Performance and Cost-effectiveness Analyses for Cloud Services Based on Rejected and Impatient Users

Yi-Ju Chiang[1], Student Member, IEEE, Yen-Chieh Ouyang[1]*, Member, IEEE, and Ching-Hsien Hsu[2], Senior Member, IEEE

Abstract—Cloud computing is an innovative service platform to offer diverse resources such as infrastructure, platform and software as services. However, one challenging aspect of such a service is the impatient user threat, which directly leads to numerous negative impacts such as poor throughput, unpredictable workload variation and resources wasted. In this paper, the problems of conducting system controls in a cost-effective way and simultaneously satisfying performance guarantees are first studied. System losses are analyzed according to the related performance factors and waiting buffer sizes. A cost model is developed to address a performances/cost tradeoff issue in which the user balking, reneging, system blocking and resources provisioning are all taken into account. The relationship between system controls and throughput variations in a multi-servers system with a finite buffer is demonstrated. A proposed policy combined with a heuristic algorithm allows cloud providers to control the service rate and buffer size within a system loss guarantee by solving constrained optimization problems. Simulation results show that more cost-saving and system throughput enhancement can be verified as compared to a system without applying our policy.

Index Terms— Cost optimality, loss probability, system blocking, throughput rate

1 Introduction

CLOUD computing is an emerging service paradigm to eliminate the burden of complex infrastructure management for companies/users. This service paradigm is developed as a utility computing to offer the pool of computing resources in a pay-as-you-go manner rather than traditional "own-and-use" patterns [1], [2]. Cloud providers supply service resources based on several fundamental models, including infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). For example, Amazon Elastic Compute Cloud (EC2), Amazon S3, Google's App Engine, Salesforce, etc. are all existing business models to provide computing infrastructure, data-storage, programming platforms, and software applications as services, respectively.

Cloud resources for providing on-demand or reserved instances can be leased through a network for temporary or long-term project needs. However, it is difficult to avoid either over-provisioning or under-provisioning when workloads have unpredictable/seasonal changes. Generally, resource over-provisioning helps to maintain quality of service (QoS), provide scalable resources, and avoid poor performances. Nevertheless, this approach will lead to many shortcomings as follows. First, it is difficult for a cloud provider to determine the best peak load provisioning. Second, huge energy consumption and resources provisioning cost are required; Third, most resources suffer from under-utilization for some days or months during off-seasons. Conversely, resource under-provisioning can conserve operational costs. However, performance degradations such as long waiting time, queuing length, high system loss, etc. are difficult to avoid.

Some impatient users may abandon this services system immediately after experiencing intolerable latency. Furthermore, "Impatient users" [3], [4] that are commonly found in network services will inevitably occur in a cloud environment for making use of application software [5], [6]. A cloud service system with impatient users/jobs has attracted some research attentions from balancing electricity bill [7], impatient task mapping [8] to pricing model [9]. Here, we focus on the "balking" and "reneging" in a queuing system, as shown in Fig. 1 (a) and Fig. 1 (b), respectively. The balking means that users refuse to join the queue due to long queuing length. Unlike the balking, users who choose not to wait in a queue after facing latency would be accused of reneging.

To avoid poor throughput, performance levels should be negotiated according to user's expectations and system's abilities in advance. To reach a consensus on service contents, signing a service level agreement (SLA) is an essential process by all interested parties [10]. The

Y. J. Chiang and Y.C. Ouyang are with Dept. of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan. E-mail: {yjchiang0320@gmail.com}, {ycouyang@nchu.edu.tw}
C. H. Hsu is with Dept. of Computer Science and Information Engineering Chung Hua University Hsinchu, Taiwan. E-mail: chh@chu.edu.tw

xxxx-xxxx/0x/$xx.00 © 200x IEEE Published by the IEEE Computer Society

SLA outlines all aspects of cloud service usages and obligations, such as Quality of Service (QoS) guarantees, billing, etc. Therefore, a penalty or compensation is required to pay when any party violates a SLA contract. In short, conducting an accurate performance analysis is required to satisfy performances guarantees in service-oriented systems.

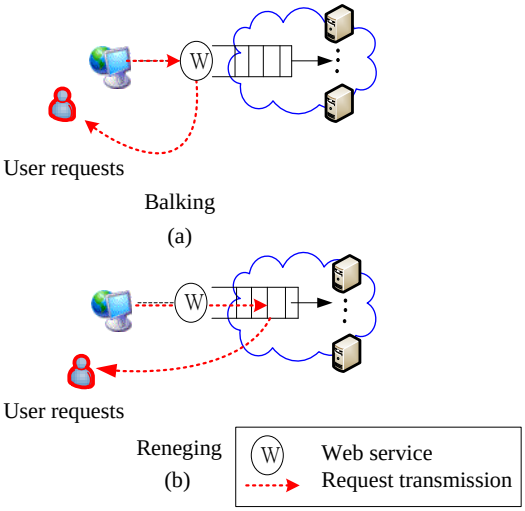

Fig. 1 Systems with (a) Balking (b) Reneging.

In this paper, we discuss the problems: (i) what is the system control effect on operational costs and performances when facing unpredictable request arrival rates? (ii) How to evaluate the relationship between user balking/reneging and system blocking on final throughput rates? (iii) How to address the optimal resource provisioning to achieve the cost optimality within a performance guarantee?

The optimal resource provisioning problem that we previously dealt with in [11] for profit optimization is further extended by taking into account the service rate control and user reneging impact. The main contributions of this paper can be summarized as follows:

- The challenge issues of arrival rate variations and resources wasted for a finite buffer queue with impatient users are studied by taking into account some related performance factors and the possible losses in a service system.
- A cost-effective policy combined with a heuristic algorithm is first proposed to address constrained optimization problems. The relationship between important performance indicators and operational costs can be determined in our designed service model.
- Simulations are conducted by considering different threats of potential balking and reneging factors. Experiment results show that more cost-saving and system throughput enhancement can be verified as compared to a system without applying our policy.

The remainder of this paper is structured as follows: Section 2 gives a brief overview related to finite-buffer queues, impatient users, cost-effectiveness analyses and latency information. Section 3 describes a multi-servers queuing system with a finite buffer and impatient users. Probabilities of related system loss are calculated. In Section 4, a cost model is developed and the **C**ost-**E**ffective policy in an **A**bandoned system (CEA policy) is presented to solve constrained optimization problems. Simulations and comparison of results are shown in Section 5. The whole work and future research are concluded in Section 6.

