_{(2)} - p_i^t}{p_{(2)} - p_{(1)}}$. We proceed as follows (see Fig. 9).

Define a function $g_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right)$ on $p_i \in \mathcal{P}_i$, where

$$g_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right) = \delta_i p_i m_i + \frac{w_i\left(\sigma + 1\right)}{\sigma \mu_i} - r_i\left(m_i - \sigma\right)\mu_i.$$

We have

$$\psi_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right) = g_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right),$$

for all $\underline{p} \leq p_i \leq p_i^t$. If $r_i \geq w_i / \left(\sigma^2 \mu_i^2\right)$, then

$$\begin{aligned} &\frac{\partial}{\partial p_i} g_i^t\left(p_i, \boldsymbol{b}_i, \lambda_i^t\right) \\ &\quad = \delta_i\left(\frac{m p_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2} + m_i\right) - \frac{m r_i \mu_i t_i \Xi_{\mathcal{N}\setminus\{i\}}}{\Xi_{\mathcal{N}}^2} \end{aligned}$$

$$\leq \delta_i \left(\frac{m p_i t_i \Xi_{\mathcal{N} \setminus \{i\}}}{\Xi_{\mathcal{N}}^2} + m_i \right) - \frac{m w_i \mu_i t_i \Xi_{\mathcal{N} \setminus \{i\}}}{\left(m_i \mu_i - \lambda_i^t\right)^2 \Xi_{\mathcal{N}}^2}$$
$$= \frac{\partial}{\partial p_i} \psi_i^t \left(p_i, \boldsymbol{b}_i, \lambda_i^t\right),$$

for all $p_i^t < p_i \leq \bar{p}_i$. We have

$$\psi_i^t \left(p_i, \boldsymbol{b}_i, \lambda_i^t\right) \geq g_i^t \left(p_i, \boldsymbol{b}_i, \lambda_i^t\right),$$

for all $p_i^t < p_i \leq \bar{p}_i$.

On the other hand, according to the earlier derivation, we know that

$$\frac{\partial^2}{\partial p_i^2} g_i^t \left(p_i, \boldsymbol{b}_i, \lambda_i^t\right) > 0,$$

for all $p_i \in \mathcal{P}_i$. That is, $g_i^t \left(p_i, \boldsymbol{b}_i, \lambda_i^t\right)$ is a convex function on $\mathcal{P}_i$, and we obtain

$$\begin{aligned}
&\psi_i^t \left(p_i^t, \boldsymbol{b}_i, \lambda_i^t\right) \\
&\quad \leq \theta_i^t g_i^t \left(p_{(1)}, \boldsymbol{b}_i, \lambda_i^t\right) + \left(1 - \theta_i^t\right) g_i^t \left(p_{(2)}, \boldsymbol{b}_i, \lambda_i^t\right) \\
&\quad = \theta_i^t \psi_i^t \left(p_{(1)}, \boldsymbol{b}_i, \lambda_i^t\right) + \left(1 - \theta_i^t\right) g_i^t \left(p_{(2)}, \boldsymbol{b}_i, \lambda_i^t\right) \\
&\quad \leq \theta_i^t \psi_i^t \left(p_{(1)}, \boldsymbol{b}_i, \lambda_i^t\right) + \left(1 - \theta_i^t\right) \psi_i^t \left(p_{(2)}, \boldsymbol{b}_i, \lambda_i^t\right).
\end{aligned}$$

Thus, we have $\psi_i^t \left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ is convex on $p_i \in \mathcal{P}_i$. This completes the proof and the result follows. □

Theorem 4.2 *If both functions $\mathcal{K}_1(x)$ and $\mathcal{K}_2(x)$ are convex in $x \in \mathcal{X}$, then the function $\mathcal{K}_3(x) = \mathcal{K}_1(x) + \mathcal{K}_2(x)$ is also convex in $x \in \mathcal{X}$.*

Proof As mentioned above, both of the functions $\mathcal{K}_1(x)$ and $\mathcal{K}_2(x)$ are convex in $x \in \mathcal{X}$. Then we have $\forall x_1, x_2 \in \mathcal{X}$,

$$\mathcal{K}_1 \left(\theta x_1 + (1 - \theta) x_2\right) \leq \theta \mathcal{K}_1 (x_1) + (1 - \theta) \mathcal{K}_1 (x_2),$$

and

$$\mathcal{K}_2 \left(\theta x_1 + (1 - \theta) x_2\right) \leq \theta \mathcal{K}_2 (x_1) + (1 - \theta) \mathcal{K}_2 (x_2),$$

where $0 < \theta < 1$. We further obtain $\forall x_1, x_2 \in \mathcal{X}$,

$$\begin{aligned}
&\mathcal{K}_3 \left(\theta x_1 + (1 - \theta) x_2\right) \\
&\quad = \mathcal{K}_1 \left(\theta x_1 + (1 - \theta) x_2\right) + \mathcal{K}_2 \left(\theta x_1 + (1 - \theta) x_2\right)
\end{aligned}$$

$$\leq \theta \left(\mathcal{K}_1(x_1) + \mathcal{K}_2(x_1)\right) + (1-\theta)\left(\mathcal{K}_1(x_2) + \mathcal{K}_2(x_2)\right)$$
$$= \theta \mathcal{K}_3(x_1) + (1-\theta)\mathcal{K}_3(x_2).$$

Thus, we can conclude that $\mathcal{K}_3(x)$ is also convex in $x \in \mathcal{X}$ and the result follows. □

Theorem 4.3 *There exists a Nash equilibrium solution set for the formulated game $G = \langle \mathcal{P}, \boldsymbol{f} \rangle$, given that the condition $r_i \geq w_i \big/ \left(\sigma^2 \mu_i^2\right)$ $(i \in \mathcal{N})$ holds.*

Proof According to [12, 20], the proof of this theorem follows if the following two conditions are satisfied. (1) For each cloud user i $(i \in \mathcal{N})$, the set $\mathcal{P}_i$ is convex and compact, and each disutility function $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ is continuous in $p_i \in \mathcal{P}_i$. (2) For each fixed tuple $\boldsymbol{b}_{-i}$, the function $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ is convex in p_i over the set $\mathcal{P}_i$.

It is obvious that the statements in the first part hold. We only need to prove the convexity of $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ in p_i for every fixed $\boldsymbol{b}_{-i}$. By Theorem 4.1, we know that if $r_i \geq w_i \big/ \left(\sigma^2 \mu_i^2\right)$ $(i \in \mathcal{N})$, then each of the functions $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ $(t \in \mathcal{T}_i)$ is convex in $p_i \in \mathcal{P}_i$. In addition, according to the property presented in Theorem 4.2, it is easy to deduce that

$$f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) = \sum_{t=1}^{t_i} \psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right),$$

is also convex in $p_i \in \mathcal{P}_i$. Thus, the result follows. □

4.2.3 Nash Equilibrium Computation

Once we have established that the Nash equilibrium of the formulated game $G = \langle \mathcal{P}, \boldsymbol{f} \rangle$ exists, we are interested in obtaining a suitable algorithm to compute one of these equilibriums with minimum information exchange between the multiple users and the cloud providers.

Note that we can further rewrite the optimization problem (52) as follows:

$$\begin{aligned} \text{minimize} \quad & f_i\left(p_i, \Xi_{\mathcal{N}}, \boldsymbol{\lambda}_i^{t_i}\right) = \sum_{t=1}^{t_i} \psi_i^t\left(p_i, \Xi_{\mathcal{N}}, \lambda_i^t\right), \\ \text{s.t.} \quad & \left(p_i, \boldsymbol{p}_{-i}\right) \in \mathcal{P}, \end{aligned} \tag{54}$$

where $\Xi_{\mathcal{N}}$ denotes the aggregated payments for each server from all cloud users, i.e., $\Xi_{\mathcal{N}} = \sum_{j \in \mathcal{N}} p_j t_j$. From (54), we can observe that the calculation of the disutility function of each individual user only requires the knowledge of the aggregated payments for a server from all cloud users ($\Xi_{\mathcal{N}}$) rather than that the specific individual bidding strategy profile ($\boldsymbol{b}_{-i}$), which can bring about two advantages. On the one hand, it can reduce communication traffic between users and the cloud provider. On the other hand, it can also keep privacy for each individual user to certain extent, which is seriously considered by many cloud users.

