

Profit Maximization for Cloud Brokers in Cloud Computing

Jing Mei¹, Kenli Li¹, *Senior Member, IEEE*, Zhao Tong², Qiang Li, and Keqin Li³, *Fellow, IEEE*

Abstract—Along with the development of cloud computing, more and more applications are migrated into the cloud. An important feature of cloud computing is pay-as-you-go. However, most users always should pay more than their actual usage due to the one-hour billing cycle. In addition, most cloud service providers provide a certain discount for long-term users, but short-term users with small computing demands cannot enjoy this discount. To reduce the cost of cloud users, we introduce a new role, which is *cloud broker*. A cloud broker is an intermediary agent between cloud providers and cloud users. It rents a number of reserved VMs from cloud providers with a good price and offers them to users on an on-demand basis at a cheaper price than that provided by cloud providers. Besides, the cloud broker adopts a shorter billing cycle compared with cloud providers. By doing this, the cloud broker can reduce a great amount of cost for user. In addition to reduce the user cost, the cloud broker also could earn the difference in prices between on-demand and reserved VMs. In this paper, we focus on how to configure a cloud broker and how to price its VMs such that its profit can be maximized on the premise of saving costs for users. Profit of a cloud broker is affected by many factors such as the user demands, the purchase price and the sales price of VMs, the scale of the cloud broker, etc.. Moreover, these factors are affected mutually, which makes the analysis on profit more complicated. In this paper, we first give a synthetically analysis on all the affecting factors, and define an optimal multiserver configuration and VM pricing problem which is modeled as a profit maximization problem. Second, combining the partial derivative and bisection search method, we propose a heuristic method to solve the optimization problem. The near-optimal solutions can be used to guide the configuration and VM pricing of the cloud broker. Moreover, a series of comparisons are given which show that a cloud broker can save a considerable cost for users.

Index Terms—Cloud broker, cloud computing, cost reduction, profit maximization, queue model, service demand, VM configuration, VM pricing

1 INTRODUCTION

OVER the past few years, cloud computing has experienced tremendous development [1]. More and more cloud providers have jumped on the cloud bandwagon, and they centrally manage a variety of resources such as hardware and software and deliver them over the internet in the form of services to customers on demand [2]. Thanks to unique properties such as elasticity, flexibility, apparently unlimited computational power [3], and pay-as-you-use pricing model, cloud computing can reduce the requirement of clients for large capital outlays for hardware necessary to deploy service and the human expenses to operate

it [4]. Hence, an increasing number of clients are transferring their business to the cloud.

One important feature of cloud computing is pay-as-you-use [5], [6], [7], which contains two meanings. First, according to the customer resource demand such as CPU, memory, etc., the physical machines are dynamically segmented using virtualization technologies and provided to customers in the form of virtual machines (VMs), and customers pay according to the amount of resources they actually consumed. Second, the VMs can be dynamically allocated and deallocated at any time, and customers should pay based on how long the resources are actually used. Nevertheless, the pay-as-you-use pricing model is presently only conceptual due to the extreme complexity in monitoring and auditing resource usage [8], and cloud providers usually adopt an hourly billing scheme; in other words, the Billing Time Unit (BTU) of the cloud providers is one hour, for instance, Amazon EC2 [9]. Therefore, the customers should pay for the resources by the hour even if they do not actually utilize the allocated resources in the whole billing horizon [10]. This leads to a waste of resources and raises the cost of customers to a certain degree.

In addition, almost all cloud providers provide two main ways to pay for their instances: On-Demand and Reserved Instances [11], [12]. With On-Demand instances, users pay for compute capacity by per hour depending on which instances they run, and they are recommended for the applications with short-term workloads. Reserved Instances

- J. Mei, Z. Tong, and Q. Li are with the ed HPCSIP, Ministry of Education of China, College of Information Science and Engineering, Hunan Normal University, Changsha, Hunan 410081, China. E-mail: jingmei1988@163.com, {tongzhao, liqiang}@hunnu.edu.cn.
- K. Li is with the College of Information Science and Engineering, and National Supercomputing Center in Changsha, Hunan University, Changsha, Hunan 410082, China. E-mail: lkl@hnu.edu.cn.
- K. Li is with the College of Information Science and Engineering, and National Supercomputing Center in Changsha, Hunan University, Changsha, Hunan 410082, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561. E-mail: lik@newpaltz.edu.

Manuscript received 19 Sept. 2017; revised 19 June 2018; accepted 22 June 2018. Date of publication 28 June 2018; date of current version 12 Dec. 2018. (Corresponding author: Kenli Li.)

Recommended for acceptance by B. He.

For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.

Digital Object Identifier no. 10.1109/TPDS.2018.2851246

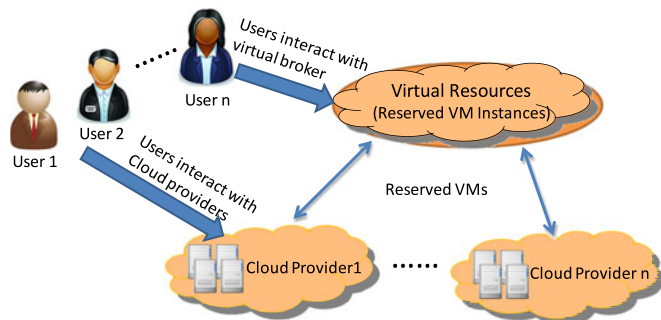


Fig. 1. The cloud broker.

provide users with a significant discount (up to 75 percent in Amazon EC2) compared to On-Demand instance pricing, but customers should rent instances for long periods, e.g., from six months to several years, according to the current plans offered by real cloud providers such as Amazon [9] and Microsoft Azure [13]. Obviously, this discount cannot be enjoyed by the short-term customers.

Due to above two reasons, the short-term customers always should pay more than they actually must pay. To reduce cost for this part of customers, we introduce the *cloud broker*, an intermediary agent between cloud providers and customers. Fig. 1 shows the relationship among the cloud broker, cloud providers, and customers.

The cloud broker rents the reserved VMs from cloud providers for long periods with the reserved price and outsources the resources as on-demand VMs to customers for a lower price with respect to the price that the cloud providers charge for the same VMs. A cloud broker can help to reduce the cost of customers from two aspects. First, the cloud broker takes advantages of the price gap between reserved and on-demand VMs, renting the reserved VMs with a good price and outsourcing them as on-demand VMs with a lower price compared with the same VMs provided by cloud providers. Second, the cloud broker adopts a smaller billing cycle (BTU) than the cloud providers. Adopting the two strategies, the resource utilization can be efficiently increased and the customer requests can be accommodated with less cost.

In addition to helping customers to reduce their cost, the cloud broker can earn a huge difference in price between the reserved and on-demand VMs [14]. Making profit is one of the main objectives of all enterprises. Hence, in this paper, we focus on how to maximize the profit of the cloud broker, and meanwhile, the customer cost can be reduced efficiently.

Like all business, the profit model of a cloud broker in cloud computing is based on two components, namely, the revenue and the cost. For a cloud broker, the revenue is the service charge to users, and the cost is the renting cost paid to cloud service providers. A profit model of a cloud broker includes many considerations, such as the scale (the number of VMs) of a cloud broker system, the customer demand (the rate that requests submitted to a cloud broker), the renting price (the cost price) that the resources are rented from cloud providers, the selling price (the sales price) that the cloud broker provides resources to users, the BTU, and so forth. To maximize the profit of a cloud broker, we should understand both revenue and cost, and in particular, how they are affected by those factors.

The revenue of a cloud broker is determined by two factors, i.e., the customer demand and the sales price. The customer demand is measured by the task arrival rate of the cloud broker in this paper. Under a given sales price, the greater (smaller, respectively) the customer demand is, the higher (lower, respectively) the revenue is. Similarly, under a given customer demand, the higher (lower, respectively) the sales price goes, the more (less, respectively) the revenue can be obtained. Moreover, the sales price has a great impact on the customer demand of a cloud broker. If the sales price of the on-demand VMs offered by the cloud broker are much cheaper compared with the same VMs provided by the cloud providers, more customers are attracted to submit their computing requests to the cloud broker. On the contrary, if the cloud broker raises the sales price of VMs, the customer demand decreases correspondingly. Hence, determining a proper sales price is a key issue for cloud brokers to maximize their profit, which will be calculated in this paper.

The cost of a cloud broker is also determined by two factors, i.e., the cost price of resources and the scale of the service system. The cost price of resources is determined by cloud providers. The service system can be modeled as a multiserver system, which consists of many resources (VMs) rented from cloud providers. The system scale determines the service capacity of the cloud broker. A cloud broker with a larger system scale can serve more customers, which can obtain more revenue but generate an increasing cost. Hence, the system scale also should be determined properly such that the profit of a cloud broker is maximized.

In this paper, we study the problem of optimal multi-server configuration and resource pricing for profit maximization of cloud brokers. To maximize the profit of cloud brokers, we provide a comprehensive analysis on the profit-affecting factors and formulate a profit maximization problem. By solving the optimization problem, the optimal VM price and system scale can be obtained such that the profit is maximized. Our main contributions are as follows.

- To reduce the cost of cloud users, a novel business role between cloud providers and cloud users, i.e., cloud broker, is introduced.
- A cloud broker is treated as a multiserver system, which is modeled as an $M/M/n/n$ queuing model. Based on this model, all the profit-affecting factors are analyzed.
- A detailed analysis on the relationship between the sales price of VMs and the customer demand is given. Based on the analysis, the expected charge to a VM request is calculated.
- The optimal multiserver configuration and VM pricing problem of cloud brokers for profit maximization is formulated and a heuristic algorithm combining a brute force search with the partial derivation method is proposed to calculate the numerical solutions for the optimization problem.
- A series of numerical calculations are conducted, which show that the cloud broker can reduce the cost for cloud users efficiently and yet make a considerable profit at the same time.