2 Related Work

2.1 Queuing Systems with Impatient Users

Researches in diverse systems with impatient users have been a long history; a review of related works is provided as follows. In [12], the performance of a telephone system with patient and impatient users were both studied. Expressions of performance measures, including the average number of patient customers or impatient customers in the system, etc. could be expressed in terms of the joint probabilities. The waiting time probabilities, the average waiting time of a customer in a buffer, and the probability that a customer would be served as a patient customer were also obtained.

In [13], Mandelbaum and Zeltyn were motivated by a phenomenon that had been observed in a telephone call data center: a clear linear relation between an abandoned probability and an average waiting time. The issues that arose in the introduction would be explored within the framework of the M/M/n+G queuing system. System performances for a variety of patience distributions were explored over different arrival rates. Under the assumption that the arrival rate converged to zero, they computed the asymptotic ratio between the probabilities of abandoning.

An analytic cost model for M/G/1/N queuing systems was presented in [14]. Doran, Lipsky and Thompson explored the interplay of queue size, customer loss, and mean service time for various service time distributions. It considered the cost of customer loss versus customer delays by varying buffer size and processor speed. In [15], Ghosh and Weerasinghe addressed a rate control problem associated with a single server. An infinite horizon cost minimization problem was considered. They obtained an explicit optimal strategy for the limiting diffusion control problem (the Brownian control problem or BCP) which consisted of a threshold-type optimal rejection process and a feedback-type optimal drift control. This solution was then used to construct an asymptotically optimal control policy.

A service system with impatient customer was an important research issue in cloud resources provisioning, but little mentioned in previous works. In [5], Mehdi et al.

evaluated the proposed model impact on impatient jobs. The proposed algorithm mapped the jobs with inspiration by using Minimum Completion Time scheduling algorithm (MCT). However, this work only presented an algorithm to evaluate the impact of the proposed model on impatient jobs. Rigorous analyses in performances factors were ignored.

2.2 Cloud Systems with Finite Buffers and Delay Information

Cloud computing has attracted considerable research attention, but only a few works done so far have addressed finite capacity models. In [16], a cloud center was model as an M/M/m/m queuing system to conduct a preliminary study of the fault recovery impact on a cloud service performance. When a user submitted a service request to the cloud, the request would first arrive at the cloud management system (CMS) which maintained a request queue. If the queue was not full, the request would enter the queue; otherwise it would be dropped and the requested service failed. Cloud service performance was quantified by a service response time, whose probability density function was derived.

In [17], Khazaei, Misic and Misic described a novel approximate analytical model for performance evaluation of cloud server farms and solved it to obtain estimation of the complete probability distribution of request response time and other important performance indicators. They also pointed out that accommodated heterogeneous services in a cloud center might impose longer waiting time for its clients than a homogeneous equivalent. However, no buffer size control or no impatient behavior was discussed or mentioned in their work.

On the other hand, for the purpose of enhancing service quality, information about delays was provided to users in cloud systems. In [18], Cappos et al. showed the Seattle program measures network latency to a list of IP addresses and displayed a webpage for showing the latency to each node. For modeling global cloud resources, a Voronoi Diagrams device was combined with near-real-time network latency information in [19]. Shouraboura and Bleher presented a novel Virtual Cloud Model (VCM) supplemented with near-real-time network latency information.

Instances would communicate state information with each other in order to keep the "world" consistent in appearance to all participants. VCM could also be used to share application placement information across different Clouds. In [20], Lim et al. proposed a Cloud Resource Estimation Module based on service latency information. A highly accurate service latency prediction mechanism was derived. Their designed system could provide a framework which facilitated service latency information collection for better cloud service management. It aimed to use service latency information to provide fast response for various delay-sensitive cloud services.

2.3 Cost Effective Analyses for Cloud Computing

In additional, some existing researches focus on the issue of cost-benefit analysis in cloud computing. In [21], Selvarani and Sadhasivam discussed a job grouping algorithm which was used to allocate the task-groups to different available resources. This scheduling algorithm measured both resource cost and computation performance, it also improved the computation/communication ratio by grouping the user tasks according to a particular cloud resource's processing capability and sent the grouped jobs to the resource.

A cost-based resource scheduling paradigm was presented in [22] by leveraging market theory to schedule compute resources and meet user's requirement. The set of computing resources with the lowest price were assigned to the user according to current suppliers' resource availability and price. An algorithm and protocol were designed for cost-based cloud resource scheduling. The scheduling algorithm and protocol were described in the pure Java based platform, which had three-tiered hierarchical and extensible architecture.

In [23], a minimum cost maximum flow algorithm was proposed for resources (e.g. virtual machines) placement in clouds. Hadji and Zeghlache focused on the optimal dynamic placement of virtual resources in data centers and cloud infrastructures to serve multiple users and tenants with time varying demands and workloads. Providers could use the minimum cost maximum flow algorithm to opportunistically select the most appropriate physical resources.

In [24], Hwang et al. presented a two-phase algorithm for service operators to minimize their service provision cost. In the first phase, a mathematical formula was proposed to compute the optimal amount of long-term reserved resources. In the second phase, the Kalman filter was used to predict resource demand and adaptively change the subscribed on-demand resources such that provision cost could be minimized. They exploited a predictive-based resource management to adaptively configure VMs. To the best of our knowledge, the problems of achieving a cost-effective cloud system with finite buffer analyses and impatient job concerns have not been studied.

3 Model Description

3.1 A Multi-Servers System with Blocking Control

We consider a cloud server farm with a finite buffer and model it as an M/M/R/K queuing system. The mathematical expressions are stated in detail as follows.

There have R identical servers and at most K user requests are allowed in the system. User requests arrive from an infinite source with a mean arrival rate λ and follow a Poisson distribution [25], while service times have an exponential distribution with parameter μ.

