PERFORMANCE EVALUATION AND NASH EQUILIBRIUM OF
A CLOUD ARCHITECTURE WITH A SLEEPING MECHANISM
AND AN ENROLLMENT SERVICE

SHUNFU JIN AND HAIXING WU
School of Information Science and Engineering, Yanshan University
Qinhuangdao 066004, China

WUYI YUE
Department of Intelligence and Informatics, Konan University
Kobe 658-8501, Japan

YUTAKA TAKAHASHI
Graduate School of Informatics, Kyoto University
Kyoto 606-8225, Japan

(Communicated by Chao Xu)

ABSTRACT. Cloud computing makes it possible for application providers to
provide services seamlessly and application users to receive services adaptively.
By offering services that give users an initial experience, application providers
can usually attract more users. This research proposes a type of sleeping
mechanism-based cloud architecture where an experience service and an en-
rollment service are provided on one virtual machine (VM). Accordingly, we
model the cloud architecture as a queue with an asynchronous multi-vacation
and a selectable extra service. We also analyze the queueing model in the
steady state by constructing a three-dimensional Markov chain. Following
this, we evaluate the system performance of the proposed cloud architecture
based on the energy conservation level of the system and the mean delay of
the visitors who select the enrollment service. Moreover, we study the Nash
equilibrium strategy of visitors by building an individual welfare function, and
develop an improved intelligent search algorithm to investigate the socially op-
timal strategy of visitors. Aiming to achieve a social optimum, we formulate a
pricing policy with a reasonable enrollment fee.

1. Introduction. Cloud computing systems promise subscription-oriented com-
puting services to global users [2], [10]. Each type of cloud service has its own
advantages and disadvantages. How to choose an appropriate service is particularly
important for potential cloud users [7]. To attract more users, application providers
always offer an experiential service to visitors. A visitor, if they are pleased with
the service experience and want better service in future, is likely to upgrade as a
member.

Under-utilized energy consumption in cloud computing systems account for a
large proportion of actual energy consumption [3]. Therefore, it is an important
challenge for application providers to offer an enrollment service with a green cloud architecture.

Given the fierce competition in the cloud environment, there is an increasing number of studies on how to reasonably build a cloud architecture that attracts more cloud users. Shi et al. proposed a structure named JointCloud Corporation Environment (JCCE), in which a mutual welfare and win-win JointCloud environment for global cloud service providers is offered [17]. Guo et al. described the Policy-Based Market-Oriented Cloud Service Management Architecture (PBMOCMSA), showing how to offer flexible, dynamic and extensible policy-based management capabilities to application providers [8]. In [12], Jin et al. proposed a cloud architecture with a free service and a service that requires registration. With this architecture, all the VMs remain awake and are always ready to provide a timely free service and a registration service to cloud users.

With growing interest in cloud computing and carbon emissions reduction, the need to develop strategies for building energy efficient cloud architecture is becoming more pressing. Ye et al. presented a virtual machine (VM) based energy-efficient data center architecture for cloud computing [22]. They investigated the potential performance overheads caused by server consolidation and live migration of VM technology. Hu et al. proposed a globally collaborative mechanism known as the Green Private Cloud Computing (GCMGPC) and built a Green Private Cloud Architecture model with virtualization technology [9].

However, in all of these studies, how to achieve a green cloud environment at the same time as cultivating powerful groups of loyal cloud users has not been considered. We know that the design and development of a competitive cloud system requires not only improving the system’s energy efficiency, but also attracting more cloud users [1]. Therefore, when constructing such cloud systems, researchers need to consider how to achieve these twin aims.

In implementing the sound operation of cloud systems, a numerical evaluation of the system performance is needed. A queueing model with a selectable extra service is appropriate to capture the enrollment service in cloud systems. Madan first proposed a queueing model with a selectable extra service [15]. Following Madan’s formulation, numerous papers on queueing models with a selectable extra service were published. Singh et al. considered a single server queue with a selectable extra service [18]. In [18], the time duration of the first service followed a general distribution, and the time duration of the selectable extra service followed an exponential distribution. Rad et al. considered a two-phase tandem queue with a selectable extra service and a random feedback [6]. Wei et al. studied a Geom/G/1 retrial queue by introducing balking users and a selectable extra service [20].

When modeling the sleeping mechanism, a vacation queue is appropriate. Doshi studied a vacation queue in the 1970s [4], and this was the first time the term “Vacation” appeared in queueing theory. Since then, many papers on vacation models have appeared. Gray analyzed a queue with multiple vacations, where the server may fail in operation [5]. Jain et al. discussed an asynchronous vacation queue aiming to solve practical problems in a multi-server repair system [11].