The rest of the paper is organized as follows. Section 3 presents the models used in this paper, including the cloud

broker model, the multiserver system model, the revenue and cost model. Based on these models, the optimal multiserver configuration and VM pricing problem is defined and the profit optimization model is formulated. Section 4 introduces our methods to solve the optimization problem to obtain the optimal decision on the VM sales price and the system scale. Section 5 conducts a series of numerical calculations to demonstrate the results of our problem. Finally, Section 6 concludes the work.

2 RELATED WORKS

In this section, we provide a snapshot of the existing research from the following aspects.

2.1 Cloud Broker as a Scheduler

Nowadays, there are numerous private and public cloud providers that typically provide many services. Since different providers usually offer distinct features, e.g., Virtual Machine (VM) types, pricing schemes, and cloud interfaces, it is becoming challenging for users to find a choice that better suits the requirements for developing/executing their applications. To assist cloud users, a cloud broker mechanism is used to transform the heterogeneous cloud market into a commodity-like service [14].

The cloud broker has been studied from different perspectives. In the beginning, the cloud broker was studied as a scheduler between cloud providers and customers. It refers to two main aspects: helping customers to select the most appropriate cloud provider and helping providers make decisions on resource allocation. Hence, the scheduling mechanisms are required to optimize the selection of cloud broker or placement of VMs amongst multiple data centers of a cloud to reduce the costs of VM deployment or satisfy other performance constraints such as response time, computing capacity, and so forth [15], [16], [17], [18]. To achieve the different objectives, many related cloud broker policies have been proposed.

Limhani et al. [16] proposed a cost-aware service proximity based broker policy. Using the proposed policy, a cost effective data center is selected to route user requests. In [19], the authors proposed a new service broker policy for data center selection based on the round-robin (RR) algorithm to minimize the service response time. To satisfy different resource requirements and application performance constraint of customers, Manasrah et al. [17] proposed a Variable Service Broker Routing Policy—VSBRP, which aims to achieve the minimum response time through considering the communication channel bandwidth, latency and the size of the job. The proposed service broker policy can also reduce the overloading of the data centers by redirecting the user requests to the next data center that yields better response and processing time. Larumbe et al. [18] took energy consumption into consideration and proposed an energy-aware VM placing broker to minimize operational expenditures while respecting constraints on Quality of Service (QoS), power consumption, and CO₂ emissions. In addition, many other existing broker policies for data center selection are based on the location of the data centers, current execution load, and so on. The above studies on broker policies focused on how to allocate resources for each request.

Since there are so many cloud providers with different features, it becomes a challenge for customers to select the one that suits the requirements while with the least costs. Many studies have focused on this problem. Rane et al. [20] researched how to choose one provider for a customer among many service providers. In this work, a cloud resource broker is proposed to govern the assignment of providers' resources to consumers dynamically. It uses various requirements and constraints specified by the consumer in the requirement description template as inputs, to calculate aggregated requirements using an aggregation algorithm. And a service scheduling algorithm is defined to find an optimized match between the aggregated requirements with the provider's offerings.

With the help of cloud brokers, the VM requests of a customer can be allocated in different clouds instead of the same one [21]. Tordsson et al. [22] took into account user requirements such as hardware configuration, aggregated service performances, total cost, and load balancing, and proposed algorithms for optimized placement of VMs in multi-cloud environments. The authors considered the VM placement problem as a 0-1 integer programming (IP) problem, and the total infrastructure capacity and the total cost of the deployed VMs are formulated. The modeling language AMPL is used to solve the 0-1 IP problem. The experimental results confirm that the multi-cloud deployment provides better performance and lower costs compared to the use of a single cloud only.

2.2 Cloud Broker as a Company

Along with the development of cloud computing, the role of the cloud broker has changed. It moves from the role of a scheduler between cloud providers and customers to the third-party company that provides cloud computing services. The difference between cloud brokers and cloud providers is a cloud broker might not have its own resources but rents them from cloud providers.

To maintain a company's normal operation, making a profit is necessary. How to increase the profit of the cloud broker becomes an important problem and it is researched from different perspectives in many works.

The services requests submitted by customers are characterized by different requirements such as security and privacy constraints, the required resources amount, the price and makespans. To improve profit, the cloud broker should properly allocate the requested services to the best suited cloud infrastructures based on the customers' QoS requirements.

The cloud broker studied in [23] is a hybrid cloud broker; that is, the cloud broker can allocate a service to its private resources or the public clouds. In the paper, the revenue of a CB consists of the brokering service price and the resource provisioning price. If the service is executed on the private resources of the CB, CB revenue is the sum of the two parts; otherwise, the CB only gets the brokering service price, and the resource provisioning price is paid to the public clouds. Hence, a greedy way to increase the total revenue of the cloud broker is to maximize the in-house execution of all services. Due to the limited number of in-house resources and the specific QoS requirements of services, the greedy method is unsuitable, so the heuristic brokering algorithm is proposed in which three allocation patterns, namely,

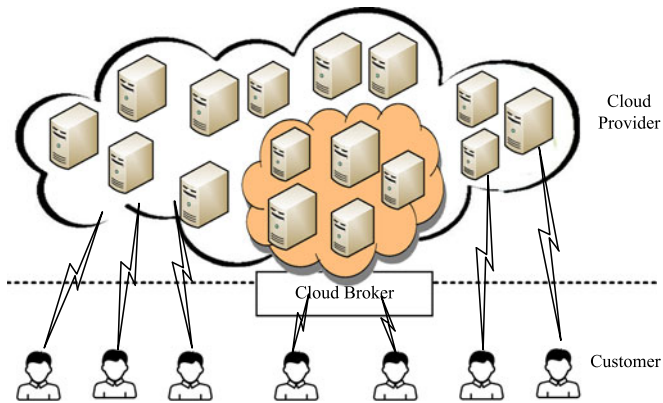


Fig. 2. The three-tier cloud structure.

Feasible, static Reservation, and Max Occupation, are adopted to allocate particularly critical services to private resources to maximize the CB revenue.

Nesmachnow et al. [3] proposed a kind of Virtual Cloud Brokers (VCB). The VCB rents a number of reserved instances of different VMs from several cloud providers for long periods of time and outsources them as on-demand VMs for a lower price with respect to what traditional cloud providers charge for the same VMs. The VCB earns the large difference in price between reserved VMs and on-demand VMs. Because the reserved instances bought by the VCB are limited, in case it cannot fulfill all requests without violating the contracted Service Level Agreement (SLA), on-demand VMs are bought from public cloud providers to satisfy the demand, which leads to a reduction in profit. This paper researches how to manage the available resources efficiently to process the most VM requests. This problem is a resource allocation problem with additional constraints which is NP-hard. The authors build this problem as a profit maximization model with resource constraints and propose a series of fast optimization techniques to solve this problem, including two-step list scheduling heuristics to get the initial solutions and the fast reordering local search to improve the solutions.

Because VM demand is sporadic, a cloud broker has to rent VMs once it faces the risk of underutilization of the VM in subsequent time slots. However, if the VM demand decreases in the following time slot, a great number of VMs are wasted and the cloud broker might suffer a loss. To improve the profit of cloud brokers, the authors in [24] adopted dynamic pricing to control user demand. At the beginning of each time slot, the broker adjusts the VM selling price according to the VM demand. If the VM demand is much greater than the previous time slot, the price is raised to decrease the demand properly. Doing this can efficiently control the loss caused by the wasted VMs.

All these works focused on how to improve the profit of the cloud brokers under a given configuration. However, for a cloud broker, to achieve the maximal profit, the most important problem is determining how many resources it needs to rent and how to price the resources. Since there are many factors that can affect the profit of cloud brokers, and these profit-affecting factors are affected mutually, it is necessary to provide a comprehensive analysis on these factors and take them into consideration when solving the broker configuration and pricing problem. Based on this idea, the profit-affecting factors, e.g., the resource renting price, the

resource selling price, the customer demand, the resource size, are analyzed comprehensively, and a profit maximization problem is formulated and solved to get the optimal configuration of the virtual platform and the optimal price of the resources.

3 THE MODELS

In this section, we first describe the cloud structure. Then, we introduce the related models used in this paper, e.g., a multiserver queuing model, a revenue model, and a cost model. Last, we give a detailed description on the optimal multiserver configuration and VM pricing problem for profit optimization.

3.1 Cloud Computing Structure

In the cloud structure (see Fig. 2), three typical parts are contained, i.e., cloud service providers, cloud brokers, and customers.

In the cloud market, there are various cloud service providers with distinct features such as capacity, price, SLA, and performance. Customers can obtain services and resources from cloud providers directly. However, it is a challenge for customers to find the best choice in terms of performance and price. In addition, the economic model of the cloud providers is to bill users solely for the time they have used the resources based on an atomic time unit that we call the Billing Time Unit, most often one hour. However, many customers might use the resources for only several minutes and still be charged for one hour. Hence, the coarse-grained BTU leads to a lot of waste for customers in terms of resources or money.

Moreover, many cloud providers, such as Amazon EC2 and Window Azure, provide on-demand instances and reserved instances. With on-demand instances, you pay for compute capacity by the hour with no long-term commitments or upfront payments. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified hourly rate for the instances you use. Reserved instances provide you with a significant discount (up to 75 percent) compared to on-demand instance pricing [9]. In this paper, the price of reserved instances and on-demand instances for unit of time is denoted as β_{re} and β_{od} (Unit: dollars per unit time), respectively. For a part of customers, they only need to rent on-demand instances due to their short-term workloads, hence, they cannot enjoy the discount of reserved instances.

The cloud broker is an intermediary entity between cloud providers and customers, which emerges to help the customers with short-term workloads enjoying the discount provided for long-term customers. It buys a lot of reserved instances from cloud providers for long periods of time to configure its virtual resource platform and outsources them as on-demand VMs for a lower price and a fine-grained BTU such as 30 minutes with respect to what the cloud service providers charge for the same VMs. The customers could submit their service requests to the cloud provider or the cloud broker, and their decisions are affected by the gap between the on-demand VM prices of the cloud broker and the cloud provider.