The first-come-first-served (FCFS) queuing discipline is adopted and let the states n (n= *0, 1, 2,..., K*) represent the number of user requests in the system. The value of the request arrival rate and the service rate are taken to be

$$\lambda_n = \begin{cases} \lambda, & 0 \le n \le K-1, \\ 0, & n \ge K, \end{cases} \qquad \mu_n = \begin{cases} n\mu, & 1 \le n \le R-1, \\ R\mu, & R \le n \le K, \end{cases} \tag{1}$$

The probability of n user requests that are being served in the system is denoted by P_n. In a steady state, the probability functions P_n can be obtained from the birth-and-death formula. According to the value n may happen, two segments are defined by the vector: [Segment 1, Segment 2] = [$1 \le n \le R-1$, $R \le n \le K$]. With expressions in Eq. (1), the initial state probability functions P_n can be derived as follows:

$$P_n = \begin{cases} \dfrac{\lambda^n}{n!\ \mu^n} P_o & 1 \le n \le R-1 \\ \dfrac{\lambda^n}{R!\ R^{n-R}\ \mu^n} P_o. & R \le n \le K \end{cases} \tag{2}$$

To obtain P_o, Eq. (2) and Eq. (3) are brought into the normalizing equation:

$$\sum_{n=0}^{K} P_n = 1 \tag{3}$$

, and the steady-state probability of zero service P_o can be obtained as follows:

$$\sum_{n=0}^{K} P_n = P_o + \sum_{n=1}^{R-1} P_n + \sum_{n=R}^{K} P_n \tag{4}$$

$$P_o = \left[\sum_{n=0}^{R-1} \frac{\lambda^n}{\mu^n n!} + \sum_{n=R}^{K} \frac{\lambda^n}{\mu^n R!\ R^{n-R}}\right]^{-1}. \tag{5}$$

System utilization is equal to $\rho=\lambda/R\mu$, the steady-state solution always exists for all positive value of λ and μ, but when $\lambda > R\mu$, the number of user requests will be restricted within K in the system since there has no buffer (waiting space).

3.2 A Finite Buffer Queue with Impatient Users

There are various balking and reneging rules discussed in previous works. The queuing length and waiting time are usually the main factors to affect user balking and reneging behaviors, respectively. As illustrated in Fig. 2 (a), the system blocking loss (denoted by K) can be reduced by expanding the buffer size. However, a larger waiting buffer will lead to a long queue and waiting time during peak load, which directly result in a higher balking rate (denoted by B) and reneging rate (denoted by R), respectively. On the contrary, balking and reneging rates can be lessened by reducing the buffer size; however, it will lead to more system blocking loss, as depicted in Fig. 2 (b).

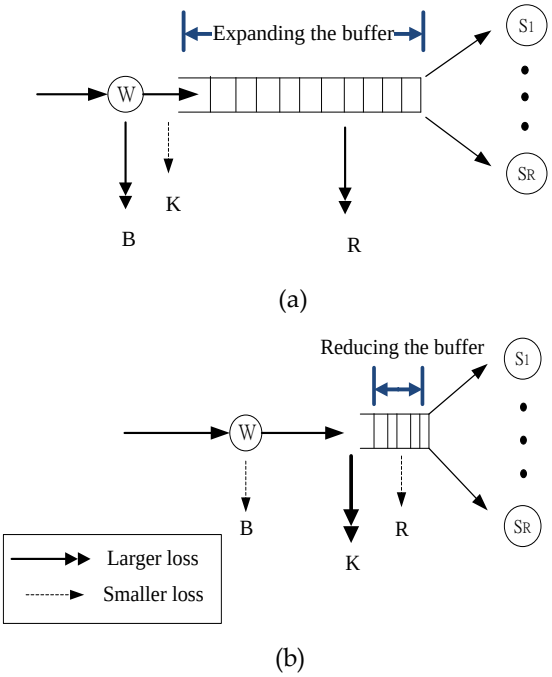

Fig. 2 (a) The effect of expanding the buffer size (b) The effect of reducing the buffer size on system losses.

Since a user whose job request is blocked will simply be dropped, a cloud provider should keep system losses as low as possible to satisfy performance guarantees. Fig. 3 shows the proposed cloud service model which is comprised of R servers with a finite buffer size, denoted by β. Dynamic system controls are used to alleviate system losses and deal with a widely varying load. Due to the fact that a system loss at a front stage directly affects subsequent performances, the request arrival rate at the middle node (MN) queue and the final system throughput are evaluated by taking into account resources provisioning, related performance factors and potential user behaviors in the following.

3.3 Balking Probability

For the purpose of delivering quality service [26], [27], the delay information about the predicted queuing length is sent to inform each arrival user [28]. By using Eq. (5), the expected queuing length L_q can be obtained as:

$$L_q = \sum_{n=R+1}^{K} (n-R) \cdot P_n = \frac{P_o \cdot (\lambda/\mu)^R \cdot \rho}{R!} \cdot \sum_{n=R+1}^{K} (n-R) \cdot \rho^{n-R-1}. \tag{6}$$

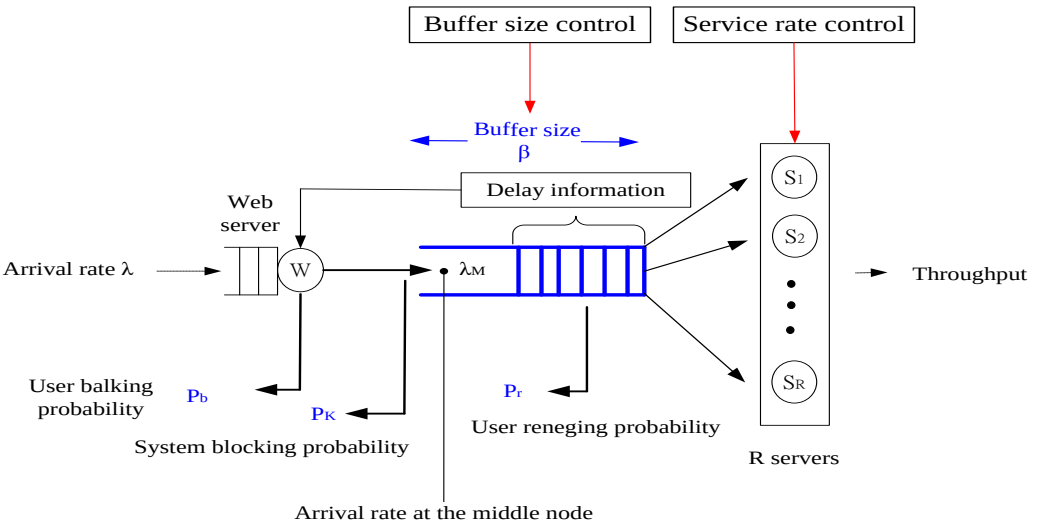

Fig. 3 A cloud service model with a finite buffer and impatient users

Few arrival users may decide not to join the queue and leave the system when the queuing length is too long to be accepted. The severity of balking and reneging rates at the end of per planning period will be recorded in this system. The corresponding notations used in this paper are listed in Table 1.

TABLE 1

List of Key Notations

Notation	Description
U_b	Potential balking factor according to historical data.
U_r	Potential reneging factor according to historical data.
$P_{b,}$	Potential balking probability of which would be expressed as a function of U_b and L_q.
P_r	Potential reneging probability of which would be expressed as a function of U_r and W_q.
P_K	Blocking probability when the system capacity is K.
λ_b	Mean balking rate.
λ_r	Mean reneging rate.
λ_K	System blocking rate when the system capacity is K.
λ_M	Arrival rate at the middle node (MN) depended on λ_b and λ_r.
λ_F	Final throughput rate.
P_L	System loss probability.