Since all users are considered to be selfish and try to minimize their own disutility while ignoring those of the others. It is natural to consider an iterative algorithm where, at every iteration k, each individual user i ($i \in \mathcal{N}$) updates his/her price bidding strategy to minimize his/her own disutility function $f_i\left(p_i, \Xi_{\mathcal{N}}, \boldsymbol{\lambda}_i^{t_i}\right)$. The idea is formalized in Algorithm 1.

Algorithm 4 $\mathcal{I}$terative $\mathcal{A}$lgorithm ($\mathcal{IA}$)

```
Input: $\mathcal{S}, \boldsymbol{\lambda}_{\mathcal{S}}, \epsilon$.
Output: $\boldsymbol{p}_{\mathcal{S}}$.
//Initialize $p_i$ for each user $i \in \mathcal{S}$
for (each cloud user $i \in \mathcal{S}$) do
    set $p_i^{(0)} \leftarrow \underline{b}$.
end for
Set $k \leftarrow 0$.
//Find equilibrium bidding prices
while ($\left\| \boldsymbol{p}_{\mathcal{S}}^{(k)} - \boldsymbol{p}_{\mathcal{S}}^{(k-1)} \right\| > \epsilon$) do
    for (each cloud user $i \in \mathcal{S}$) do
        Receive $\Xi_{\mathcal{S}}^{(k)}$ from the cloud provider and compute $p_i^{(k+1)}$ as follows (By Algorithm 2):
            $$p_i^{(k+1)} \leftarrow \underset{p_i \in \mathcal{P}_i}{\arg\min}\, f_i\left(p_i, \Xi_{\mathcal{S}}^{(k)}, \boldsymbol{\lambda}_i^{t_i}\right).$$
        Send the updated price bidding strategy to the cloud provider.
    end for
    Set $k \leftarrow k + 1$.
end while
return $\boldsymbol{p}_{\mathcal{S}}^{(k)}$.
```

Given $\mathcal{S}$, $\boldsymbol{\lambda}_{\mathcal{S}}$, and ϵ, where $\mathcal{S}$ is the set of cloud users who want to use the cloud service, $\boldsymbol{\lambda}_{\mathcal{S}}$ is the request vector of all cloud users in $\mathcal{S}$, i.e., $\boldsymbol{\lambda}_{\mathcal{S}} = \left\{\boldsymbol{\lambda}_i^{t_i}\right\}_{i \in \mathcal{S}}$, and ϵ is a relative small constant. The iterative algorithm ($\mathcal{IA}$) finds optimal bidding prices for all cloud users in $\mathcal{S}$. At the beginning of the iterations, the bidding price of each cloud user is set as the conservative bidding price ($\underline{p}$). We use a variable k to index each of the iterations, which is initialized as zero. At the beginning of the iteration k, each of the cloud users i ($i \in \mathcal{N}$) receives the value $\Xi_{\mathcal{S}}^{(k)}$ from the cloud provider and computes his/her optimal bidding price such that his/her own disutility function $f_i\left(p_i, \Xi_{\mathcal{S}}^{(k)}, \boldsymbol{\lambda}_i^{t_i}\right)$ ($i \in \mathcal{S}$) is minimized. Then, each of the cloud users in $\mathcal{S}$ updates their price bidding strategy and sends the updated value to the cloud provider. The algorithm terminates when the price bidding strategies of all cloud users in $\mathcal{S}$ are kept unchanged, i.e., $\left\| \boldsymbol{p}_{\mathcal{S}}^{(k)} - \boldsymbol{p}_{\mathcal{S}}^{(k-1)} \right\| \leq \epsilon$.

In subsequent analyses, we show that the above algorithm always converges to a Nash equilibrium if one condition is satisfied for each cloud user. If so, we have an algorithmic tool to compute a Nash equilibrium solution. Before addressing the convergency problem, we first present a property presented in Theorem 4.4, which is helpful to derive the convergence result.

Theorem 4.4 *If* $r_i > \max\left\{\frac{2\delta_i \bar{p}_i}{\mu_i}, \frac{w_i}{\sigma^2\mu_i^2}\right\}$ *($i \in \mathcal{N}$), then the optimal bidding price p_i^* ($p_i^* \in \mathcal{P}_i$) of cloud user i ($i \in \mathcal{N}$) is a non-decreasing function with respect to $\Xi_{\mathcal{N}\backslash\{i\}}$, where $\Xi_{\mathcal{N}\backslash\{i\}} = \sum_{j\in\mathcal{N}} p_j t_j - p_i t_i$.*

Proof According to the results in Theorem 4.1 we know that for each cloud user i ($i \in \mathcal{N}$), given a fixed $\boldsymbol{b}_{-i}$, there are t_i breakpoints for the function $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$. We denote $\mathcal{B}_i$ as the set of the t_i breakpoints, then we have $\mathcal{B}_i = \left\{p_i^t\right\}_{t\in\mathcal{T}_i}$, where

$$p_i^t = \frac{m_i\, \Xi_{\mathcal{N}\backslash\{i\}}}{(m - m_i)\, t_i} = \frac{\left(\lambda_i^t + \sigma\mu_i\right) \Xi_{\mathcal{N}\backslash\{i\}}}{\left((m-\sigma)\,\mu_i - \lambda_i^t\right) t_i}.$$

Combining the above t_i breakpoints with two end points, i.e., $\underline{p}$ and $\bar{p}_i$, we obtain a new set $\mathcal{B}_i \cup \left\{\underline{p}, \bar{p}_i\right\}$. Reorder the elements in $\mathcal{B}_i \cup \left\{\underline{p}, \bar{p}_i\right\}$ such that $p_i^{(0)} \le p_i^{(1)} \le \cdots \le p_i^{(t_i)} \le p_i^{(t_i+1)}$, where $p_i^{(0)} = \underline{p}$ and $p_i^{(t_i+1)} = \bar{p}_i$. Then, we obtain a new ordered set $\mathcal{B}_i^{'}$. We discuss the claimed theorem by distinguishing three cases according to the first derivative results of the disutility function $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ on $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$.

Case 1 $\frac{\partial}{\partial \bar{p}_i} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) < 0$. According to the results in Theorem 4.2, we know that the second derivative of $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ on $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$ is positive, i.e., $\frac{\partial^2}{\partial p_i^2} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) > 0$ for all $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$. In addition, if $r_i \ge w_i \big/ \left(\sigma^2\mu_i^2\right)$, the left partial derivative is less than that of the right partial derivative in each of the breakpoints in $\mathcal{B}_i$. Therefore, if $\frac{\partial}{\partial \bar{p}_i} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) < 0$, then $\frac{\partial}{\partial p_i} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) < 0$ for all $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$. Namely, $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ is a decreasing function on $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$. Hence, the optimal bidding price of cloud user i is $p_i^* = \bar{p}_i$. That is to say, the bidding price of cloud user i increases with respect to Ξ_{-i}.

Case 2 $\frac{\partial}{\partial \underline{p}} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) > 0$. Similar to Case 1, according to the results in Theorem 4.2, we know that $\frac{\partial^2}{\partial p_i^2} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) > 0$ for all $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$. Hence, if $\frac{\partial}{\partial \underline{p}} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) > 0$, $f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right)$ is an increasing function for all $p_i \in \mathcal{P}_i\backslash\mathcal{B}_i$. Therefore, under this situation, the optimal bidding price of cloud user i is $p_i^* = \underline{p}$, i.e., the optimal bidding price is always the conservative bidding price, which is the initialized value.