This three-tier structure is adopted and researched commonly from different aspects [3], [14]. In this paper, we

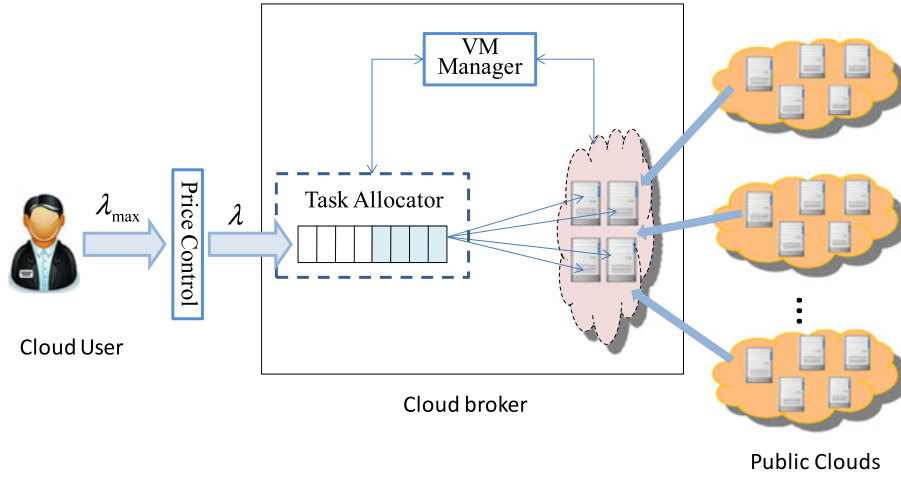


Fig. 3. The $M/M/n/n$ queue model.

focus on the profit maximization problem of the cloud broker.

3.2 Multiserver Queue System

The broker studied in this paper only rents resources from a single cloud provider and provides identical VMs for customers. Therefore, the VMs provided by the cloud broker are homogeneous, and they have identical configurations in terms of memory, bandwidth, CPU, etc. More complicated situations will be studied in further works. In this paper, we assume that a cloud broker serves users' requests by using a multiserver system, which can be modeled as an $M/M/n/n$ queuing system as Fig. 3 [25]. This similar models are adopted in many literatures such as [26], [27], [28].

In this $M/M/n/n$ queuing model, the arrival of VM requests is assumed to be a Poisson stream with arrival rate λ (measured by the number of requests per unit of time); i.e., the interarrival times are independent and identically distributed (i.i.d.) exponential random variables with mean $1/\lambda$. In the rest of the paper, the default time unit is one hour (unless explicitly stated). This setting is for convenience of calculation under different BTU values. Since the cloud broker attracts customers by the low price, the actual request arrival rate λ of the cloud broker is determined by two factors: the total customer demand, denoted by λ_{max} , and the resource price. The relationship will be introduced in Section 3.4.

The cloud broker rents VMs from cloud providers to configure its private cloud platform. Assume that the platform size, in others words, the number of VMs owned by the cloud broker is n . When customers submit service requests to the cloud broker, the cloud broker determines whether the requests are executed in-house or reassigned to public clouds according to the status of its private cloud. If there are available VMs in the private cloud, the incoming service requests will be executed in-house. However, if there is not any VM available and the incoming service requests cannot be processed immediately in the private cloud, the cloud broker will resubmit them to public clouds. These part of customers are lost by the cloud broker. In this scenario, the number of requests in the mutliserver system at any time will not exceed n , that is, the queue length of the mutliserver system is n . The execution times of tasks on the mutliserver

system are i.i.d. exponential random variables t with mean \bar{t} . The average service rate of each system is calculated as $\mu = 1/\bar{t}$, and the server utilization is defined as $\rho = \lambda/n\mu = \lambda/n \times \bar{t}$. Let π_k be the probability that there are k ($k \leq n$) service requests (being processed) in the $M/M/n/n$ queuing system. Then, we have

$$\pi_k = \pi_0 \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k, \quad k = 1, 2, \dots, n, \quad (1)$$

where

$$\pi_0 = \left[\sum_{k=0}^n \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k \right]^{-1},$$

and Eq. (1) holds only when $\rho < 1$ [25].

Because the number of resources is limited, when and only when all of the resources in the multiserver system are busy, the incoming requests are resubmitted to another cloud. Hence, this part of requests are lost to the cloud broker. Hence, the loss probability P_L of customers of the cloud broker is equal to the probability that there are n requests in the system, which is calculated as [25]

$$P_L = \pi_n = \left[\sum_{k=0}^n \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k \right]^{-1} \frac{1}{n!} \left(\frac{\lambda}{\mu} \right)^n, \quad (2)$$

and the number of loss customers in unit time is

$$\lambda_L = \lambda P_L = \lambda \left[\sum_{k=0}^n \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k \right]^{-1} \frac{1}{n!} \left(\frac{\lambda}{\mu} \right)^n. \quad (3)$$

3.3 Cost Modeling

The cost of a cloud broker consists of many parts such as management cost, configuration cost, etc. However, we only consider the cost conducted by configuring and operating the virtual multiserver platform in this paper. To configure the multiserver platform, a cloud broker rents a lot of reserved instances from cloud providers for a long-term period, and pays them the corresponding rents. Since the rental price per reserved instance for unit of time is β_{re} and n reserved instances are required to configure the

multiserver platform. Then, the cost per unit of time of the cloud broker is

$$C = n\beta_{re}.$$

3.4 Revenue Modeling

3.4.1 Analysis on the Revenue-Affecting Factors

As previously noted, the cloud broker buys a lot of reserved instances from cloud providers for long periods of time and outsources them as on-demand VMs to obtain revenue. The on-demand VMs provided by the cloud broker have a lower price and a fine-grained BTU with respect to what the cloud providers charge for the same VMs. Hence, there are two main factors affecting the revenue of cloud brokers.

The first revenue-affecting factor is customer demand, which is measured as the request arrival rate λ . Under a fixed price, the more the request arrival rate is, the more revenue that can be obtained. Hence, to improve the revenue of a cloud broker, an obvious way is to increase its customer demand. However, customer demand is changing with different VM sales prices. Hence, the second affecting factor is the VM sales price.

Let the price of the on-demand VMs provided by the cloud broker be β per unit of time. The price affects the revenue of a cloud broker from two aspects. First, the price has a direct impact on revenue. Under a given demand, a higher price conducts a higher revenue. Second, the price affects the revenue indirectly. The explanations are given as follows. The cloud broker rents reserved instances from cloud providers with a discount compared with the on-demand instances and outsources them as on-demand VMs in a lower price than the same VMs provided by cloud providers. The low price is the core competitive advantage of the cloud broker, and its objective customers are those customers whose service requests are submitted occasionally and the execution time is uncertain or short. This portion of customers are inclined to rent on-demand VMs rather than reserved VMs, but they also want to enjoy the discount that the cloud providers provide for long-term customers. The cloud broker can provide customers the needed resources at a lower price. Since the main advantage for the cloud broker to attract customers is its lower price compared with public clouds, the price certainly will affect the request arrival rate, thus affecting revenue, corresponding. Hence, proper pricing is an important issue for the cloud broker.

To obtain profit, the VM sales price of the cloud broker should be greater than its cost price obviously; that is, the rental price that the cloud broker rents reserved instances from cloud providers. Meanwhile, the VM sales price should be lower than the on-demand price of cloud providers to attract customers. That is because customers are inclined to select the services of public clouds when the VM sales price of the cloud broker is same as public clouds. To sum up, the VM sales price of the broker, denoted as β , should be between the range of $[\beta_{re}, \beta_{od}]$.

To determine a proper sales price for VMs, it is necessary to understand the relationship between the VM sales price and the customer demand. In general, the greater the gap between the on-demand price β_{od} and the VM sales price of the cloud broker β is, the more customers that will be attracted. Let the total customer demand be λ_{max} . And we

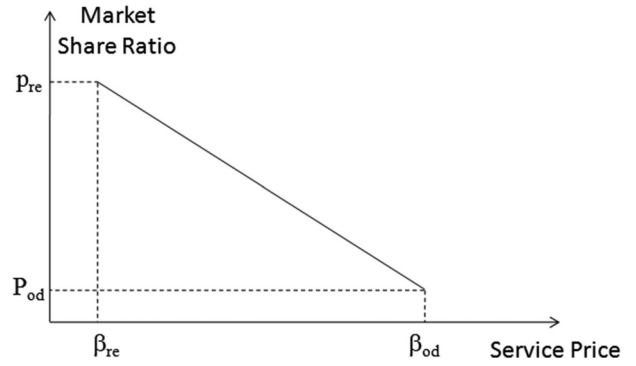


Fig. 4. The relationship between price and customer demand.

define the ratio of the actual customer demand of a cloud broker and the total customer demand as “market share ratio”, which is denoted by p . Fig. 4 shows the relationship between the price β and the market share ratio p . The price-demand function in Fig. 4 is a linear demand curve which means the percentage p is linearly decreasing with the increasing price β . From Fig. 4, the market share ratio of a cloud broker p is equal to p_{re} when the VM sales price is set as β_{re} , and the market share ratio is decreasing to p_{od} when the price is increasing to β_{od} . Hence, the price-demand function can be formulated as

$$p = p_{re} + (p_{od} - p_{re}) \frac{\beta - \beta_{re}}{\beta_{od} - \beta_{re}}.$$

We adopt the linear price-demand function in this paper because it is one of the most commonly used function. Other complicated functions will be researched in further works.

3.4.2 Calculation of the Revenue

Based on the above analysis, the price and the customer demand are the two main factors having an effect on revenue. To calculate the revenue, it is necessary to calculate the expected charge to each request and the actual customer demand.