By taking λ_b divided by the mean queuing length and initial arrival rate, the potential balking factor, denoted by U_b can be obtained as follow:

$$U_b = \frac{\lambda_b}{L_q \cdot \lambda}. \tag{7}$$

The balking probability can be calculated by multiplying the mean queuing length and its potential balking factor together as below:

$$P_b = L_q \cdot U_b. \tag{8}$$

Based on the same service type at a subsequent period [29] with an arrival rate λ, the mean balking rate can be obtained as follow.

$$\lambda P_b = \lambda_b \tag{9}$$

3.4 System Blocking Probability

The blocking probability means that user requests cannot be retained in the queue due to lack of waiting-space when all servers are busy. That is, user requests are allowed to enter and obtain service if the buffer hasn't been completely occupied. The subsequent request arrival rate is $(\lambda - \lambda_b)$ after excluding the balking loss. Since the controlled system capacity in the proposed system is K, the blocking probability can be calculated by using Eq. (5) as below:

$$P_K = \frac{(\lambda - \lambda_b)^K}{R! \cdot R^{K-R} \cdot \mu^K} P_o. \tag{10}$$

Then the system blocking rate, denoted by λ_K can be obtained as follow.

$$\lambda_K = (\lambda - \lambda_b) \cdot P_K. \tag{11}$$

According to the user balking rate and the system blocking rate, the request arrival rate at the MN queue, denoted by λ_M can be given as follow.

$$\lambda_M = \lambda - \lambda_b - \lambda_K. \tag{12}$$

3.5 Reneging probability

Based on the rate of λ_M, the system performances at the MN queue, denoted by P_0^*, P_n^* and the mean queuing length L_q^* can be calculated by using Eq.(1)–Eq. (5) .

$$L_q^* = \frac{P_0^* \cdot (\lambda_M / \mu)^R \cdot \rho^*}{R!} \cdot \frac{d}{d\rho^*}\left(\frac{1-\rho^{*K-R+1}}{1-\rho^*}\right). \quad (13)$$

where $\rho^* = \frac{\lambda_M}{\mu \times R}$.

To find the predicted waiting time W_q^* at the MN queue, the well-known Little's law is applied [25]. It states that the average number of items waiting to receive service is equal to the average arrival rate multiplied by the mean time. Historically, it has been written as

$$L=\lambda W \quad (14)$$

Then, the waiting time at the MN queue can be obtained as

$$W_q^* = \frac{L_q^*}{\lambda_M}. \quad (15)$$

Dividing the mean recorded reneging rate λ_r by the mean waiting time and the arrival rate at the MN queue, the potential reneging factor can be obtained as follow.

$$U_r = \frac{\lambda_r}{W_q^* \cdot \lambda_M}. \quad (16)$$

Similarly, the reneging probability can be calculated by multiplying the mean waiting time and the potential reneging factor together as below:

$$P_r = W_q^* \cdot U_r \quad (17)$$

Then, the expected reneging rate in the queue can be obtained as follow:

$$\lambda_M P_r = \lambda_r \quad (18)$$

For the system with finite waiting buffer and impatient users, the final throughput rate, denoted by λ_F, is calculated as below:

$$\lambda_F = \lambda - \lambda_b - \lambda_K - \lambda_r. \quad (19)$$

After excluding all system losses, the final system utilization can be obtained as follow:

$$\rho_F = \frac{\lambda_M - \lambda_r}{R\mu}. \quad (20)$$

Hence, the system loss probability can be estimated as:

$$P_L = \frac{\lambda - \lambda_F}{\lambda}. \quad (21)$$

4. Cost analyses and the CEA policy

4.1 A Cost Model

In a cloud system, the major operational costs of resources provisioning (incurred by server quantity, power consumption and buffer capacity), system losses (incurred by impatient users, system blocking and activating VMs) and performances (incurred by system rejection penalty and system congestion) are all taken into account, as shown in Fig. 4.

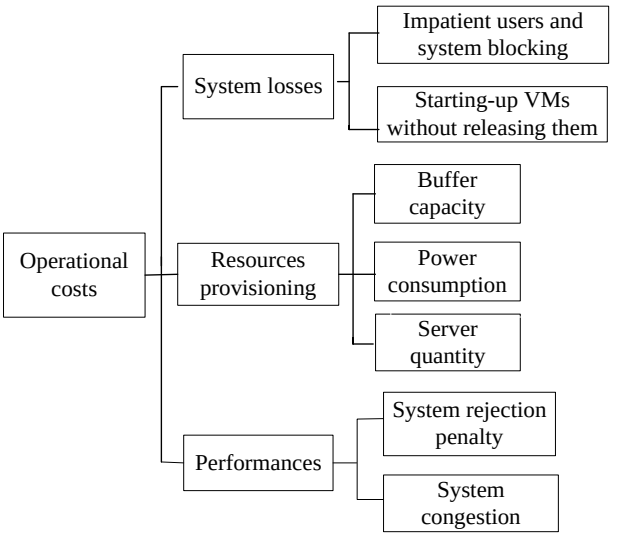

Fig. 4 The major operational costs in a cloud system.

In general, virtual machines (VMs) will be activated when a job request has been accepted and forwarded into the buffer. However, some impatient users will renege after entering the queue and abandon the VMs immediately. Therefore, the specific problem of cost overhead for activating VMs but without releasing them is required to be evaluated for a system with impatient users. The descriptions of cost notations are summarized as follows.