Case 3 $\frac{\partial}{\partial \underline{p}} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) < 0$ and $\frac{\partial}{\partial \bar{p}_i} f_i\left(p_i, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) > 0$. Under this situation, it means that there exists an optimal bidding price $p_i^* \in \mathcal{P}_i \backslash \mathcal{B}_i^{'}$ such that

$$\begin{aligned} \frac{\partial}{\partial p_i} f_i\left(p_i^*, \boldsymbol{b}_{-i}, \boldsymbol{\lambda}_i^{t_i}\right) &= \sum_{t=1}^{t_i} \frac{\partial}{\partial p_i} \psi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^{t_i}\right) \\ &= \sum_{t=1}^{t_i} \left(\frac{\partial P_i^t}{\partial p_i} + w_i \frac{\partial \bar{T}_i^t}{\partial p_i} - r \frac{\partial \chi_i^t}{\partial p_i}\right) = 0. \end{aligned} \tag{55}$$

Otherwise, the optimal bidding price for cloud user i ($i \in \mathcal{N}$) is in $\mathcal{B}_i^{'}$. If above equation holds, then there exists an integer $t^{'}$ ($0 \le t^{'} \le t_i$), such that the optimal bidding price p_i^* is in $(p_i^{(t^{'})}, p_i^{(t^{'}+1)}) \subseteq \mathcal{P}_i \backslash \mathcal{B}_i{'}$.

According to the derivations in Theorem 4.1, we know that the first derivative of $\psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ is

$$\begin{aligned} &\frac{\partial}{\partial p_i} \psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) \\ &= \begin{cases} \delta_i\left(\frac{m p_i t_i \Xi_{\mathcal{N}\backslash\{i\}}}{\Xi_{\mathcal{N}}^2} + m_i\right) - \frac{m r_i \mu_i t_i \Xi_{\mathcal{N}\backslash\{i\}}}{\Xi_{\mathcal{N}}^2}, & p_i < p_i^t; \\ \delta_i\left(\frac{m p_i t_i \Xi_{\mathcal{N}\backslash\{i\}}}{\Xi_{\mathcal{N}}^2} + m_i\right) - \frac{m w_i \mu_i t_i \Xi_{\mathcal{N}\backslash\{i\}}}{(m_i \mu_i - \lambda_i^t)^2 \Xi_{\mathcal{N}}^2}, & p_i > p_i^t, \end{cases} \end{aligned}$$

That is,

$$\begin{aligned} &\frac{\partial}{\partial p_i} \psi_i^t\left(p_i, \boldsymbol{b}_{-i}, \lambda_i^t\right) \\ &= \begin{cases} \frac{m t_i}{\Xi_{\mathcal{N}}^2}\left(\delta_i p_i\left(p_i t_i + 2\Xi_{\mathcal{N}\backslash\{i\}}\right) - r_i u_i \Xi_{\mathcal{N}\backslash\{i\}}\right), & p_i < p_i^t; \\ \frac{m t_i}{\Xi_{\mathcal{N}}^2}\left(\delta_i p_i\left(p_i t_i + 2\Xi_{\mathcal{N}\backslash\{i\}}\right) - \frac{w_i u_i \Xi_{\mathcal{N}\backslash\{i\}}}{(m_i \mu_i - \lambda_i^t)^2}\right), & p_i > p_i^t. \end{cases} \end{aligned}$$

Therefore, the Eq. (55) is equivalent to the following equation:

$$h\left(p_i^*\right) = \sum_{t=1}^{t_i} \varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right) = 0,$$

where

$$\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right)$$

$$= \begin{cases} \delta_i p_i^* \left(p_i^* t_i + 2\Xi_{\mathcal{N}\backslash\{i\}}\right) - r_i u_i \Xi_{\mathcal{N}\backslash\{i\}}, & p_i^* < p_i^t; \\ \delta_i p_i^* \left(p_i^* t_i + 2\Xi_{\mathcal{N}\backslash\{i\}}\right) - \frac{w_i u_i \Xi_{\mathcal{N}\backslash\{i\}}}{\left(m_i \mu_i - \lambda_i^t\right)^2}, & p_i^* > p_i^t. \end{cases}$$

After some algebraic manipulation, we can write the first derivative result of $\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ on p_i^* as

$$\frac{\partial}{\partial p_i^*}\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right)$$
$$= \begin{cases} 2\delta_i \left(p_i^* t_i + \Xi_{\mathcal{N}\backslash\{i\}}\right), & p_i^* < p_i^t; \\ 2\delta_i \left(p_i^* t_i + \Xi_{\mathcal{N}\backslash\{i\}}\right) + \frac{2 w_i t_i \mu_i^2 \Xi_{\mathcal{N}\backslash\{i\}}^2}{\left(m_i \mu_i - \lambda_i^t\right)^3 \Xi_{\mathcal{N}}^2}, & p_i^* > p_i^t, \end{cases}$$

and the first derivative result of the function $\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right)$ on $\Xi_{\mathcal{N}\backslash\{i\}}$ as

$$\frac{\partial}{\partial \Xi_{\mathcal{N}\backslash\{i\}}}\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right)$$
$$= \begin{cases} 2\delta_i p_i^* - r_i u_i, & p_i^* < p_i^t; \\ 2\delta_i p_i^* - r_i u_i - \frac{w_i \mu_i}{\left(m_i \mu_i - \lambda_i^t\right)^2} \\ \qquad - \frac{2 m w_i \mu_i^2 p_i^* t_i \Xi_{\mathcal{N}\backslash\{i\}}}{\left(m_i \mu_i - \lambda_i^t\right)^3 \Xi_{\mathcal{N}}^2}, & p_i^* > p_i^t. \end{cases}$$

Obviously, we have

$$\frac{\partial}{\partial p_i^*}\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right) > 0,$$

for all $p_i^* \in \mathcal{P}_i \backslash \mathcal{B}_i^{\prime}$. If $r_i > 2\delta_i \bar{p}_i \big/ \mu_i$, then

$$\frac{\partial}{\partial \Xi_{\mathcal{N}\backslash\{i\}}}\varphi_i^t\left(p_i^*, \boldsymbol{b}_{-i}, \lambda_i^t\right) < 0.$$

Therefore, if $r_i > \max\left\{\frac{2\delta_i \bar{p}_i}{\mu_i}, \frac{w_i}{\sigma^2 \mu_i^2}\right\}$, the function $h\left(b_i^*\right)$ decreases with the increase of $\Xi_{\mathcal{N}\backslash\{i\}}$. If $\Xi_{\mathcal{N}\backslash\{i\}}$ increases, to maintain the equality $h\left(b_i^*\right) = 0$, b_i^* must increase. Hence, b_i^* increases with the increase of $\Xi_{\mathcal{N}\backslash\{i\}}$. This completes the proof and the result follows. □

Theorem 4.5 *Algorithm $\mathcal{IA}$ converges to a Nash equilibrium, given that the condition $r_i > \max\left\{\frac{2\delta_i \bar{p}_i}{\mu_i}, \frac{w_i}{\sigma^2 \mu_i^2}\right\}$ $(i \in \mathcal{N})$ holds.*

Proof We are now ready to show that the proposed $\mathcal{IA}$ algorithm always converges to a Nash equilibrium solution, given that $r_i > \left\{\frac{2\delta_i \bar{p}_i}{\mu_i}, \frac{w_i}{\sigma^2\mu_i^2}\right\}$ $(i \in \mathcal{N})$ holds. Let $p_i^{(k)}$ be the optimal bidding price of cloud user i $(i \in \mathcal{N})$ at the k-th iteration. We shall prove above claim by induction that $p_i^{(k)}$ is non-decreasing in k. In addition, since p_i^* is bounded by $\bar{p}_i$, this establishes the result that $p_i^{(k)}$ always converges.