Expected Charge. The expected charge of a request is affected by three factors: the VM sales price, the execution time of a request, and the BTU of the cloud broker. To study the expected service charge of a request, we need a complete specification on those factors. The following theorem gives the expected charge to a service request.

Theorem 3.1. Assume that the BTU of a cloud broker is U units of time. Under a given VM sales price β and average execution time of tasks \bar{t} , the expected charge to a service request is

$$E = U\beta \frac{1}{1 - e^{-U/\bar{t}}}. \quad (4)$$

Proof 3.1. Since the BTU of a cloud broker is U units of time, the charge function of a service request with execution time t can be calculated as

$$R(t) = U \left\lceil \frac{t}{U} \right\rceil \beta. \quad (5)$$

This equation means that a service request is charged $(n+1)U\beta$ if its execution time t is in the interval $(nU, (n+1)U]$. The β means the VM price per unit of time.

Recall that the execution time t of each request is i.i.d. exponential random variables with mean \bar{t} ; hence, the probability distribution function of t is

$$f(t) = \frac{1}{\bar{t}} e^{-t/\bar{t}}.$$

The expected charge to a service request is

$$\begin{aligned} E &= \int_0^{\infty} f(t)R(t)dt \\ &= \sum_{n=0}^{\infty} \int_{nU}^{(n+1)U} f(t)(n+1)U\beta dt \\ &= \sum_{n=0}^{\infty} (n+1)U\beta \left(-e^{-t/\bar{t}} \right) \Big|_{nU}^{(n+1)U} \\ &= U\beta \sum_{n=0}^{\infty} e^{-nU/\bar{t}} \\ &= U\beta \lim_{n \rightarrow \infty} \frac{1 - (e^{-U/\bar{t}})^n}{1 - e^{-U/\bar{t}}} \\ &= U\beta \frac{1}{1 - e^{-U/\bar{t}}}. \end{aligned}$$

The theorem is proven. \square

Customer Demand. The actual customer demand is affected by the VM sales price. Under a given VM sales price β , the request arrival rate of a cloud broker is

$$\lambda = p\lambda_{max} = \left(p_{re} + (p_{od} - p_{re}) \frac{\beta - \beta_{re}}{\beta_{od} - \beta_{re}} \right) \lambda_{max}.$$

However, there are a small portion of requests rejected by the system due to the limited resources. Under a given system size n and average execution time of requests \bar{t} , the percentage of rejected requests P_L can be calculated using Eq. (2); then, the revenue obtained by a cloud broker in an unit of time can be calculated as

$$\begin{aligned} \mathcal{R} &= \lambda(1 - P_L)E \\ &= \left(p_{re} + (p_{od} - p_{re}) \frac{\beta - \beta_{re}}{\beta_{od} - \beta_{re}} \right) \lambda_{max} U\beta \frac{1}{1 - e^{-U/\bar{t}}} \\ &\quad \times \left(1 - \left[\sum_{k=0}^n \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k \right]^{-1} \frac{1}{n!} \left(\frac{\lambda}{\mu} \right)^n \right). \end{aligned} \quad (6)$$

3.5 Problem Description

Because the profit is defined as the revenue minus the cost. Hence, according to the above analysis, the expected net profit of a cloud broker in one unit of time is

$$\begin{aligned} Pro &= \mathcal{R} - \mathcal{C} \\ &= \lambda(1 - P_L)E - n\beta_{re}, \end{aligned} \quad (7)$$

where

$$\lambda = p\lambda_{max} = \left[p_{re} + (p_{od} - p_{re}) \frac{\beta - \beta_{re}}{\beta_{od} - \beta_{re}} \right] \lambda_{max},$$

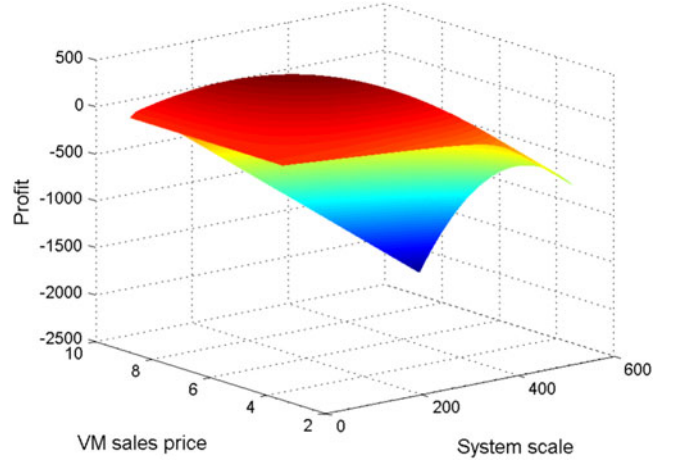


Fig. 5. The mesh of profit versus n and β .

$$P_L = \pi_n = \left[\sum_{k=0}^n \frac{1}{k!} \left(\frac{\lambda}{\mu} \right)^k \right]^{-1} \frac{1}{n!} \left(\frac{\lambda}{\mu} \right)^n,$$

and

$$E = U\beta \frac{1}{1 - e^{-U/\bar{t}}}.$$

From Eq. (7), we can see that the profit is determined by two parameters, i.e., the VM price β and the system size n . To maximize the profit of a cloud broker, we should find an optimal combination of β and n . Hence, we define the profit maximization problem as an optimal multiserver configuration and VM pricing problem, which is defined as follows: Given the total customer demand λ_{max} , the average request execution time \bar{t} , the on-demand resource price β_{od} , the reserved resource price β_{re} , and the corresponding market share ratio p_{od} and p_{re} , find an optimal combination of VM sales price β and system size n for the cloud broker such that its profit is maximized. The optimization problem can be formulated as

$$\max Pro(\beta, n),$$

subject to

$$0 < \rho < 1.$$

Fig. 5 gives the graph of function Pro where $\lambda_{max} = 100$, $\bar{t} = 8$, $\beta_{re} = 4$, $\beta_{od} = 9$, $p_{re} = 0.5$, $p_{od} = 0$, and $U = 0.5$.

From the figure, we can see that the profit of a cloud broker is varying with server size n and service price β , and there must be an optimal combination of n and β where the profit is maximized. In the next section, we will show how to solve this optimization problem.

In Table 1, we summarize all the notations used in the paper to improve readability.

4 OPTIMAL SOLUTION

In this section, a partial derivative combined with the bisection search method is adopted to solve the profit optimization problem.

4.1 The Analytical Method

We first solve our optimization problem analytically, assuming that n and β are continuous variables. To this end,

TABLE 1
Notations Used in this Paper

Notation	Description
β_{re}	the unit price of reserved resources rented from cloud providers
β_{od}	the unit price of on-demand resources rented from cloud providers
β	the unit price of resources provided by cloud brokers
p_{re}	the percentage of customers attracted by cloud brokers at the price β_{re}
p_{od}	the percentage of customers attracted by cloud brokers at the price β_{od}
n	the server size of virtual platform owned by the cloud broker
t	the execution time of a request and \bar{t} is the average execution time
λ_{max}	the maximal task arrival rate, which presents the total customer demand
λ	the actual task arrival rate of a cloud broker
ρ	the server utilization of a cloud broker
π_k	the probability that k requests are in the system
\mathcal{R}	the total revenue of a cloud broker
\mathcal{C}	the total cost of a cloud broker
P_L	the loss probability of customers due to limited resources

a closed-form expression of P_L is needed. In this paper, we use the same closed-form expression as [26], which is

$$\sum_{k=0}^n \frac{(n\rho)^k}{k!} \approx e^{n\rho}.$$

This expression is very accurate when n is not too small and ρ is not too large [29]. Since Stirling's approximation of $n!$ is $\sqrt{2\pi n}(\frac{n}{e})^n$, one closed-form expression of P_L is

$$P_L \approx \frac{e^{n(1-\rho)} \rho^n}{\sqrt{2\pi n}}. \quad (8)$$

In the following, we will solve our optimization problems based on above closed-form expression of P_L . Before the solutions are given, we first rewrite Pro as

$$Pro = \lambda\beta T \left(1 - \frac{e^{n(1-\rho)} \rho^n}{\sqrt{2\pi n}} \right) - n\beta_{re},$$

where

$$\lambda = \left[p_{re} + \left(p_{od} - p_{re} \right) \frac{\beta - \beta_{re}}{\beta_{od} - \beta_{re}} \right] \lambda_{max},$$

and

$$\rho = \lambda \bar{t} / n,$$

and

$$T = U \frac{1}{1 - e^{-U/\bar{t}}}.$$

4.1.1 Optimal Price

Given λ_{max} , \bar{t} , β_{re} , β_{od} , p_{re} , p_{od} , and n , our objective is to find the optimal β such that Pro is maximized. To maximize profit, β must be found such that

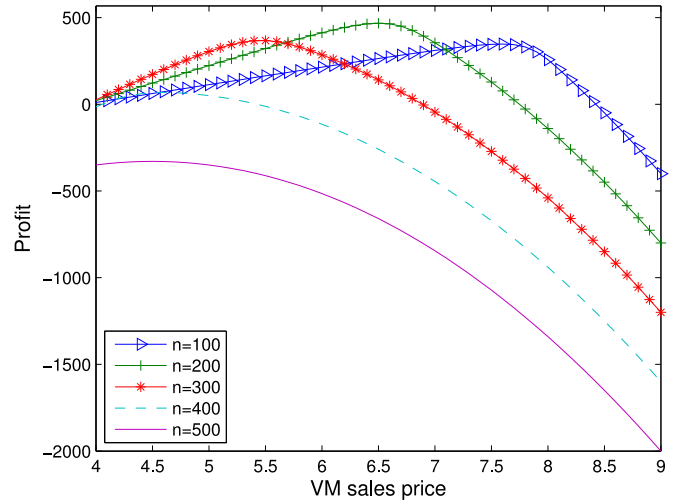


Fig. 6. Optimal profit versus resource price.