However, the studies of all the mentioned papers were carried out either on the queueing model with only the selectable extra service, or on the queueing model with only the vacation mechanism. For this reason, these studies could not be directly used to model the stochastic behavior of the networks in an energy efficient cloud architecture with experience services and enrollment services. A new queue should
be studied by considering both of the selectable extra service and the vacation mechanism to capture the stochastic behavior of the cloud systems.

This paper is a substantial and appropriate extension of our previous work \cite{21} appearing in the conference proceedings. In this paper, we propose a type of sleeping mechanism-based cloud architecture, in which each server offers two kinds of services: experience services and enrollment services. If no visitors are waiting in the system buffer, a newly evacuated server will go to sleep. We classify the users into visitors and members. A visitor who demands for an experience service is likely to enroll as a member to receive better service in future. We build a queue with an asynchronous multiple-vacation and a selectable extra service to model the proposed cloud architecture. And then, we construct a three-dimensional Markov chain to analyze the queueing model. In addition, we evaluate the mean delay of the visitors who select the enrollment service and the energy conservation level of the system. We note that, the more that visitors access the system, the higher the mean delay will be, and the lower the Quality of Service (QoS) will be for visitors. The fewer visitors there are accessing the system, the higher the energy conservation level will be, but the profit for the application provider will be lower. Aiming to get a better trade-off between the QoS for visitors and the profit for the application provider, we study the Nash equilibrium strategy of visitors, and then we develop an improved Bat algorithm to discuss the socially optimal strategy of visitors. We propose a pricing policy to align the Nash equilibrium and the socially optimal strategies.

The reminder of this research is presented in 7 sections: In Section 2, a type of cloud architecture with a sleeping mechanism and a selectable extra service is proposed. Accordingly, we build a queue model with an asynchronous multiple-vacation and a selectable extra service. In Section 3, applying the matrix-geometric solution method, the queueing model is derived in the steady state. In Section 4, the mean delay and the energy conservation level are estimated. In Section 5, numerical results are carried out to demonstrate the impact of system parameters on system performance. In Section 6, a pricing policy is presented to maximize the social welfare. Finally, in Section 7, concluding remarks are given.

2. System model. We introduce a sleeping mechanism to a cloud architecture that offers an experience service and an enrollment service. Accordingly, we build a queue model with an asynchronous multiple-vacation and a selectable extra service.

2.1. Sleeping mechanism-based cloud architecture. In general, application providers offer an experience service to attract more visitors. In a traditional cloud environment, VMs remain awake all the time, whether or not some users are receiving service. This leads to a great deal of wasted energy.

Considering the level of energy efficiency and the functioning of enrollment services in cloud systems, we present a sleeping mechanism-based cloud architecture shown in Fig. 1.

In a cloud system, a highly configurable physical machine (PM) is usually composed of several VMs. On each VM, the operating system runs independently, so there is scope for the realization of a sleeping mechanism in a VM.

(1) A visitor entering the system will always firstly queue in the system buffer waiting for the experience service. Once a VM is evacuated, or a VM on any PM wakes up, this VM will be allocated by the task scheduler to the first visitor queueing in the system buffer, and that visitor will just getting the VM will accept the experience service on the cloud system.
Figure 1. Sleeping mechanism-based cloud architecture.

(2) After a visitor completes the experience service, the visitor will decide whether or not to enroll as a member. If the visitor decides to enroll as a member, the visitor must pay the membership dues. Otherwise, the visitor will maintain their original status as a non-member.

(3) When a user leaves a VM, if the system buffer is empty, the evacuated VM will go to sleep. At the instant when a VM begins a sleep period, a sleep timer will be activated. When the sleep timer expires, if the system buffer is empty, the VM will start another sleep period with a new sleep timer. Otherwise, the VM will wake up to serve the visitors queueing in the system buffer.

We next build a queueing model with the proposed architecture to mathematically evaluate and reasonably optimize the system performance.

2.2. Model description. Taking the experience service, the enrollment service, the sleep period and the VM as the essential service, the selectable extra service, the vacation and server, respectively, we build a queue with an asynchronous multiple-vacation and a selectable extra service.