By Algorithm 1, we know that the bidding price of each cloud user is initialized as the conservative bidding price, i.e., $p_i^{(0)}$ is set as $\underline{p}$ for each of the cloud users i $(i \in \mathcal{N})$. Therefore, after the first iteration, we obtain the results $p_i^{(1)} \geq p_i^{(0)}$ for all $i \in \mathcal{N}$. This establishes our induction basis.

Assuming that the result is true in the k-th iteration, i.e., $p_i^{(k)} \geq p_i^{(k-1)}$ for all $i \in \mathcal{N}$. Then, we need to show that in the $(k+1)$-th iteration, $p_i^{(k+1)} \geq p_i^{(k)}$ is satisfied for all $i \in \mathcal{N}$. We proceed as follows.

By Theorem 4.4, we know that if $r_i > 2\delta_i \bar{p}_i \big/ \mu_i$, the optimal bidding price p_i^* of cloud user i $(i \in \mathcal{N})$ increases with the increase of $\Xi_{\mathcal{N}\backslash\{i\}}$, where $\Xi_{\mathcal{N}\backslash\{i\}} = \sum_{j\in\mathcal{N}, j\neq i} p_j t_j$. In addition, we can deduce that

$$\Xi_{\mathcal{N}\backslash\{i\}}^{(k)} = \sum_{j\in\mathcal{N}, j\neq i} p_j^{(k)} t_j \geq \sum_{j\in\mathcal{N}, j\neq i} p_j^{(k-1)} t_j = \Xi_{\mathcal{N}\backslash\{i\}}^{(k-1)}.$$

Therefore, the optimal bidding price of cloud user i $(i \in \mathcal{N})$ in the $(k+1)$-th iteration $p_i^{(k+1)}$, which is a function of $\Xi_{\mathcal{N}\backslash\{i\}}^{(k)}$, satisfies $p_i^{(k+1)} \geq p_i^{(k)}$ for all $i \in \mathcal{N}$. Thus, the result follows. □

Next, we focus on the calculation for the optimal bidding price p_i^* in problem (54), i.e., calculate

$$p_i^* \in \underset{p_i \in \mathcal{P}_i}{\arg\min}\, f_i\left(p_i, \Xi_{\mathcal{N}}, \boldsymbol{\lambda}_i^{t_i}\right). \tag{56}$$

From Theorem 4.5, we know that the optimal bidding price p_i^* of cloud user i $(i \in \mathcal{N})$ is either in $\mathcal{B}_i^{'}$ or in $\mathcal{P}_i \backslash \mathcal{B}_i^{'}$ such that

$$\begin{aligned}\frac{\partial}{\partial p_i} f_i\left(p_i^*, \Xi_{\mathcal{N}}, \boldsymbol{\lambda}_i^{t_i}\right) &= \sum_{t=1}^{t_i} \frac{\partial}{\partial p_i} \psi_i^t\left(p_i^*, \Xi_{\mathcal{N}}, \lambda_i^t\right) \\ &= \sum_{t=1}^{t_i} \left(\delta_i \frac{\partial P_i^t}{\partial p_i} + w_i \frac{\partial \bar{T}_i^t}{\partial p_i} - r_i \frac{\partial \chi_i^t}{\partial p_i}\right) = 0, \end{aligned} \tag{57}$$

where $\mathcal{B}_i^{'}$ is an ordered set for all elements in $\mathcal{B}_i \cup \left\{\underline{p}, \bar{p}_i\right\}$, and $\mathcal{B}_i$ is the set of t_i breakpoints of cloud user i $(i \in \mathcal{N})$, i.e., $\mathcal{B}_i = \left\{p_i^t\right\}_{t\in\mathcal{T}_i}$ with

Algorithm 5 $Calculate\, p_i(\Xi, \boldsymbol{\lambda}_i^{t_i}, \epsilon)$

```
Input: $\Xi, \boldsymbol{\lambda}_i^{t_i}, \epsilon$.
Output: $p_i^*$.
Set $t^{'} \leftarrow 0$.
//Find $p_i^*$ in $\mathcal{P}_i \backslash \mathcal{B}_i^{'}$
while $(t^{'} \leq t_i)$ do
    Set $ub \leftarrow p_i^{(t^{'}+1)} - \epsilon$, and $lb \leftarrow p_i^{(t^{'})} + \epsilon$.
    if $(\frac{\partial}{\partial p_i} f_i\left(lb, \Xi, \boldsymbol{\lambda}_i^{t_i}\right) > 0$ or $\frac{\partial}{\partial p_i} f_i\left(ub, \Xi, \boldsymbol{\lambda}_i^{t_i}\right) < 0)$ then
        Set $t^{'} \leftarrow t^{'} + 1$; continue.
    end if
    while $(ub - lb > \epsilon)$ do
        Set $mid \leftarrow (ub + lb)/2$, and $p_i \leftarrow mid$.
        if $(\frac{\partial}{\partial p_i} f_i\left(p_i, \Xi, \boldsymbol{\lambda}_i^{t_i}\right) < 0)$ then
            Set $lb \leftarrow mid$.
        else
            Set $ub \leftarrow mid$.
        end if
    end while
    Set $p_i \leftarrow (ub + lb)/2$; break.
end while
//Otherwise, find $p_i^*$ in $\mathcal{B}_i^{'}$
if $(t^{'} = t_i + 1)$ then
    Set min $\leftarrow +\infty$.
    for (each break point $p_i^{(t^{'})} \in \mathcal{B}_i^{'}$) do
        if $(f_i\left(p_i^{(t^{'})}, \Xi, \boldsymbol{\lambda}_i^{t_i}\right) <$ min) then
            Set min $\leftarrow f_i\left(p_i^{(t^{'})}, \Xi, \boldsymbol{\lambda}_i^{t_i}\right)$, and $p_i \leftarrow p_i^{(t^{'})}$.
        end if
    end for
end if
return $p_i$.
```

$$p_i^t = \frac{m_i \Xi_{\mathcal{N} \backslash \{i\}}}{(m - m_i) t_i} = \frac{\left(\lambda_i^t + \sigma \mu_i\right) \Xi_{\mathcal{N} \backslash \{i\}}}{\left((m - \sigma) \mu_i - \lambda_i^t\right) t_i}. \tag{58}$$

Assuming that the elements in $\mathcal{B}_i^{'}$ satisfy $p_i^{(0)} \leq p_i^{(1)} \leq \cdots \leq p_i^{(t_i+1)}$, where $p_i^{(0)} = \underline{p}$ and $p_i^{(t_i+1)} = \bar{p}_i$. If equation (57) holds, then there exists an integer $t^{'}$ $(0 \leq t^{'} \leq t_i)$ such that the optimal bidding price $p_i^* \in (p_i^{(t^{'})}, p_i^{(t^{'}+1)}) \subseteq \mathcal{P}_i \backslash \mathcal{B}_i^{'}$. In addition, from the derivations in Theorem 4.5, we know that

$$\frac{\partial^2}{\partial p_i^2} f_i\left(p_i, \Xi_{\mathcal{N}}, \boldsymbol{\lambda}_i^{t_i}\right) > 0, \tag{59}$$

for all $p_i \in \mathcal{P}_i \backslash \mathcal{B}_i^{'}$. Therefore, we can use a binary search method to search the optimal bidding price p_i^* in each of the sets $(p_i^{(t^{'})}, p_i^{(t^{'}+1)}) \subseteq \mathcal{P}_i \backslash \mathcal{B}_i^{'}$ $(0 \leq t_i^{'} \leq$

t_i), which satisfies (57). If we cannot find such a bidding price in $\mathcal{P}_i \backslash \mathcal{B}_i^{'}$, then the optimal bidding price p_i^* is in $\mathcal{B}_i^{'}$. The idea is formalized in Algorithm 2.