$$\frac{\partial Pro}{\partial \beta} = 0.$$

Since

$$\frac{\partial \lambda}{\partial \beta} = \frac{p_{od} - p_{re}}{\beta_{od} - \beta_{re}} \lambda_{max},$$

we have

$$\frac{\partial \lambda \beta}{\partial \beta} = \lambda + \frac{p_{od} - p_{re}}{\beta_{od} - \beta_{re}} \lambda_{max} \beta,$$

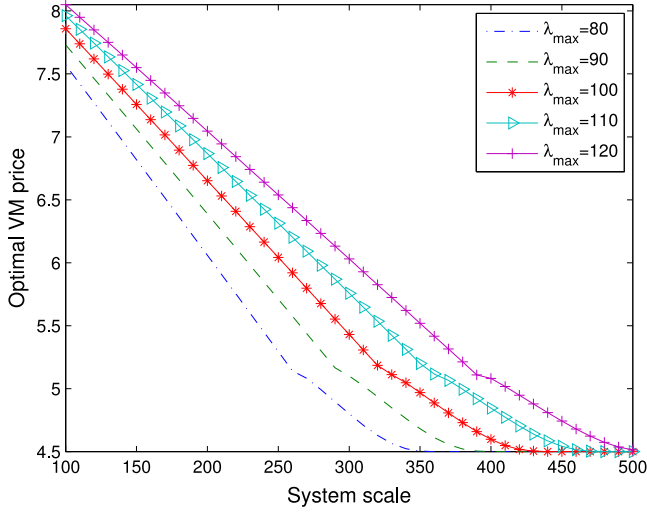
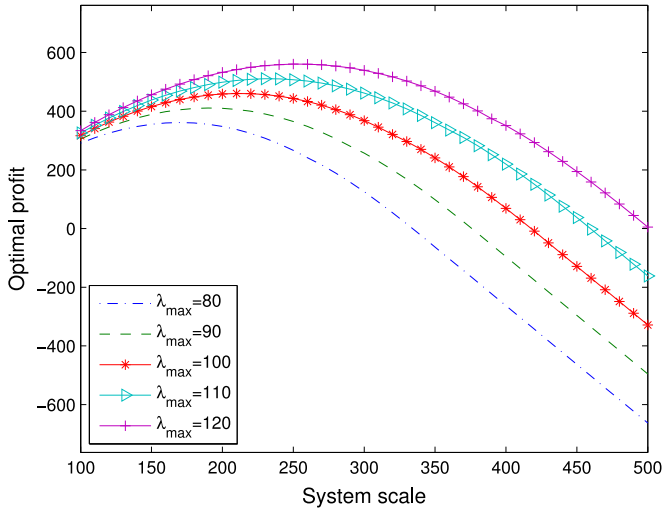
and

$$\begin{aligned} \frac{\partial P_L}{\partial \beta} &= \frac{1}{\sqrt{2\pi n}} \left[\frac{\partial (e^{n(1-\rho)})}{\partial \beta} \rho^n + e^{n(1-\rho)} \frac{\partial \rho^n}{\partial \beta} \right] \\ &= \frac{1}{\sqrt{2\pi n}} \left[\left(\rho^n e^{n(1-\rho)} \left(-\frac{1}{\mu} \frac{\partial \lambda}{\partial \beta} \right) \right) \right. \\ &\quad \left. + e^{n(1-\rho)} n \rho^{n-1} \frac{1}{n\mu} \frac{\partial \lambda}{\partial \beta} \right] \\ &= \frac{e^{n(1-\rho)} \rho^n}{\sqrt{2\pi n}} \frac{\partial \lambda}{\partial \beta} \frac{1-\rho}{\mu \rho} \\ &= P_L \frac{\partial \lambda}{\partial \beta} \frac{1-\rho}{\mu \rho}. \end{aligned} \quad (9)$$

Furthermore, we get

$$\begin{aligned} \frac{\partial Pro}{\partial \beta} &= \left(\lambda + \frac{\partial \lambda}{\partial \beta} \beta \right) (1 - P_L) T + \lambda \beta T \left(-P_L \frac{\partial \lambda}{\partial \beta} \frac{1-\rho}{\mu \rho} \right) \\ &= \lambda (1 - P_L) T + \beta T \frac{\partial \lambda}{\partial \beta} \left[(1 - P_L) - P_L n (1 - \rho) \right]. \end{aligned} \quad (10)$$

We cannot obtain a closed-form solution to β , but we can get the numerical solution to it. In Fig. 6, we demonstrate the net profit in one unit of time as a function of the price β and the server size n . The profit is calculated based on the precise P_L as Eq. (2), and the parameters are set as follows. β_{re} and β_{od} are set as 4 and 9; p_{re} and p_{od} are set as 0.5 and 0; \bar{t} is set as 8, BTU is set 0.5, and λ_{max} is set as 100. From Fig. 6, it is apparent that the Pro function is a convex curve

(a) Optimal size versus n and λ_{max} .(b) Maximal profit versus n and λ_{max} .Fig. 7. Optimal size and maximal profit versus n and λ_{max} .

on which exists an extreme point. So, the $\partial Pro/\partial \beta$ must have a decreasing interval during which we could find the zero point by the standard bisection method [30]. The algorithm is given as Algorithm 1. By the algorithm, the optimal value of β in Fig. 6 is 7.5885, 6.5064, 5.4675, 4.5985, 4.5000 for $n = 100, 200, 300, 400, 500$, respectively.

In Fig. 7, we demonstrate the optimal price and maximal profit in one unit of time as a function of n and λ_{max} . Therefore, for each combination of n and λ_{max} , we find the optimal price for a cloud broker and the corresponding maximal profit it can obtain. The parameters are set to be same as Fig. 6.

From the figures, we can see that under a given λ_{max} , the optimal price is decreasing with the increase in system size. This is explained as follows. It is obvious that more VMs lead to more cost. To utilize the resources sufficiently and improve the revenue, the VM price is lowered to attract more customers, which is so-called small profits but quick turnover (SPQT) strategy. However, the optimal profit is not monotone increasing with the increasing system size. When the system size reaches a certain point, the extra cost conduct by increasing VMs further starts to exceed the

TABLE 2
Quality of Solutions (Optimal Price)

n	Brute Force Search		Partial Derivatives		Error(%)
	β	Pro	β	Pro	
50	8.1371	202.6171	8.4496	166.2390	17.954%
100	7.5885	347.8085	7.8580	316.6733	8.952%
150	7.0442	436.0837	7.2576	415.8876	4.631%
200	6.5064	468.1743	6.6521	458.9012	1.981%
250	5.9787	445.1308	6.0431	443.4117	0.386%
300	5.4675	368.5473	5.4309	368.0427	0.137%
350	4.9883	241.1389	4.9688	241.0231	0.048%
400	4.5985	68.8678	4.5979	68.8678	0.000%
450	4.5001	-128.8486	4.5001	-128.8486	0.000%

increased revenue by adopting the SPQT strategy. Hence, the total profit increases at the early stage and then decreases. Moreover, the figures show that the optimal price and the optimal profit are all related with the λ_{max} . Under a given system size, a greater λ_{max} will lead to a higher optimal price and more profit.

Algorithm 1. Finding the Optimal Price

Input: $\lambda_{max}, \bar{t}, n, \beta_{ref}, \beta_{od}, P_{ref}$, and p_{od} ;

Output: optimal price opt_beta of resources and optimal profit

```

 $opt\_pro$ ;
1:  $opt\_beta = -\infty, opt\_pro = -\infty$ ;
2:  $\beta_{start} \leftarrow$  the minimal price satisfying  $\rho < 1$ ;
3:  $\beta_{end} \leftarrow \beta_{od}$ ;
4: calculate  $Der_{start}$  and  $Der_{end}$  using Eq. (10);
5: if  $Der_{start} \times Der_{end} > 0$  then
6:    $opt\_beta = \beta_{start}$ ;
7:   calculate  $opt\_pro$  using Eq. (7);
8:   exit;
9: end if
10: while  $Der_{start} - Der_{end} > error$  do
11:    $\beta_{middle} = (\beta_{start} + \beta_{end})/2$ ;
12:   calculate  $Der_{middle}$  using Eq. (10);
13:   if  $Der_{start} \times Der_{middle} > 0$  then
14:      $\beta_{start} \leftarrow \beta_{middle}$ ;
15:   else
16:      $\beta_{end} \leftarrow \beta_{middle}$ ;
17:   end if
18: end while
19:  $opt\_beta = (\beta_{start} + \beta_{end})/2$ ;

```

In Algorithm 1, the partial derivative is calculated based on the estimation value of P_L first, and then the extremal solutions are solved using the bisection search method. Hence, the solutions obtained by Algorithm 1 have a certain of error with the precise solutions. To verify the precision of the solutions, we compare the optimal solutions obtained by our method with that obtained by a brute force search method. The comparison results are given in Table 2. In the comparison, the System size n is set from 50 to 450 in step of 50, λ_{max} is set as 100, and other parameters are same as Fig. 7. From the results, we can see that the error is less than 2 percent when the n is greater than 200. When the n is smaller than 200, with the decrease of n , the error becomes greater. That is because the error between the estimation value and precision value of P_L is very large when n is small.

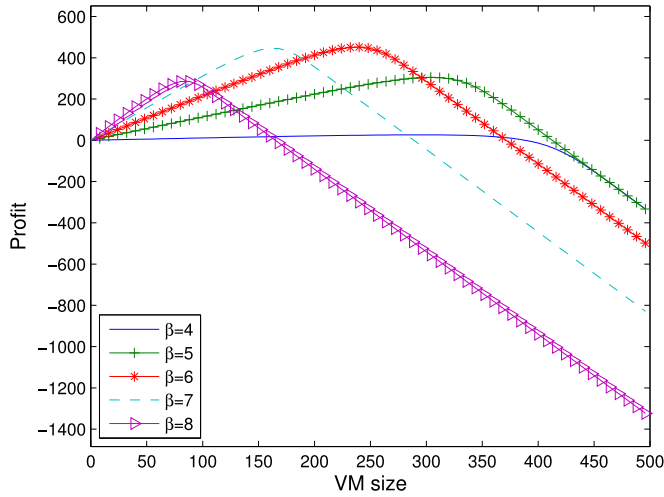


Fig. 8. Optimal profit versus total investment.