C_1≡ Expected server provisioning cost per server per unit time;
C_2≡ Expected power consumption cost per service rate per unit time;
C_3≡ Expected cost incurred by preparing per buffer space per unit time;
C_4≡ Impatient users and system blocking losses incurred by per request;
C_5≡ Starting-up cost incurred by activating per VM;
C_6≡ System rejection penalty;
C_7≡ Cost incurred by holding jobs in the system per unit time;
C_8≡ Cost incurred by jobs waiting in the system per unit time;

Since system performances, loss probability and operational costs strongly depend on the buffer space and the service rate, an expected cost function per unit time is developed in which both the service rate and the buffer size are the main decision variables. Apparently, no users want to be blocked or abandon service due to inadequate buffer space or intolerable system delay, respectively. Hence, there should has a loss probability guarantee in a service system, which is also perceived as one of the most important performance concerns to measure service levels [30]. Here, the SLA constraint is specified by guaranteeing: loss probability ≤ x%, where x is the maximum threshold value, denoted by SLA (x%). The cost minimization (CM) with a loss probability guarantee can be rep-

resented mathematically as

Minimize CM

Subject to

$$0 \le P_L < x$$

Where $CM = F(\mu, \beta)$

$$= (RC_1+\mu C_2)/\rho_F + \beta C_3 + (\lambda r+\lambda_b+\lambda_K)C_4 + \lambda r C_5 + P_K C_6 + L_q C_7 + W_q^* C_8 \quad (22)$$

4.2 The Proposed CEA Policy

Here, the designed **C**ost-**E**ffective policy in an **A**bandoned system (CEA policy) is presented to address the optimal solution of (μ, β), say (μ^*, β^*), so as to minimize the operational cost without violating a SLA constraint. It focuses on solving the contradictory problem among reducing system losses and conserving operational cost. However, it is extremely difficult to obtain the analytical result of the optimal solution due to the fact that the cost function is highly nonlinear and complex. Instead, we present the CEA heuristic algorithm to find the minimum total cost by solving nonlinear constrained optimization problems under various incurred costs and user behavior variations.

CAE heuristic algorithm

Input Data:
1. Arrival rate λ.
2. Potential balking and reneging factors $[U_b, U_r]$.
3. Cost matrix $[C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8]$.
4. The number of servers R.
5. The upper bound of service rate and buffer size in the cloud server farm, denoted by μ_p and β_d.
6. The loss probability guarantees x in the SLA constraint.

Output: μ^*, β^* and $F(\mu^*, \beta^*)$

Step1. For $i = 1; i = p; i$++
  Set $\mu_i \leftarrow$ current service rate;
  For $j = 1; j = d; j$++
  Set $\beta_j \leftarrow$ current buffer size;
Step2. Calculate L_q and λ_b using Eq. (5) - Eq. (9)
  Calculate P_K and λ_K using Eq. (2) and Eq. (10-12)
  $\lambda_M \leftarrow \lambda - \lambda_b - \lambda_K$;
Step3. Set $\lambda_M \leftarrow$arrival rate at the middle node queue;
  Calculate L_q^*, W_q^* and λ_r using Eq. (13)-Eq. (19)
  $\lambda_F \leftarrow \lambda_M - \lambda_r$;
Step4. Set $\lambda_F \leftarrow$expected final throughput rate;
  Calculate ρ_F using Eq. (20)- Eq. (21)
  $P_L \leftarrow (\lambda - \lambda_F)/\lambda$;
  If $P_L < x$, then
  Bring all cost parameters into the developed cost model and begin to calculate $F(\mu_i, \beta_j)$
  Else
  Return to step 1 and begin to test a next index.
  End
Step5. If the joint value of (μ_i, β_j) can obtain the minimum cost value in all tests, then,
  Output (μ_i, β_j) and $F(\mu_i, \beta_j)$
Else
  Return to step 1 and begin to test a next index.
End

5 NUMERICAL VALIDATION

5.1 System Performances

To gain more insight into the designed system behavior, first of all we provide several experiments to observe the effect of resources provisioning on performances. Numerical simulations are demonstrated by assuming λ=2500 request/min, $U_r = U_b$=0.01, R=64 and the buffer size is made variable from 0, 16 to 32 in three steps. All computational programs are developed by using MATLAB.

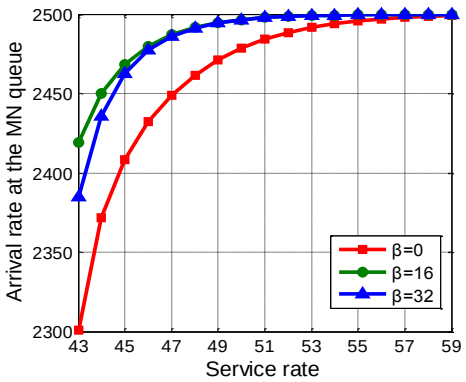

Fig. 5 Arrival rate at the MN queue under various μ and the given β values.

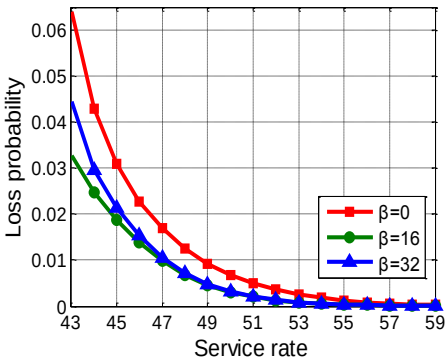

Fig. 6 Loss probability under various μ and the given β values.

Fig. 5 and Fig. 6 demonstrate the variation of the request arrival rate at the MN queue and the loss probability distribution under different service rates and buffer sizes, respectively. It's observed that increasing the service rate can certainly improve the arrival rate at the MN queue. However, increasing the buffer size does not necessarily improve the arrival rate at the MN queue. It can be seen that it will cause the highest loss probability if there has no buffer available and it does not contribute to reducing the loss probability through buffer over-provisioning. Both performance gaps between different β values become smaller and converge to nearly equal as

the service rate further increases.

5.2 Experimental Results

The experiments have been conducted to validate that the optimal resources provisioning can be obtained by applying the heuristic algorithm and show that the CEA policy is practical. It's assumed that λ=5200 request/min, U_r=U_b=0.006, R=100, while the loss probability constraint is *0.5%*, denoted by SLA(*0.5%*). Since a server provisioning cost is mostly determined by the rent/purchase cost and power consumption cost, the server provisioning cost can be roughly estimated according to different requirements.

Here, we assume [C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8] = [200, 30, 20, 60, 50, 10, 5, 5] in experiments. The effect of varying μ and β values to find the minimum cost is shown in Fig. 7. It's noted that the minimum cost of *27544.85* can be obtained at the optimal solution (μ^*, β^*) = (*62, 12*). The loss probability distributions under various μ and β values are demonstrated in Fig. 8. The effect of the service rate on the loss probability variation is larger than the buffer size. As can be seen, reducing the loss probability at beginning can lower the cost (corresponds to Fig. 7). However, as the loss probability further reduces, it leads to no more cost reduction.

This behavior is due to the fact that, maintaining an extremely low loss probability requires more resources provisioning, which directly results in high cost burden. The corresponding loss probability at the optimal solution is *0.29%*. Simulation results have verified that the system can satisfy the SLA constraint and simultaneously obtain the minimum operational cost by applying the CEA policy. In the next experiments, system blocking probabilities under various μ and β values are shown in Fig. 9. As can be expected, the system blocking probabilities can be reduced by increasing either the service rate or the buffer size. The lower blocking probability of *0.1%* can be obtained at the optimal solution. After excluding the balking and the system blocking rate, the remaining arrival rates at the MN queue is shown in Fig. 10. The effect of the buffer size on the arrival rate at the MN queue is larger when the service rate is low; however, it becomes virtually undetectable when the service rate is high.