Given Ξ, $\boldsymbol{\lambda}_i^{t_i}$, and ϵ, where $\Xi = \sum_{j \in \mathcal{N}} p_j t_j$, $\boldsymbol{\lambda}_i^{t_i} = \left\{\lambda_i^t\right\}_{t \in \mathcal{T}_\rangle}$, and ϵ is a relatively small constant. Our optimal price bidding configuration algorithm to find p_i^* is given in Algorithm $Calculate p_i$. The key observation is that the first derivative of function $f_i\left(p_i, \Xi, \boldsymbol{\lambda}_i^{t_i}\right)$, i.e., $\frac{\partial}{\partial p_i} f_i\left(p_i, \Xi, \boldsymbol{\lambda}_i^{t_i}\right)$, is an increasing function in $p_i \in (p_i^{(t^{'})}, p_i^{(t^{'}+1)}) \subset \mathcal{P}_i \backslash \mathcal{B}_i^{'}$ (see (59)), where $0 \leq t^{'} \leq t_i$. Therefore, if the optimal bidding price is in $\mathcal{P}_i \backslash \mathcal{B}_i^{'}$, then we can find p_i^* by using the binary search method in one of the intervals $(p_i^{(t^{'})}, p_i^{(t^{'}+1)})$ $(0 \leq t^{'} \leq t_i)$ (Steps 3–17). In each of the search intervals $(p_i^{(t^{'})}, p_i^{(t^{'}+1)})$, we set ub as $(p_i^{(t^{'}+1)} - \epsilon)$ and lb as $(p_i^{(t^{'})} + \epsilon)$ (Step 4), where ϵ is relative small positive constant. If the first derivative of function $f_i\left(p_i, \Xi, \boldsymbol{\lambda}_i^{t_i}\right)$ on lb is positive or the first derivative on ub is negative, then the optimal bidding price is not in this interval (Step 5). Once the interval, which contains the optimal bidding price is decided, we try to find the optimal bidding price p_i^* (Steps 8–16). Notice that, the optimal bidding price may in $\mathcal{B}_i^{'}$ rather than in $\mathcal{P}_i \backslash \mathcal{B}_i^{'}$ (Step 19). Under this situation, we check each of the breakpoints in $\mathcal{B}_i^{'}$ and find the optimal bidding price (Steps 21–25).

By Algorithm 2, we note that the inner while loop (Steps 8–15) is a binary search process, which is very efficient and requires $\Theta\left(\log \frac{\bar{p}_{\max} - \underline{p}}{\epsilon}\right)$ to complete, where $\bar{p}_{\max}$ is the maximum upper bidding bound of all users, i.e., $\bar{p}_{\max} = \max_{i \in \mathcal{N}} (\bar{p}_i)$. Let $t_{\max} = \max_{i \in \mathcal{N}} (t_i)$, then the outer while loop (Steps 3–17) requires time $\Theta\left(t_{\max} \log \frac{\bar{p}_{\max} - \underline{p}}{\epsilon}\right)$. On the other hand, the for loop (Steps 21–25) requires $\Theta\left(t_{\max}\right)$ to find solution in set $\mathcal{B}_i^{'}$. Therefore, the time complexity of Algorithm 2 is $\Theta\left(t_{\max}\left(\log \frac{\bar{p}_{\max} - \underline{p}}{\epsilon} + 1\right)\right)$.

4.2.4 A Near-Equilibrium Price Bidding Algorithm

Notice that, the equilibrium bidding prices obtained by $\mathcal{IA}$ algorithm are considered under the condition that the allocated number servers can be fractional, i.e., in the computation process, we use

$$m_i = \frac{p_i t_i}{\sum_{j \in \mathcal{N}} p_j t_j} \cdot m, \tag{60}$$

instead of

$$m_i = \left\lfloor \frac{p_i t_i}{\sum_{j \in \mathcal{N}} p_j t_j} \cdot m \right\rfloor. \tag{61}$$

Therefore, we have to revise the solution and obtain a near-equilibrium price bidding strategy. Note that, under Eq. (61), there may exist some remaining servers, which is at most n. Considering for this, we reallocate the remaining servers according to the bidding prices. The idea is formalized in our proposed near-equilibrium price bidding algorithm ($\mathcal{NPBA}$), which characterizes the whole process.

Algorithm 6 $\mathcal{N}$ear-equilibrium $\mathcal{P}$rice $\mathcal{B}$idding $\mathcal{A}$lgorithm ($\mathcal{NPBA}$)

```
Input: $\mathcal{N}, \mathcal{P}, \boldsymbol{\lambda}_{\mathcal{N}}, \epsilon$.
Output: $\boldsymbol{p}_{\mathcal{N}}$.
 1: Set $\mathcal{S}_c \leftarrow \mathcal{N}$, $\mathcal{S}_l \leftarrow \emptyset$, and $k \leftarrow 0$.
 2: while ($\mathcal{S}_c \neq \mathcal{S}_l$) do
 3:     Set $\boldsymbol{p}_{\mathcal{N}} \leftarrow \mathbf{0}$, $\mathcal{S}_l \leftarrow \mathcal{S}_c$, $\boldsymbol{p}_{\mathcal{S}_c} \leftarrow \mathcal{IA}(\mathcal{S}_c, \boldsymbol{\lambda}_{\mathcal{S}_c}, \epsilon)$, and $\Xi \leftarrow \sum_{j \in \mathcal{N}} p_j t_j$.
 4:     for (each cloud user $i \in \mathcal{S}_c$) do
 5:         Compute the allocated servers as (61), i.e., calculate: $m_i \leftarrow \left\lfloor \frac{p_i t_i}{\Xi} \cdot m \right\rfloor$.
 6:     end for
 7:     Set $m_{\mathcal{R}} \leftarrow m - \sum_{i \in \mathcal{S}_c} m_i$, and $flag \leftarrow$ true.
 8:     while ($m_{\mathcal{R}} \neq 0$ and $flag =$ true) do
 9:         Set $flag \leftarrow$ false.
10:         for (each cloud user $i \in \mathcal{S}_c$) do
11:             Compute the reallocated servers, i.e., calculate: $m_i^t \leftarrow \left\lfloor \frac{p_i t_i}{\Xi} \cdot m_{\mathcal{R}} \right\rfloor$.
12:             if ($u_i \left( m_i + m_i^t, p_i, \boldsymbol{\lambda}_i^{t_i} \right) > u_i \left( m_i, p_i, \boldsymbol{\lambda}_i^{t_i} \right)$) then
13:                 Set $m_i \leftarrow m_i + m_i^t$, $m_{\mathcal{R}} \leftarrow m_{\mathcal{R}} - m_i^t$, and $flag \leftarrow$ false.
14:             end if
15:         end for
16:     end while
17:     for (each cloud user $i \in \mathcal{S}_c$) do
18:         if ($u_i(m_i, p_i, \boldsymbol{\lambda}_i^{t_i}) < v_i$) then
19:             Set $p_i \leftarrow 0$, and $\mathcal{S}_c \leftarrow \mathcal{S}_c - \{i\}$.
20:         end if
21:     end for
22: end while
23: return $\boldsymbol{p}_{\mathcal{N}}$.
```

At the beginning, the cloud provider sets a proper conservative bidding price ($\underline{p}$) and puts its value to into public information exchange module. Each cloud user i ($i \in \mathcal{N}$) sends his/her reserved time slots value (t_i) to the cloud provider. We denote the current set of cloud users who want to use cloud service as $\mathcal{S}_c$ and assume that in the beginning, all cloud users in $\mathcal{N}$ want to use cloud service, i.e., set $\mathcal{S}_c$ as $\mathcal{N}$ (Step 1). For each current user set $\mathcal{S}_c$, we calculate the optimal bidding prices for all users in $\mathcal{S}_c$ by $\mathcal{IA}$ algorithm, under the assumption that the allocated servers can fractional (Step 3). And then, we calculate their corresponding allocated servers (Steps 4–6). We calculate the remaining servers and introduce a $flag$ variable. The inner while loop tries to allocate the remaining servers according to the calculated bidding strategies of the current users in $\mathcal{S}_c$ (Steps 8–16). The variable $flag$ is used to flag whether there is a user in $\mathcal{S}_c$ can improve his/her utility by the allocated number of servers. The while loop terminates until the remaining servers is zero or there is no one such user can improve his/her utility by reallocating the remaining servers. For each user in $\mathcal{S}_c$, if his/her utility value is less than the reserved value, then we assume that he/she refuses to use cloud service (Steps 17–21). The algorithm terminates when the users who want to use cloud service are kept unchanged (Steps 2–22).