4.1.2 Optimal Size

Given λ_{max} , \bar{v} , β_{re} , β_{od} , p_{re} , p_{od} , and β , our objective is to find n such that Pro is maximized. To maximize profit, n must be found such that

$$\frac{\partial Pro}{\partial n} = 0.$$

Since

$$(e\rho)^n = e^{\ln(e\rho)^n} = e^{n \ln(e\rho)},$$

we have

$$\frac{\partial (e\rho)^n}{\partial n} = (\ln(e\rho) - 1)(e\rho)^n,$$

and then

$$\begin{aligned} \frac{\partial P_L}{\partial n} &= \frac{\partial \left(\frac{e^{n(1-\rho)} \rho^n}{\sqrt{2\pi n}} \right)}{\partial n} = \frac{\partial \left(\frac{(e\rho)^n e^{-n\rho}}{\sqrt{2\pi n}} \right)}{\partial n} \\ &= e^{-\frac{\lambda}{\mu}} \frac{\frac{\partial (e\rho)^n}{\partial n} \sqrt{2\pi n} - (e\rho)^n \frac{\partial \sqrt{2\pi n}}{\partial n}}{2\pi n} \\ &= e^{-\frac{\lambda}{\mu}} \frac{(\ln(e\rho) - 1)(e\rho)^n \sqrt{2\pi n} - (e\rho)^n \sqrt{\frac{\pi}{2n}}}{2\pi n} \\ &= e^{n(1-\rho)} \rho^n \left[\frac{\ln(e\rho) - 1}{\sqrt{2\pi n}} - \frac{1}{2n} \frac{1}{\sqrt{2\pi n}} \right] \\ &= P_L \left[\ln(e\rho) - 1 - \frac{1}{2n} \right]. \end{aligned} \quad (11)$$

Then, we get

$$\frac{\partial Pro}{\partial n} = \lambda \beta T P_L \left(1 + \frac{1}{2n} - \ln(e\rho) \right) - \beta_{re}. \quad (12)$$

Similarly, we cannot get the closed-form expression of n , so we can use the bisection method [30] to find the numerical solution of n . In Fig. 8, we demonstrate the net profit in one unit of time as a function of the price β and the system size n . The parameters are same as Fig. 6. From Fig. 8, it is apparent that the Pro function is also a convex curve that contains an extreme point. Adopting the bisection method, the optimal size n in Fig. 8 is 400, 320, 240, 163, 84 for $\beta = 4, 5, 6, 7, 8$, respectively.

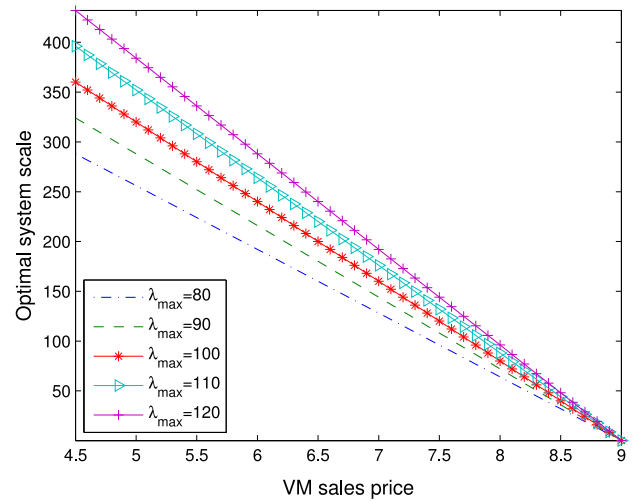
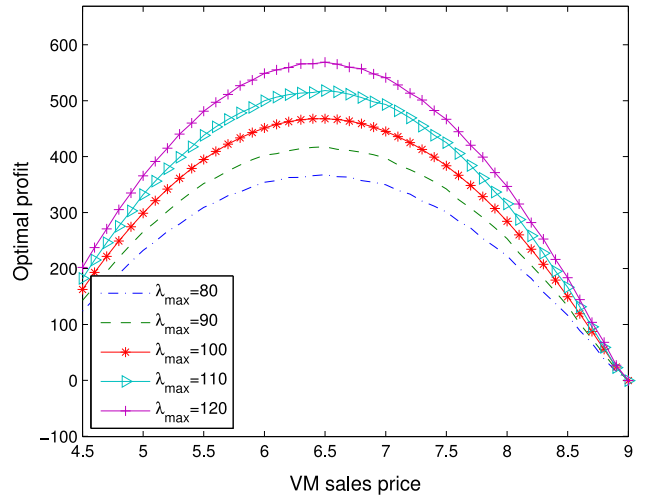

 (a) Optimal size versus β and λ_{max} .

 (b) Maximal profit versus β and λ_{max} .

 Fig. 9. Optimal profit and maximal profit versus β and λ_{max} .

In Fig. 9, we demonstrate the optimal size and maximal profit in one unit of time as a function of β and λ_{max} . Under different λ_{max} , we observe the changing trend of the optimal size for a cloud broker and the corresponding maximal profit it can obtain. The parameters are same as those in Fig. 6. Fig. 9a shows the trend of the optimal system size with the increasing VM sales price under different λ_{max} . From the figure we can see that under a given λ_{max} , the optimal system size decreases with the increase of VM price. Thus, if a cloud broker sets a higher price, the number of VMs it rented from public clouds should be reduced. That is because a higher sales price leads to a decrease of customer demand; hence, on the premise of satisfying customer demand, the rented VMs should be reduced to cut down the rental cost. In addition, we can find from Fig. 9a that under a fixed price, more VMs are needed when the λ_{max} is increasing, as a greater λ_{max} also leads to more VM requests in a unit of time arriving at the cloud broker. The cloud broker should rent more VMs to provide enough computing capacity to the increasing VM requests.

Fig. 9b shows the corresponding profit under the configurations in Fig. 9a. The figure shows that the optimal profit shows a consistent trend with the λ_{max} value, whereas under a given λ_{max} , the optimal profit first increases with

TABLE 3
Quality of Solutions (Optimal Price)

β	Brutal Force Search		Partial Derivatives		Error(%)
	n	Pro	n	Pro	
4.0	400	-13.6067	400.08	-13.9267	2.352%
4.5	360	162.8083	360.08	162.4883	0.197%
5.0	320	299.0461	320.08	298.7261	0.107%
5.5	280	395.2308	280.08	394.9108	0.081%
6.0	240	451.5263	240.08	451.2063	0.071%
6.5	201	468.2427	200.08	467.8389	0.086%
7.0	163	446.1830	160.08	445.1413	0.233%
7.5	124	385.5531	120.08	383.6483	0.494%
8.0	84	286.8998	80.08	284.3525	0.888%
8.5	43	152.0922	40.08	149.6756	1.589%

the increasing sales price and starts decreasing after an extreme point. This is explained as follows. At the beginning, the VM sales price is low, and the revenue is increased with the increasing price. Although the cost is also increased correspondingly, the increased cost is lower than the increased revenue, so the total profit shows an increasing trend. Surpassing a special price, increasing the VM sales price can no longer raise the total profit because the increased cost starts surpassing the increased revenue. Hence, the profit shows a decreasing trend.

Similar with Table 2, we compare the optimal size obtained by our method with that obtained by a brute force search method, and the comparison results are given in Table 3. In the comparison, the VM price β is set from 4.0 to 8.5 in step of 0.5, the λ_{max} is set as 100, and the other parameters are same as Fig. 7.

4.1.3 Optimal Size and Price

Given $\lambda_{max}, \bar{t}, \beta_{re}, \beta_{od}, p_{re}$, and p_{od} , our third problem is to find n and β such that the profit is maximized. We just need to find n and β such that $\partial Pro/\partial n = 0$ and $\partial Pro/\partial \beta = 0$, where $\partial Pro/\partial n$ and $\partial Pro/\partial \beta$ have been derived in the last two sections. The two equations can be solved by the grid bisection method which is adopted in [26]. The algorithm is given as follows.

In Table 4, we demonstrate the optimal size and price that a cloud broker should be configured under different λ_{max} , and the corresponding profit per unit of time that the cloud broker can obtain. The λ_{max} is set from 60 to 130 in step of 10. From the table, we can see that with the increase of λ_{max} , the optimal price undergoes no evident change. The optimal size is increasing with the increased λ_{max} as well as the optimal profit.