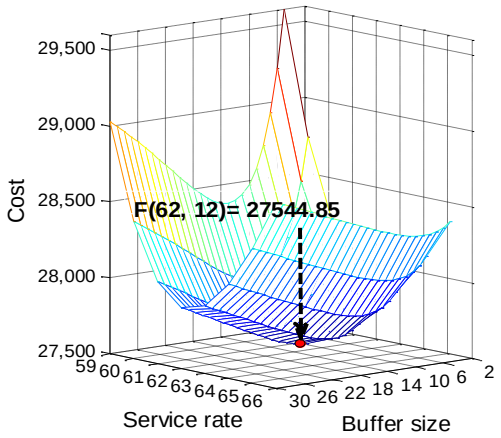

Fig. 7 Cost distributions under various μ and β values.

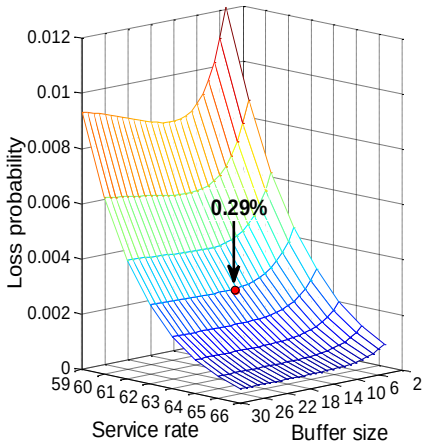

Fig. 8 Loss probability distributions under various μ and β values.

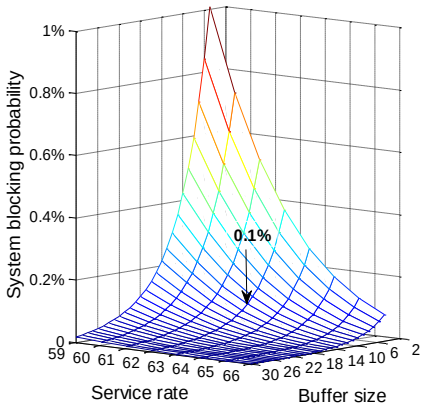

Fig. 9 System blocking probabilities under various μ and β values.

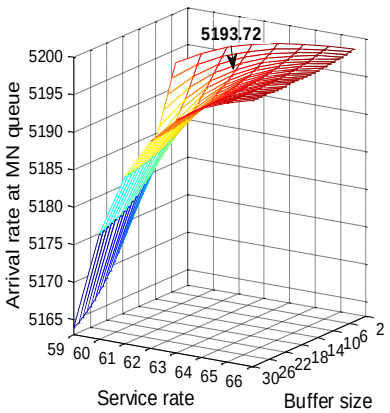

Fig. 10 Arrival rates at the MN queue under various μ and β values.

The higher arrival rates at the MN queue of *5193.72* also can be achieved by adopting the CEA policy. The reneging rate and the system throughput rate distributions are shown in Fig. 11 and Fig. 12, respectively. It's noted

that the reneging rate can be reduced by increasing the service rate or lessening the buffer size. Besides, the impact of the buffer size becomes virtually undetectable when the system operates at a higher service rate. The same tendency can be found in the throughput rate, as shown in Fig. 12.

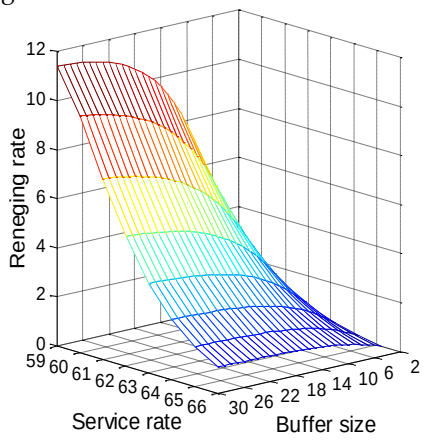

Fig. 11 Reneging rates under various μ and β values

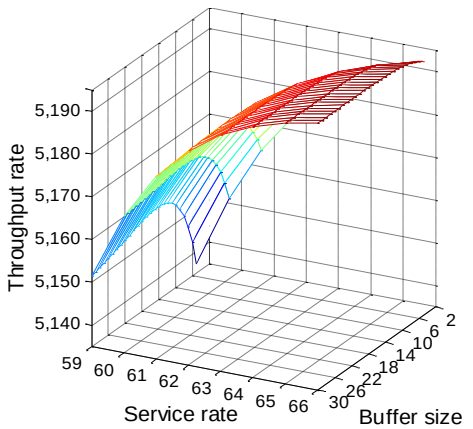

Fig. 12 Throughput rates under various μ and β values.

5.3 Comparison of Results

A general approach, which implies that a solution is calculated only by considering an absolute performance guarantee is used as a basis for comparison since most of the previous works [31], [32], [33] had adopted this approach to manage cloud resources. For the sake of simplicity, here it's referred to as a non-CEA policy since no system loss evaluations or cost optimization analyses are considered. Next, we try to show that the CEA policy is also applicable for a system with a fixed buffer size. Experiments are conducted by assuming that λ=500 request/min, R=20 and both policies need to comply with the same loss probability guarantee of SLA(5%).

Naturally, users react variously to different degrees of latency. In additional, many potential factors such as individual feelings, satisfaction, delay tolerant, etc. cannot be ignored since it may also influence user's decision. In the designed system, the abandonment information will be recorded at per planning period. For an existing cloud computing service, the balking and reneging potential factors can be obtained from an actual historical statistic. Here both parameters are randomly chosen and three different cases are performed. Both policies are evaluated by assuming potential balking/reneging factors (U_b; U_r) = (0.005; 0.005), (0.005; 0) and (0; 0.005) in order to study and compare the influence of different impatient situations on the operational costs and performances. Besides, different buffers of size 1, 10 and 20 are assigned.

- (U_b; U_r) = (0; 0.005), in order to study the behavior of both policies when a system without balking but has the threat of reneging users.
- (U_b; U_r) = (0.005; 0), in order to study the behavior of both policies when a system without reneging but has the threat of balking users.
- (U_b; U_r) = (0.005; 0.005), in order to study the behavior of both policies when there have both threats of balking and reneging users in the system.