4.3 Performance Evaluation

In this section, we provide some numerical results to validate our theoretical analyses and illustrate the performance of the $\mathcal{NPBA}$ algorithm.

In the following simulation results, we consider the scenario consisting of maximal 200 cloud users. Each time slot is set as one hour of a day and the maximal time slots of a user can be 72. As shown in Table 2, the conservative bidding price ($\underline{p}$) is varied from 200 to 540 with increment 20. The number of cloud users (n) is varied from 50 to 200 with increment 10. The maximal bidding price ($\bar{p}_i$) and market benefit factor (r_i) of each cloud user are randomly chosen from 500 to 800 and 30 to 120, respectively. Each cloud user i ($i \in \mathcal{N}$) chooses a weight value from 0.1 to 2.5 to balance his/her time utility and profit. We assume that the request arrival rate (λ_i^t) in each time slot of each cloud user is selected randomly and uniformly between 20 and 480. The processing rate (μ_i) of a server to the requests from cloud user i ($i \in \mathcal{N}$) is randomly chosen from 60 to 120. For simplicity, the reservation value (v_i) and payment cost weight (δ_i) for each of the cloud users are set as zero and one, respectively. The number of servers m in the cloud provider is set as a constant 600, σ is set as 0.1, and ϵ is set as 0.01 (Table 3).

Figure 10 shows an instance for the bidding prices of six different cloud users versus the number of iterations of the proposed $\mathcal{IA}$ algorithm. Specifically, Fig. 10 presents the bidding price results of 6 randomly selected cloud users (users 8, 18, 27, 41, 59, and 96) with a scenario consisting of 100 cloud users. We can observe that the bidding prices of all users seem to be non-decreasing with the increase of iteration number and finally reach a relative stable state, which verifies the validness

Table 3 System parameters

System parameters	(Fixed)–[Varied range] (increment)
Conservative bidding price ($\underline{p}$)	(200)–[200, 540] (20)
Number of cloud users (n)	(100)–[50, 200] (10)
Maximal bidding price ($\bar{p}_i$)	[500, 800]
Market profit factor (r_i)	[30, 120]
Weight value (w_i)	[0.1, 2.5]
Request arrival rates (λ_i^t)	[20, 480]
Processing rate of a server (μ_i)	[60, 120]
Reserving time slots (t_i)	[1, 72]
Reservation value (v_i)	0
Payment cost weight (δ_i)	1
Other parameters (ϵ, σ, m)	(0.01, 0.1, 600)

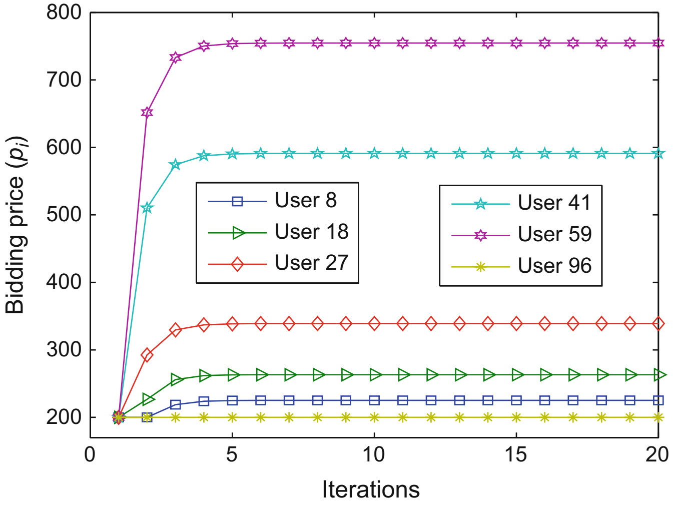

Fig. 10 Convergence process of bidding price

of Theorem 3.4. That is, the bidding prices of all cloud users keep unchanged, i.e., reach a Nash equilibrium solution after several iterations. In addition, it can also be seen that the developed algorithm converges to a Nash equilibrium very quickly. Specifically, the bidding price of each user has already achieved a relatively stable state after 5 iteration, which shows the high efficiency of our developed algorithm.

In Fig. 11, we show the trend of the aggregated payment from all cloud users (P_T), i.e., the revenue of the cloud provider, versus the increment of the conservative bidding price. We compare two kinds of results with the situations by computing the allocated number of servers for each cloud user i ($i \in \mathcal{N}$) as (60) and (61), respectively. Specifically, we denote the obtained payment as V_T when compute m_i as (60) and P_T for (61). Obviously, the former is the optimal value computed from the Nash equilibrium solution and bigger than that of the latter. However, it cannot be applied in a real application, because the allocated number of servers cannot be fractional. We just obtain a near-equilibrium solution by assuming that the allocated number of servers can be fractional at first. Even though the obtained solution is not optimal, we can compare these two kinds of results and show that how closer our proposed algorithm can find a near-equilibrium solution to that of the computed optimal one.

We can observe that the aggregated payment from all cloud users tends to increase with the increase of conservative bidding price at first. However, it decreases when conservative bidding price exceeds a certain value. The reason behind lies in that when conservative bidding price increases, more and more cloud users refuse to use the cloud service due to the conservative bidding price exceeds their possible maximal price bidding values or their utilities are less than their reservation values, i.e., the number of users who choose cloud service decreases (see

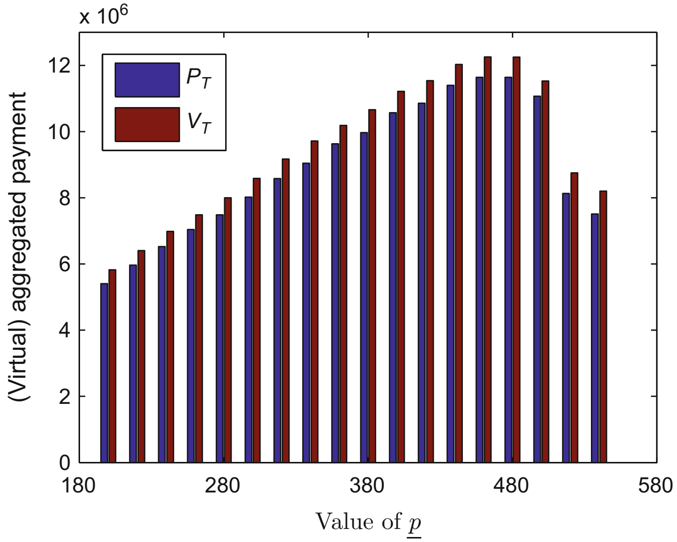

Fig. 11 Aggregated payment of all users

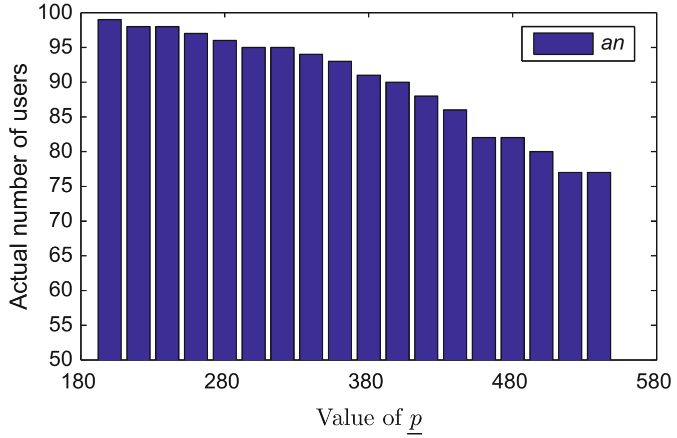

Fig. 12 Actual number of cloud users

Fig. 12). We can also observe that the differences between the values of P_T and V_T are relatively small and make little differences with the increase of the conservative bidding price. Specifically, the percent differences between the values of V_T and P_T range from 3.99% to 8.41%, which reflects that our $\mathcal{NPBA}$ algorithm can find a very well near-optimal solution while ignoring the increment of conservative bidding price. To demonstrate this phenomenon, we further investigate the specific utilities of some users and their corresponding bidding prices, which are presented in Figs. 13 and 14.