TABLE 4
Quality of Solutions (Optimal Size and Price)

λ_{max}	Brutal Force Search				Partial Derivatives				Error(%)	TSR(%)
	n_{opt}	β_{opt}	Pro_{opt}	$T(\text{unit: s})$	n_{opt}	β_{opt}	Pro_{opt}	$T(\text{unit: s})$		
60	122	6.474	268.8419	22.381	128	6.5223	263.3711	0.519	2.03%	97.68%
70	142	6.474	318.2826	22.112	150	6.5004	312.1968	0.509	1.91%	97.70%
80	163	6.468	368.0537	22.064	171	6.4971	361.4982	0.556	1.78%	97.48%
90	183	6.474	418.0824	22.155	192	6.4923	411.2474	0.493	1.63%	97.77%
100	204	6.462	468.3418	21.95	205	6.5865	460.6158	0.481	1.65%	97.81%
110	224	6.468	518.7865	21.978	234	6.4899	510.7347	0.497	1.55%	97.74%
120	245	6.462	569.3975	21.974	252	6.514	561.4408	0.463	1.40%	97.89%
130	266	6.456	620.1480	21.926	276	6.4852	611.2303	0.438	1.44%	98.00%

Algorithm 2. Finding the Global Optimal Size and Price

Input: $\lambda_{max}, \bar{t}, \beta_{re}, \beta_{od}, p_{re}$, and p_{od} ;

Output: optimal number opt_n of rented VMs and optimal price opt_beta ;

- 1: find a proper interval of VM size $[n_{start}, n_{end}]$;
- 2: calculate the optimal price opt_beta_{start} under n_{start} by Algorithm 1;
- 3: calculate the optimal price opt_beta_{end} under n_{end} by Algorithm 1;
- 4: calculate Der_{start} and Der_{end} using Eq. (12) with parameters $(n_{start}, opt_beta_{start})$ and $(n_{end}, opt_beta_{end})$, separately;
- 5: **if** $Der_{start} \times Der_{end} > 0$ **then**
- 6: $opt_n = n_{start}$ and $opt_beta = opt_beta_{start}$;
- 7: calculate opt_pro using Eq. (7);
- 8: **break**;
- 9: **else**
- 10: **while** $Der_{start} - Der_{end} > error$ **do**
- 11: $n_{middle} = (n_{start} + n_{end})/2$;
- 12: calculate the optimal price opt_beta_{middle} under n_{middle} by Algorithm 1;
- 13: calculate Der_{middle} using Eq. (12) with parameters $(n_{middle}, opt_beta_{middle})$;
- 14: **if** $Der_{start} \times Der_{middle} > 0$ **then**
- 15: $n_{start} \leftarrow n_{middle}$;
- 16: **else**
- 17: $n_{end} \leftarrow n_{middle}$;
- 18: **end if**
- 19: **end while**
- 20: $opt_n = n_{start}$ and $opt_beta = opt_beta_{start}$;
- 21: **end if**

Moreover, Table 4 also shows a comparison between the optimal solution of our method (Partial Derivative Optimization, *PDO* for short) and that solved by the brute force search method (*BFS*). The results show that the profit obtained by our strategy is close to the global optimal profit adopting the (*BFS*), and the error rate is less than 2 percent. For example, the profit calculated by *PDO* is 460.6158 when λ_{max} is 100, which is only 1.65 percent less than the accurate global optimal profit calculated by *BFS*. Moreover, we compare the computation time (T) of two methods, and the performance parameter is defined as Computation Time Saving Ratio (*TSR*), which is calculated as

$$TSR = \frac{\text{Time of BFS} - \text{Time of PDO}}{\text{Time of BFS}}.$$

TABLE 5
User Cost versus β ($\bar{t}=8$, E of AEC=76.5937)

U	β	4	5	6	7	8	9
0.5	E	33.0104	41.2630	49.5156	57.7682	66.0208	74.2734
	Cost Saving Rate	56.902%	46.127%	35.353%	24.578%	13.804%	3.029%
1	E	34.0417	42.5521	51.0625	59.5729	68.0833	76.5937
	Cost Saving Rate	55.556%	44.444%	33.333%	22.222%	11.111%	0.000%

The comparative results show that the computing efficiency of our method is much higher than *BFS* but only a little accuracy loss.

5 PERFORMANCE ANALYSIS AND COMPARISON

In this section, a series of numerical calculations are conducted to verify the function of the cloud broker.

5.1 Performance Analysis

The emergence of the cloud broker provides customers one more choice when selecting the providers of cloud computing. It can not only provide the same service as the public clouds but also save a great amount of cost for customers. In the following, we conduct a series of numerical calculations to compare the cost of users when they submit requests to a cloud broker or public clouds, respectively.

According to Theorem 3.1, it is known that the expected charge to a service request is determined by three factors: the BTU U , the VM sales price β , and the average execution time \bar{t} . To verify the effect of the three factors on the user cost, we conduct three groups of calculations in the following. Amazon EC2, *AEC* for short, is adopted as the comparison. *AEC* is compared with the cloud broker under different parameters to verify how much cost they can save for users. The performance metric is formulated as

$$\text{Cost Saving Rate} = \frac{E \text{ of } AEC - E \text{ of a Cloud Broker}}{E \text{ of } AEC}.$$

Here, the BTU of *AEC* is set as 1 unit of time and its on-demand price β_{od} and reserved price β_{re} are set as 9 per unit of time and 4 per unit of time, respectively. The average execution time \bar{t} is set as 8 unit of time. Substituting these parameters into Eq. (4)

$$E \text{ of } AEC = 76.5937.$$

5.1.1 User Cost versus β

In the first group of calculations, we observe how the VM sales price β affects the user cost under the given BTU.

From Eq. (4) it is apparent that the user cost is linearly increasing with the price β . In Table 5, we show the average cost of users when β is varying from β_{od} to β_{re} in step of 1 (\bar{t} is set as 8, and the BTU U is set as 0.5 and 1, respectively).

We have calculated that the average cost of users in *AEC* is 76.5937. The table shows that when users submit their requests to a cloud broker with a lower VM sales price compared with *AEC*, a great deal of cost can be saved. And the lower the price is, the more cost that can be saved.

5.1.2 User Cost versus BTU

In the second group of calculations, we observe how the BTU affects the user cost under a given VM sales price. We set the VM sales price β of the cloud broker as β_{od} and $2/3\beta_{od}$, respectively. The BTU of cloud brokers is varying from $1/6$ to 1 in step of $1/6$. In addition, the average execution time \bar{t} is set as 8. The user cost of cloud brokers in different situations and the cost saving rate compared with *AEC* are given in Table 6.

The results in Table 6 show that the user cost is affected greatly by the BTU of cloud brokers. The smaller the BTU is, the more cost that can be saved for users on average. For example, assume that the execution time of a request is 2.1; if it is submitted to a cloud broker with a BTU of 1, the total cost is $1 \times \lceil 2.1/1 \rceil \beta = 3\beta$, and if it is submitted to a cloud broker with $U=\{5/6, 4/6, 3/6, 2/6, 1/6\}$, the total cost is 2.50β , 2.67β , 2.50β , 2.33β , and 2.17β , respectively. Although sometimes a smaller BTU leads to a higher cost, e.g., the cost of $4/6$ -BTU is greater than that in $5/6$ -BTU, the overall trend of the average cost is decreasing with the decreasing BTU.

Moreover, even though the VM sales price of a cloud broker is set the same as the price of *AEC*, it still can reduce cost for users by setting a smaller BTU. For example, when the VM sales price of a cloud broker β is set as β_{od} , the cost as high as 5.015 percent can be saved compared with *AEC*.

5.1.3 User Cost versus \bar{t}

In the third group of calculations, we observe how the parameter \bar{t} affects the user cost under the given VM sales price and BTU. The VM sales price β of the cloud broker is

TABLE 6
User Cost versus BTU ($\bar{t}=8$, E of AEC=76.5937)

β	U	1/6	2/6	3/6	4/6	5/6	1
$\frac{2}{3}\beta_{od}$	E	48.5017	49.0069	49.5156	50.0278	50.5434	51.0625
	Cost Saving Rate	36.677%	36.017%	35.353%	34.684%	34.011%	33.333%
β_{od}	E	72.7526	73.5104	74.2734	75.0417	75.8151	76.5937
	Cost Saving Rate	5.015%	4.026%	3.029%	2.026%	1.017%	0.000%

TABLE 7
User Cost versus \bar{t}

U	\bar{t}	0.5	4	8	12	24
<i>E of AEC (U=1, $\beta=6$)</i>		10.4087	40.6873	76.5937	112.5625	220.5312
1/6	<i>E ($\beta=6$)</i>	3.5277	24.5035	48.5017	72.5012	144.5006
	<i>Cost Saving Rate</i>	66.108%	39.776%	36.677%	35.590%	34.476%
3/6	<i>E ($\beta=6$)</i>	4.7459	25.5312	49.5156	73.5104	145.5052
	<i>Cost Saving Rate</i>	54.404%	37.250%	35.353%	34.694%	34.021%
1	<i>E ($\beta=6$)</i>	6.9391	27.12487	51.0625	75.0417	147.0208
	<i>Cost Saving Rate</i>	33.333%	33.333%	33.333%	33.333%	33.333%

set as $2/3\beta_{od} = 6$ and the BTU is set as 1/6, 3/6, and 1, respectively. The user cost is calculated for the cloud broker and AEC separately when the \bar{t} value is set as 0.5, 4, 8, 12, 24, respectively. The results are given in Table 7.

Table 7 shows that when the cloud broker has the same BTU and lower price compared with AEC, the cost of users in the cloud broker is always smaller than that in AEC, but the ratio of saved cost is not changing with the increasing \bar{t} . When both of the BTU and the VM sales price of the cloud broker are smaller than that of AEC, the ratio of saved cost by the cloud broker is affected by \bar{t} , and the smaller the \bar{t} is, the greater the amount of cost that can be saved for the users by the cloud broker.

5.1.4 Saved Cost Ratio Under Optimal Solution

In the last group of calculations, we show the average user cost of a cloud broker under the optimal system size and VM sales price calculated by our method. The BTU of the cloud broker is set as 0.5, and the average execution time of requests \bar{t} is set as 8. In Table 8, the optimal size, the optimal price, the optimal profit, and the average user cost are listed when the λ_{max} is varying from 60 to 130 in step of 10. The results show that for a cloud broker with a given BTU, no matter how great the λ_{max} is, the average user cost remains unchanged so long as the average execution time of requests is fixed. Under the optimal configuration, the cloud broker can save about 30 percent cost for users on average.