Comparisons of the controlled service rates are shown in Fig. 13. As can be seen, it will result in a higher service rate for reducing system losses when the given buffer size is less. The results show that the service rates determined by the non-CEA policy are lower than the CEA policy since the former tries to reduce resources provisioning cost as more as possible. However, the lower operational cost can be achieved by applying the CEA policy, as shown in Fig. 14. It's noted that the system will result in higher cost under a larger given buffer size when there has the threat of reneging.

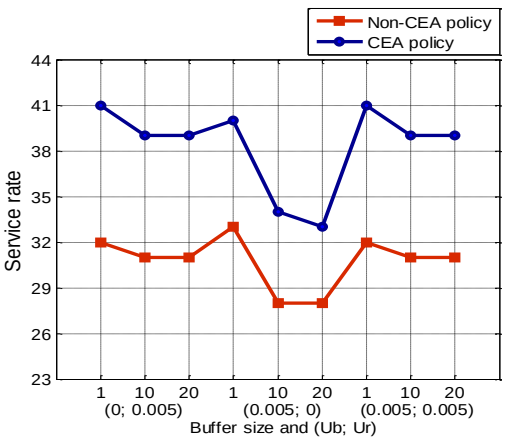

Fig. 13 Comparison of the service rate.

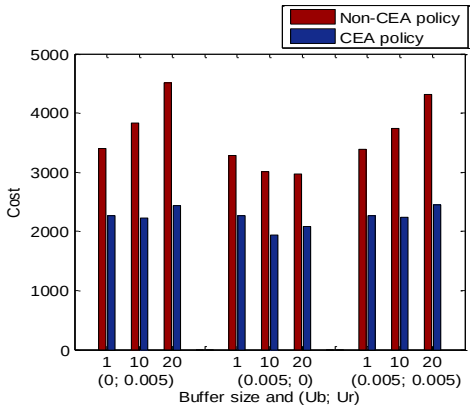

Fig. 14 Comparison of the operational cost.

Nevertheless, the costs can be reduced by applying the CEA policy and they also can be maintained relatively stable as compared to the non-CEA policy. Comparisons of the throughput rate are shown in Fig. 15. It's noted that the throughput rate can be improved significantly by applying the CEA policy.

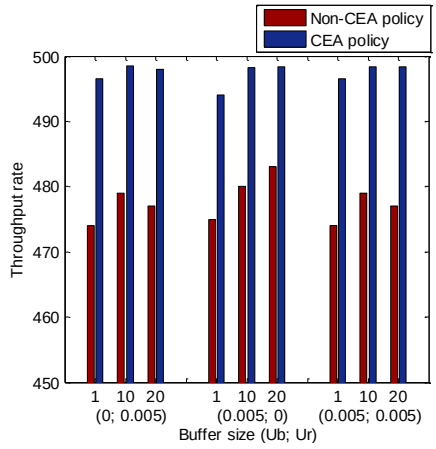

Fig. 15 Comparison of the throughput rate.

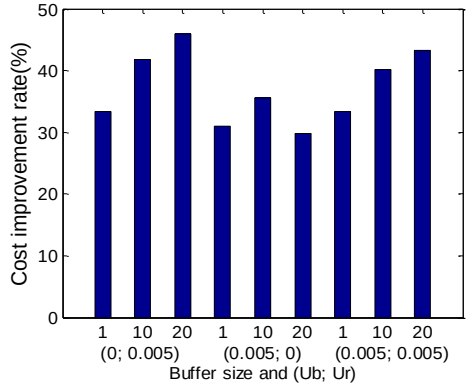

Fig. 16 Cost improvement rate.

Finally, we measure the cost improvement ratio, which calculates the relative value of improvements to the original value instead of the absolute value; the results are shown in Fig 16. The relative improvement rate is up to 46% in terms of cost reduction. The comparison of results has shown that more cost-saving and throughput rate enhancement can be achieved by applying the CEA policy.

6 CONCLUSION AND FUTURE WORK

Developing a successful service system necessitates taking into account not only system control factors but also user behaviors. However, most existing studies fail to offer an effective system control to capture optimization opportunities when facing impatient users and various incurred costs. To tackle the problem, the effect of resources provisioning on system losses and the throughput rate are studied in our work. A cost model is developed to conduct the costs/performances tradeoff according to the incurred costs, system losses, resources provisioning and system performances.

The proposed CEA policy contributes to addressing the optimal service rate and buffer size controls in a system with a finite buffer and impatient users. Experiment results have shown that realizing cost-effective resources provisioning within a loss probability guarantee can be obtained by applying the CEA policy. As compared to a system without applying our approach, the benefit of reducing cost is up to 46 percent. As for future works, we plan to analyze more challenging issues such as traffic load control mechanisms, finite population concerns, etc. for conducting a comprehensive control strategy.

REFERENCES

[1] S. Sivathanu, L. Liu, M. Yiduo, and X. Pu, "Storage management in virtualized cloud environment," *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, pp. 204-211, 2010.

[2] R. Zhang, and L. Liu, "Security models and requirements for healthcare application clouds," *IEEE 3rd International Conference on Cloud Computing (CLOUD)*, pp. 268-275, 2010.

[3] K. Wang, N. Li, and Z. Jiang, "Queueing System with Impatient Customers: A Review," *IEEE International Conference on Service Operations and Logistics and Informatics (SOLI)*, pp. 82-87, 2010.

[4] J. Li, T. Dai, J. Huo, and Q. Su, "A Method of Service Quality Monitoring in Contact Centers with Impatient Customers," *9th International Conference on Service Systems and Service Management (ICSSSM)*, pp. 114-117, 2012.

[5] N. Mehdi, A. Mamat, H. Ibrahim and S. Symban, "Virtual machines cooperation for impatient jobs under cloud paradigm," *International Journal of Information and Communication Engineering*, vol. 7, no. 1, pp.13-19, 2011.

[6] C. Cardonha, M. D. Assunção, M. A. Netto, R. L. Cunha, and C. Queiroz, "Patience-aware scheduling for cloud services: Freeing users from the chains of boredom," *Springer Berlin Heidelberg In Service-Oriented Computing*," pp. 550-557, 2013.

[7] M. Mazzucco and D. Dyachuk, "Balancing electricity bill and performance in server farms with setup costs," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 415-426, 2012.

[8] N. A. Mehdi, A. Mamat, H. Ibrahim, and S. K. Subramaniam, "Impatient task mapping in elastic cloud using genetic algorithm," *Journal of Computer Science*, 7(6), 877, 2011.

[9] T. Keskin, and N. Taskin, "A pricing model for cloud computing service," *Hawaii International Conference on System Sciences (HICSS)*, pp. 699-707, 2014.