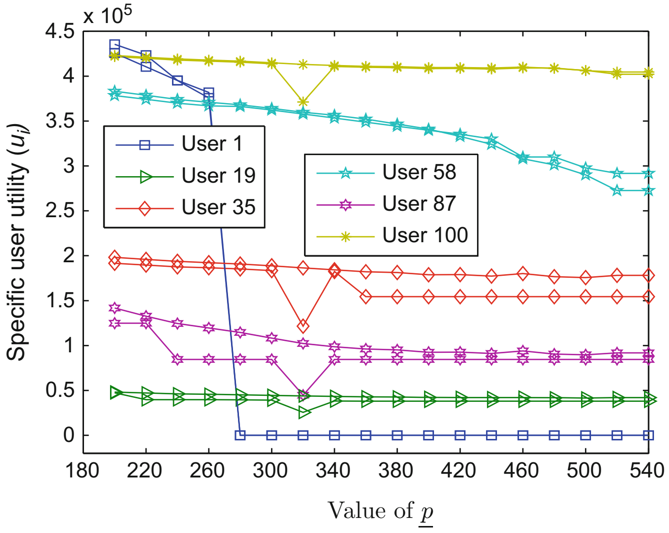

Fig. 13 Specific user utility

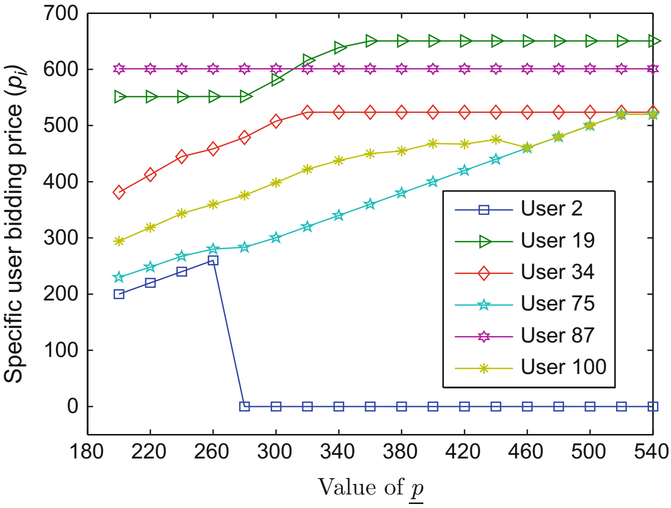

Fig. 14 Specific user bidding price

In Figs. 13 and 14, we plot the utility shape and the bidding prices of some cloud users for the developed $\mathcal{NPBA}$ algorithm. Figure 13 presents the utility shape under the developed algorithm versus the increment of conservative bidding price. We randomly select 6 users (users 1, 19, 35, 58, 87, and 100). It can be seen that the utility trends of all cloud users tend to decreases with the increase of conservative bidding price. However, under every conservative bidding price,

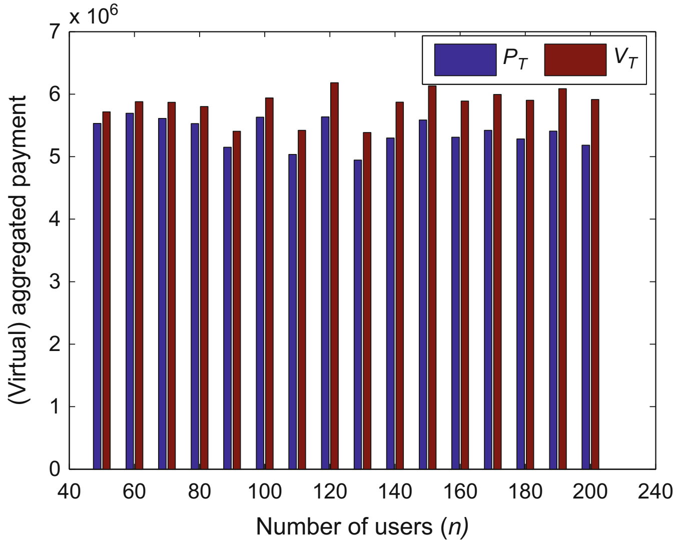

Fig. 15 Aggregated payment on number of users

for each user, the differences between the utilities computed by using m_i as (60) (the larger one) and (61) (the smaller one) for each cloud user are relatively small. Therefore, the differences between the aggregated payments of (P_T) and (V_T) are small (see Fig. 11). Figure 14 exhibits the corresponding bidding prices of the users shown in Fig. 13. We can observe that some users may refuse to use cloud service when conservative bidding price exceeds a certain value (user 2). When users choose to use cloud service, the treads of their bidding prices tend to be non-decreasing with the increment of conservative bidding price (user 19, 34, 75, 87, and 100). This phenomenon also verifies the aggregated payment trend shown in Fig. 11. Specifically, due to the increases of users' bidding prices, the aggregated payment from all cloud users tend to increase at first. However, when conservative bidding price exceeds a certain value, more and more cloud users refuse to use cloud service. Therefore, the aggregated payment tends to decrease when conservative bidding price is large enough.

In Fig. 15, we show the impact of number of cloud users on aggregated payment. Similar to Fig. 11, the differences between the values of P_T and V_T are relatively small. Specifically, the percent differences between the values of V_T and P_T range from 3.14% to 12.37%. That is, the aggregated payment results for different number of users are largely unchanged. In Fig. 16, we can observe that with the increase of number of cloud users, the trend of the differences between the number of cloud users and the actual number of cloud users who choose cloud service also increases. The reason behind lies in that with the increase of number of cloud users, more and more users refuse to use cloud service due to their utilities are less than their conservative values. This also partly verifies the aggregated payment trend shown in

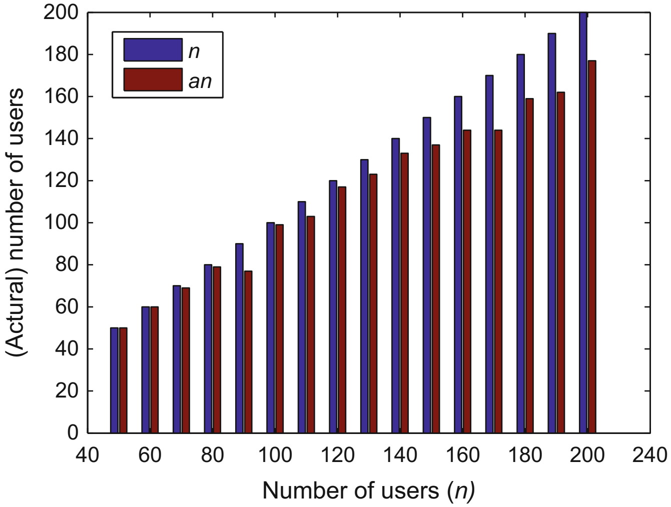

Fig. 16 (Actural) number of cloud users

Fig. 15, in which the aggregated payments are largely unchanged with the increase of number cloud users.

5 Conclusions

With the popularization of cloud computing and its many advantages such as cost-effectiveness, flexibility, and scalability, more and more applications are moved from local to cloud. However, most cloud providers do not provide a mechanism in which the users can configure optimizing strategies and decide whether to use the cloud service. To remedy these deficiencies, we focus on proposing a framework to obtain an appropriate strategy for each cloud user.