6 CONCLUSIONS

In this paper, we focus on the profit maximization problem of cloud brokers. A cloud broker is an intermediary entity between cloud service providers and customers, which buys

reserved instances from cloud providers for long periods of time and outsources them as on-demand VMs for a lower price and fine-grained BTU with respect to what the cloud service providers charge for the same VMs. Due to the lower service price and the finer-grained BTU compared with the public clouds, the cloud broker can save much cost for customers. This paper tries to guide cloud brokers on how to configure the virtual resource platform and how to price their service such that they can obtain the maximal profit. To solve this problem, the virtual resource platform is modeled as an $M/M/n/n$ queue model, and a profit maximization problem is built in which many profit-affecting factors are analyzed based on the queuing theory, as well as the relationship between them. The optimal solutions are solved combining the partial derivative and bisection method. Lastly, a series of calculations are conducted to analyze the changing trend of profit and the ratio of user cost savings.

In this paper, we adopt the linear price-demand price when we analyze the broker's profit since it is the most common function in real market. Whereas, different cloud markets might show different price-demand relationship. Hence, we will extend our study to consider more complicated price-demand curves in the further.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their valuable comments and suggestions. The research was partially funded by the Key Program of National Natural Science Foundation of China (Grant No. 61432005), the National Outstanding Youth Science Program of National Natural Science Foundation of China (Grant No. 61625202), the National Natural Science Foundation of China (Grant Nos. 61602170, 61502165, 61772182, 61370095, 61472124), China Postdoctoral Science Foundation (Grant No. 2018T110829).

REFERENCES

- [1] B. Rajkumar, J. Broberg, and A. M. Goscinski, *Cloud Computing: Principles and Paradigms*, vol. 8. Hoboken, NJ, USA: Wiley, 2011, pp. 1–41.
- [2] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "Above the clouds: A Berkeley view of cloud computing," Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2009-28, 2009.
- [3] S. Nesmachnow, S. Iturriaga, and B. Dorransoro, "Efficient heuristics for profit optimization of virtual cloud brokers," *IEEE Comput. Intell. Mag.*, vol. 10, no. 1, pp. 33–43, Feb. 2015.
- [4] K. Li, J. Mei, and K. Li, "A fund-constrained investment scheme for profit maximization in cloud computing," *IEEE Trans. Serv. Comput.*, p. 1, 2016, doi: 10.1109/TSC.2016.2589241.

TABLE 8
User Cost Under Optimal Configuration ($\bar{t}=8, U=0.5$)

λ_{max}	Derivation Optimal				
	n_{opt}	β_{opt}	Pro_{opt}	User Cost	Cost Saving Rate
60	128	6.5223	263.3711	53.8262	29.725%
70	150	6.5004	312.1968	53.6455	29.961%
80	171	6.4971	361.4982	53.6182	29.997%
90	192	6.4923	411.2474	53.5787	30.048%
100	205	6.5865	460.6158	54.3558	29.034%
110	234	6.4899	510.7347	53.5586	30.074%
120	252	6.5140	561.4408	53.7575	29.815%
130	276	6.4852	611.2303	53.5194	30.126%

- [5] L. M. Vaquero, L. Roderer-Merino, J. Caceres, and M. Lindner, "A break in the clouds: Towards a cloud definition," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2008.
- [6] P. Mell and T. Grance, "The NIST definition of cloud computing," *Nat. Inst. Standards Technol.*, vol. 53, no. 6, 2009, Art. no. 50.
- [7] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.
- [8] R. Zhang, K. Wu, M. Li, and J. Wang, "Online resource scheduling under concave pricing for cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 1131–1145, Apr. 2016.
- [9] Amazon EC2. 2017. [Online]. Available: <http://aws.amazon.com>
- [10] W. Wang, D. Niu, B. Li, and B. Liang, "Dynamic cloud resource reservation via cloud brokerage," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 400–409.
- [11] J. Mei, K. Li, A. Ouyang, and K. Li, "A profit maximization scheme with guaranteed quality of service in cloud computing," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3064–3078, Nov. 2015.
- [12] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of resource provisioning cost in cloud computing," *IEEE Trans. Serv. Comput.*, vol. 5, no. 2, pp. 164–177, Apr.–Jun. 2012.
- [13] Microsoft Azure. 2017. [Online]. Available: <http://www.microsoft.com/windowsazure>
- [14] O. Rogers and D. Cliff, "A financial brokerage model for cloud computing," *J. Cloud Comput.*, vol. 1, no. 1, pp. 1–12, 2012.
- [15] D. D'Agostino, A. Galizia, A. Clematis, M. Mangini, I. Porro, and A. Quarati, "A QoS-aware broker for hybrid clouds," *Comput.*, vol. 95, no. 1, pp. 89–109, 2013.
- [16] D. Limbani and B. Oza, "A proposed service broker strategy in cloudanalyst for cost-effective data center selection," *Int. J. Eng. Res. Appl.*, vol. 2, no. 1, pp. 793–797, 2014.
- [17] A. M. Manasrah, T. Smadi, and A. Almomani, "A variable service broker routing policy for data center selection in cloud analyst," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 29, pp. 365–377, 2017.
- [18] F. Larumbe and B. Sanso, "Green cloud broker: On-line dynamic virtual machine placement across multiple cloud providers," in *Proc. IEEE Int. Conf. Cloud Netw.*, 2016, pp. 119–125.
- [19] M. Radi, "Efficient service broker policy for large-scale cloud environments," *Int. J. Comput. Sci.*, vol. 12, 2015, Art. no. 1.
- [20] D. Rane and A. Srivastava, "Cloud brokering architecture for dynamic placement of virtual machines," in *Proc. IEEE 8th Int. Conf. Cloud Comput.*, 2015, pp. 661–668.
- [21] H. Sarbazi-Azad and A. Zomaya, "A cloud broker architecture for multicloud environments," pp. 359–376, 2013.
- [22] J. Tordsson, R. S. Montero, R. Moreno-Vozmediano, and I. M. Llorente, "Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers," *Future Generation Comput. Syst.*, vol. 28, no. 2, pp. 358–367, 2012.
- [23] A. Quarati, A. Clematis, A. Galizia, and D. D'Agostino, "Hybrid clouds brokering: Business opportunities, QoS and energy-saving issues," *Simul. Modelling Practice Theory*, vol. 39, no. 8, pp. 121–134, 2013.
- [24] G. Saha and R. Pasumarthy, "Maximizing profit of cloud brokers under quantized billing cycles: A dynamic pricing strategy based on ski-rental problem," in *Proc. Allerton Conf. Commun. Control Comput.*, 2015, pp. 1000–1007.
- [25] L. Kleinrock, *Queueing Systems: Theory*, vol. 1. Hoboken, NJ, USA: Wiley, 1975.
- [26] J. Cao, K. Hwang, K. Li, and A. Y. Zomaya, "Optimal multiserver configuration for profit maximization in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1087–1096, Jun. 2013.
- [27] V. Hajipour, V. Khodakarami, and M. Tavana, "The redundancy queueing-location-allocation problem: A novel approach," *IEEE Trans. Eng. Manage.*, vol. 61, no. 3, pp. 534–544, Aug. 2014.
- [28] S. Liu, S. Ren, G. Quan, M. Zhao, and S. Ren, "Profit aware load balancing for distributed cloud data centers," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, 2013, pp. 611–622.
- [29] K. Li, "Optimal configuration of a multicore server processor for managing the power and performance tradeoff," *J. Supercomput.*, vol. 61, no. 1, pp. 189–214, 2012.
- [30] A. Eiger, K. Sikorski, and F. Stenger, "A bisection method for systems of nonlinear equations," *ACM Trans. Math. Softw.*, vol. 10, no. 4, pp. 367–377, 1984.



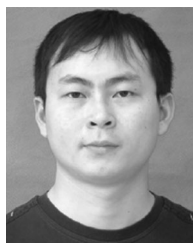
Jing Mei received the PhD in computer science from Hunan University, China, in 2015. She is currently an assistant professor with the College of Information Science and Engineering, Hunan Normal University. Her research interests include parallel and distributed computing, cloud computing, etc. She has published 14 research articles in international conference and journals, such as the *IEEE Transactions on Computers*, the *IEEE Transactions on Service Computing*, the *Cluster Computing*, the *Journal of Grid Computing*, and the *Journal of Supercomputing*.



Kenli Li received the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2003. He was a visiting scholar with the University of Illinois at Urbana-Champaign from 2004 to 2005. He is currently the dean and a full professor of computer science and technology with Hunan University and deputy director of National Supercomputing Center in Changsha. His major research areas include parallel computing, high-performance computing, grid, and cloud computing. He has published more than 150 research papers in international conferences and journals such as the *IEEE Transactions on Computers*, the *IEEE Transactions on Parallel and Distributed Systems*, the *IEEE Transactions on Signal Processing*, the *Journal of Parallel and Distributed Computing*, *ICPP*, and *CCGrid*. He serves on the editorial board of the *IEEE Transactions on Computers*. He is an outstanding member of the CCF. He is a senior member of the IEEE.



Zhao Tong received the BSc degree in computer science from the Beijing Institute of Technology, in 2007, and the PhD degree in computer science from Hunan University, China, in 2014. Currently, he is a lecturer with the College of Information Science and Engineering, Hunan Normal University. His research interests include modeling and scheduling for parallel and distributed computing systems, parallel system reliability, and parallel algorithms.



Qiang Li received the MS degree in computer science and technology from the National University of Defense Technology, in 2004, and the PhD degree in computer science and technology from Beihang University, China, in 2011. Currently, he is an associate professor with the College of Information Science and Engineering, Hunan Normal University. His research interests include distributed computing, cloud computing, and software engineering.



Keqin Li is a SUNY distinguished professor of computer science. His current research interests include parallel computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU-GPU hybrid and cooperative computing, multicore computing, storage and file systems, wireless communication networks, sensor networks, peer-to-peer file sharing systems, mobile computing, service computing, Internet of things, and cyber-physical systems. He has published more than 510 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, the *IEEE Transactions on Computers*, the *IEEE Transactions on Cloud Computing*, the *IEEE Transactions on Services Computing*, the *IEEE Transactions on Sustainable Computing*. He is a fellow of the IEEE.