[10] K. M. Sim, "Agent-Based Cloud Computing," *IEEE Transitions on Services Computing*, vol. 5, no. 4, pp. 564-577, 2012.

[11] Y. J. Chiang and Y. C. Ouyang, "Profit Optimization in SLA-Aware Cloud Services with a Finite Capacity Queuing Model." *Mathematical Problems in Engineering*, 2014.

[12] Y. Q. Zhao and A. S. Alfa, "Performance analysis of a telephone system with both patient and impatient customers," *Telecommunication Systems*, pp. 201-215, 1995.

[13] A. Mandelbaum and S. Zeltyn, "The impact of customers' patience on delay and abandonment: some empirically-driven experiments with the M/M/n+G queue," *OR Spectrum*, vol. 26, no. 3, pp. 377-411, 2004.

[14] D. Doran, L. Lipsky and S. Thompson, "Cost-based Optimization of Buffer Size in M/G/1/N Systems Under Different Service-time Distributions," *Proceedings of 9th IEEE Network Computing and Applications (NCA)*, pp. 28-35, 2010.

[15] A. P. Ghosh, and A. P. Weerasinghe, "Optimal buffer size and dynamic rate control for a queueing system with impatient customers in heavy traffic," *Stochastic Processes and their Applications*, pp. 2103-2141, 2010.

[16] B. Yang, F. Tan, Y. S. Dai, and S. Guo, "Performance Evaluation of Cloud Service Considering Fault Recovery," *Cloud Computing. Springer Berlin Heidelberg*, pp. 571-576, 2009.

[17] H. Khazaei, J. Misic, and V. B. Misic, "Performance Analysis of Cloud Computing Centers Using M/G/m/m+r Queuing Systems," *IEEE Transactions on Parallel and Distributed Systems*, pp. 936-943, 2012.

[18] J. Cappos, I. Beschastnikh, A. Krishnamurthy, and T. Anderson, "Seattle: a platform for educational cloud computing," *ACM SIGCSE Bulletin*, vol. 41, no. 1, pp. 111-115. 2009.

[19] C. Shouraboura and P. Bleher, "Placement of applications in computing clouds using Voronoi diagrams," *Journal of Internet Services and Applications*, vol. 2, no.3, pp. 229-241, 2011.

[20] B. P. Lim, P. K. Chong, E. K. Karuppiah, Y. M. Yassin, A. Nazir,and M. F. N. Batcha, "FARCREST: Euclidean Steiner Tree-based cloud service latency prediction system," *Consumer Communications and Networking Conference (CCNC)*, pp. 665-668, 2013.

[21] S. Selvarani and G. S. Sadhasivam, "Improved cost-based algorithm for task scheduling in cloud computing," *IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pp.1-5, 2010.

[22] Z. Yang, C. Yin, and Y. Liu, "A Cost-based Resource Scheduling Paradigm in Cloud Computing," *12th International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 417- 422, 2011.

[23] M. Hadji and D. Zeghlache, "Minimum Cost Maximum Flow Algorithm for Dynamic Resource Allocation in Clouds," *IEEE Fifth International Conference on Cloud Computing*, pp.876-882, 2012.

[24] R. Hwang, C. Lee, Y. Chen, and J. D. Zhang, "Cost Optimization of Elasticity Cloud Resource Subscription Policy," *IEEE Transition on Services Computing*, 2013.

[25] D. Gross, J. F. Shortle, J. M. Thompson and C. M. Harris. Fundamentals of Queuing Theory (Fourth edition), *A John Wiley & Sons, Inc., New York*, 2008.

[26] J. Cao et al., "Social attribute based web service information publication mechanism in Delay Tolerant Network," *IEEE 14th International Conference on Computational Science and Engineering (CSE)*, pp. 435- 442, 2011.

[27] M. Armony and C. Maglaras, "Contact Centers with a Call-Back Option and Real-Time Delay Information," *Operations Research*, vol. 52, no. 4, pp. 527–545, 2004.

[28] P. Guo and P. Zipkin, "Analysis and Comparison of Queues with Different Levels of Delay Information," *Management Science*, vol. 53, no. 46, pp. 962–970, 2007.

[29] Amazon EC2 On-Demand Instance Prices, http://aws.amazon.com/ec2/pricing/.

[30] M. Y. Luo and C. S. Yang, "Constructi.ng zero-loss Web services," IEEE Computer and Communications Societies Conference, pp.1781-1790, 2001.

[31] J. Shao, and Q. Wang, "A performance guarantee approach for cloud applications based on monitoring," *IEEE 35th Annual Computer Software and Applications Conference Workshops (COMPSACW)*, pp. 25-30, 2011.

[32] R. Nathuji, A. Kansal, and A. Ghaffarkhah, "Q-clouds: managing performance interference effects for qos-aware clouds," *ACM Proceedings of the 5th European conference on Computer systems*, pp. 237-250, 2010.

[33] R. N. Calheiros, R. Ranjan,and R. Buyya, "Virtual machine provisioning based on analytical performance and QoS in cloud computing environments," *International Conference on Parallel Processing (ICPP)*, pp. 295-304, 2011.

Yi-Ju Chiang received her BS and MS degrees from the Department of Electrical Engineering (EE) at National Chung-Hsing University of Taiwan in 2011 and 2013, respectively. She is currently working toward the PhD degree in Department of Electrical Engineering (EE) at National Chung-Hsing University. Her research interests include cloud computing, optimal control algorithm, performance evaluation, queuing theory and green computing system. She is an IEEE student member.

Yen-Chieh Ouyang (S'86–M'92) received the BSEE degree in 1981 from Feng Chia University, Taiwan, and the MS degree in 1987 and the PhD degree in 1992 from the Department of Electrical Engineering, University of Memphis, Memphis, Tennessee. He joined the Faculty of the Department of Electrical Engineering at National Chung Hsing University, Taiwan, in August 1992. He currently is a professor and the department chair in the Department of Electrical Engineering, NCHU. His research interests include cloud computing, hyperspectral image processing, medical imaging, communication networks, network security in mobile networks, multimedia system design, and performance evaluation. He is a member of the IEEE.

Professor Ching-Hsien (Robert) Hsu is a professor in department of computer science and information engineering at Chung Hua University, Taiwan; and distinguished chair professor in school of computer and communication engineering at Tianjin University of Technology, China. His research includes high performance computing, cloud computing, parallel and distributed systems. He has published 200 papers in refereed journals, conference proceedings and book chapters in these areas. He has been involved in more than 100 conferences and workshops as various chairs and more than 200 conferences/workshops as a program committee member. He is an IEEE senior member.