We try to enhance services in cloud computing by considering from multiple users' perspective. Specifically, we try to improve cloud services by simultaneously optimizing multiple users' utilities which involve both time and payment. We use game theory to analyze the situation and try to obtain a Nash equilibrium to simultaneously maximize multiple users' utilities. We prove the existence of Nash equilibrium and design two different approaches to obtain a Nash equilibrium for the two problems, respectively. Extensive experiments are also conducted, which verify our analyses and show the efficiencies of our methods.

References

1. Y.F.B. Li, B. Li, Price competition in an oligopoly market with multiple IaaS cloud providers. IEEE Trans. Comput. **63**(1), 59–73 (2014)
2. R. Pal, P. Hui, Economic models for cloud service markets: pricing and capacity planning. Theor. Comput. Sci. **496**, 113–124 (2013)
3. P.D. Kaur, I. Chana, A resource elasticity framework for QoS-aware execution of cloud applications. Futur. Gener. Comput. Syst. **37**, 14–25 (2014)
4. N.B. Rizvandi, J. Taheri, A.Y. Zomaya, Some observations on optimal frequency selection in DVFS-based energy consumption minimization. CoRR, abs/1201.1695 (2012)
5. R. Cohen, N. Fazlollahi, D. Starobinski, Path switching and grading algorithms for advance channel reservation architectures. IEEE/ACM Trans. Netw. **17**(5), 1684–1695 (2009)
6. S. Son, K.M. Sim, A price- and-time-slot-negotiation mechanism for cloud service reservations. IEEE Trans. Syst. Man Cybern. Part B Cybern. **42**(3), 713–728 (2012)
7. J. Cao, K. Hwang, K. Li et al., Optimal multiserver configuration for profit maximization in cloud computing. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1087–1096 (2013)
8. S. Zaman, D. Grosu, Combinatorial auction-based allocation of virtual machine instances in clouds. J. Parallel Distrib. Comput. **73**(4), 495–508 (2013)
9. P. Samimi, Y. Teimouri, M. Mukhtar, A combinatorial double auction resource allocation model in cloud computing. Inf. Sci. **357**(357), 201–216 (2014)
10. T.T. Huu, C.K. Tham, An auction-based resource allocation model for green cloud computing, in *Proceedings of Cloud Engineering (IC2E), 2013 IEEE International Conference on* (2013), pp. 269–278
11. A.H. Mohsenian-Rad, V.W. Wong, J. Jatskevich et al., Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid. IEEE Trans. Smart Grid **1**(3), 320–331 (2010)
12. Chen H, Li Y, Louie R et al., Autonomous demand side management based on energy consumption scheduling and instantaneous load billing: an aggregative game approach. IEEE Trans. Smart Grid **5**(4), 1744–1754 (2014)
13. Z. Fadlullah, D.M. Quan, N. Kato et al., GTES: an optimized game-theoretic demand-side management scheme for smart grid. IEEE Syst. J. **8**(2), 588–597 (2014)
14. H. Soliman, A. Leon-Garcia, Game-theoretic demand-side management with storage devices for the future smart grid. IEEE Trans. Smart Grid 5(3):1475–1485 (2014)
15. I. Atzeni, L.G. Ordóñez, G. Scutari et al., Noncooperative and cooperative optimization of distributed energy generation and storage in the demand-side of the smart grid. IEEE Trans. Signal Process. **61**(10), 2454–2472 (2013)
16. M. Rahman, R. Rahman, CAPMAuction: reputation indexed auction model for resource allocation in Grid computing, in *Proceedings of Electrical Computer Engineering (ICECE), 2012 7th International Conference on* (2012), pp. 651–654
17. A. Ozer, C. Ozturan, An auction based mathematical model and heuristics for resource co-allocation problem in grids and clouds, in *Proceedings of Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control, 2009. ICSCCW 2009. Fifth International Conference on* (2009), pp. 1–4
18. X. Wang, X. Wang, C.L. Wang et al., Resource allocation in cloud environment: a model based on double multi-attribute auction mechanism, in *Proceedings of Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on* (2014), pp. 599–604
19. X. Wang, X. Wang, H. Che et al., An intelligent economic approach for dynamic resource allocation in cloud services. IEEE Trans. Cloud Comput. PP(99):1–1 (2015)
20. G. Scutari, D. Palomar, F. Facchinei et al., Convex optimization, game theory, and variational inequality theory. IEEE Signal Process. Mag. **27**(3), 35–49 (2010)
21. M.J. Osborne, A. Rubinstein, *A Course in Game Theory* (MIT Press, Cambridge, 1994)
22. J.P. Aubin, *Mathematical Methods of Game and Economic Theory* (Courier Dover Publications, New York, 2007)

23. S.S. Aote, M. Kharat, A game-theoretic model for dynamic load balancing in distributed systems, in *Proceedings of the International Conference on Advances in Computing, Communication and Control* (ACM, 2009), pp. 235–238
24. N. Li, J. Marden, Designing games for distributed optimization, in *Proceedings of Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on* (2011), pp. 2434–2440
25. E. Tsiropoulou, G. Katsinis, S. Papavassiliou, Distributed uplink power control in multiservice wireless networks via a game theoretic approach with convex pricing. IEEE Trans. Parallel Distrib. Syst. **23**(1), 61–68 (2012)
26. G. Scutari, J.S. Pang, Joint sensing and power allocation in nonconvex cognitive radio games: Nash equilibria and distributed algorithms. IEEE Trans. Inf. Theory **59**(7), 4626–4661 (2013)
27. N. Immorlica, L.E. Li, V.S. Mirrokni et al., Coordination mechanisms for selfish scheduling. Theor. Comput. Sci. **410**(17), 1589–1598 (2009)
28. S. Penmatsa, A.T. Chronopoulos, Game-theoretic static load balancing for distributed systems. J. Parallel Distrib. Comput. **71**(4), 537–555 (2011)
29. K. Li, C. Liu, K. Li, An approximation algorithm based on game theory for scheduling simple linear deteriorating jobs. Theor. Comput. Sci. **543**, 46–51 (2014)
30. N. Mandayam, G. Editor, S. Wicker et al., Game theory in communication systems [Guest Editorial]. IEEE J. Select. Areas Commun. **26**(7), 1042–1046 (2008)
31. E. Larsson, E. Jorswieck, J. Lindblom et al., Game theory and the flat-fading gaussian interference channel. IEEE Signal Process. Mag. **26**(5), 18–27 (2009)
32. C. Liu, K. Li, C. Xu et al., Strategy configurations of multiple users competition for cloud service reservation. IEEE Trans. Parallel Distrib. Syst. **27**(2), 508–520 (2016)
33. P. Samadi, H. Mohsenian-Rad, R. Schober et al., Advanced demand side management for the future smart grid using mechanism design. IEEE Trans. Smart Grid **3**(3), 1170–1180 (2012)
34. I. Atzeni, L. Ordonez, G. Scutari et al., Demand-side management via distributed energy generation and storage optimization. IEEE Trans. Smart Grid 4(2):866–876 (2013)
35. J. Cao, K. Li, I. Stojmenovic, Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers. IEEE Trans. Comput. **63**(1), 45–58 (2014)
36. S. Boyd, L. Vandenberghe, *Convex Optimization* (Cambridge University Press, Cambridge, 2009)
37. G. Scutari, D. Palomar, F. Facchinei et al., Monotone games for cognitive radio systems, in *Proceedings of Distributed Decision Making and Control*, ed. by R. Johansson, A. Rantzer (Springer, London, 2012), pp. 83–112
38. Altman E, Basar T, Jimenez T et al., Competitive routing in networks with polynomial costs. IEEE Trans. Autom. Control **47**(1), 92–96 (2002)
39. K. Akkarajitsakul, E. Hossain, D. Niyato, Distributed resource allocation in wireless networks under uncertainty and application of Bayesian game. IEEE Commun. Mag. **49**(8), 120–127 (2011)
40. S. Misra, S. Das, M. Khatua et al., QoS-guaranteed bandwidth shifting and redistribution in mobile cloud environment. IEEE Trans. Cloud Comput. **2**(2), 181–193 (2014)