We suppose that the system buffer is infinite, and we denote the total number of VMs in the cloud system as $c$. Also, we denote $N(t) = i$, $i \in \{0,1,2,\ldots\}$ as being the total number of visitors in the system buffer and the VMs, $Y(t) = j$, $j \in \{0,1,2,\ldots,\min(i,c)\}$ as the number of VMs operating at normal speed and $S(t) = k$, $k \in \{0,1,2,\ldots,j\}$ as the number of visitors being enrolled as members. $N(t)$, $Y(t)$ and $S(t)$ are called the system level, the system stage and the system phase, respectively. The working principle of the system model is described in a three-dimensional continuous-time stochastic process $\{N(t), Y(t), S(t), t \geq 0\}$. The state space $\Omega$ of the $\{N(t), Y(t), S(t), t \geq 0\}$ is given as

$$\Omega = \{(i,j,k) \mid i \in \{0,1,2,\ldots\}, j \in \{0,1,2,\ldots,\min(i,c)\}, k \in \{0,1,2,\ldots,j\}\}. \quad (1)$$
Referencing [12], we make the following assumptions to analyze the system model:

- The arrivals of visitors obey a Poisson distribution with arrival rate $\lambda$ ($\lambda > 0$);
- The experience service time and the enrollment service time of a visitor obey exponential distributions with service rates $\mu_1$ ($\mu_1 > 0$) and $\mu_2$ ($\mu_2 > 0$), respectively;
- The probability of a visitor enrolling as a member is $q$ ($0 < q < 1$, $\bar{q} = 1 - q$);
- The time duration of a sleep timer is exponentially distributed with a sleeping parameter $\theta$ ($\theta > 0$).

Based on these assumptions, the stochastic process ${N(t), Y(t), S(t), t \geq 0}$ can be naturally seen as a Markov chain. We give the traffic load $\rho$ of the Markov chain as

$$\rho = \frac{\lambda(\mu_2 + q\mu_1)}{\mu_1 \mu_2}. \quad (2)$$

The Markov chain ${N(t), Y(t), S(t), t \geq 0}$ is stable if and only if $\rho < 1$.

Let $\pi_{i,j,k}$ be the steady-state probability distribution of the Markov chain when the system level is $i$, the system stage is $j$ and the system phase is $k$. $\pi_{i,j,k}$ is given by

$$\pi_{i,j,k} = \lim_{t \to \infty} P\{N(t) = i, Y(t) = j, S(t) = k\}, \ (i, j, k) \in \Omega. \quad (3)$$

Let $\pi_i$ be the steady-state probability distribution when the number of visitors in the cloud system is $i$. The steady-state probability distribution $\Pi$ of the Markov chain is written as a partitioned vector. The partitioned vector can be given by

$$\Pi = (\pi_0, \pi_1, \pi_2, \ldots). \quad (4)$$

### 3. Analysis of system model

We analyze the system model in steady state with transition rate matrix of the three-dimensional Markov chain.

#### 3.1. Construction of the transition rate matrix

Denoting $Q$ as the one-step state transition rate matrix of the Markov chain ${N(t), Y(t), S(t), t \geq 0}$, and denoting $Q_{x,y}$ as the one-step state transition rate sub-matrix for the system level transferring from $x$ ($x = 0, 1, 2, \ldots$) to $y$ ($y = 0, 1, 2, \ldots$), we discuss $Q_{x,y}$ in the following cases.

1. **System level decreases:**
   - $1 \leq x \leq c$ indicates that the number of visitors does not exceed the total number of VMs in the cloud system.

   When the number of visitors equals the number of VMs operating normally, if the system level decreases by one, the number of VMs operating normally also decreases. When a visitor leaves the cloud system after receiving the experience service, both the system level and the system stage decrease by one, whereas the system phase remains unchanged. So the transition rate can be calculated as $(j - k)\bar{q}\mu_1$, where $j$ is the total number of active VMs and $k$ is the number of visitors experiencing the enrollment service. When a visitor finishes the enrollment service and leaves the cloud system, all of the system level, the system stage, and the system phase decrease by one, and the transition rate can be calculated as $k\mu_2$.

   When there are more visitors than the VMs working normally, if the system level decreases by one, the number of VMs operating normally remains fixed. When a visitor leaves the cloud system after receiving the experience service, the system level decreases by one, whereas the system stage and the system phase remain unchanged. Therefore, the transition rate can be calculated as $(j - k)\bar{q}\mu_1$, where $j$ is the number of active VMs in the cloud system and $k$ is the number of visitors experiencing the enrollment service. When a visitor leaves the system after receiving the experience service, both the system level and the system phase decrease by one, while the
system stage remains unchanged, and the transition rate can be calculated as \( k \mu_2 \). Thus, the sub-matrix \( Q_{x,x-1} \) with an order of \( \left\lfloor \frac{1}{2} (x(x+1)) \right\rfloor \) is given as follows:

\[
Q_{x,x-1} = \begin{pmatrix}
0 & 0 & \bar{q} \mu_1 \\
0 & \mu_2 & 0 \\
0 & 2\bar{q} \mu_1 & \mu_2 \bar{q} \mu_1 \\
& & 2\mu_2 \end{pmatrix}.
\]

\( x > c \) indicates that there are more visitors than VMs in the cloud system. Therefore, if the system level decreases by one, the number of VMs working normally remains fixed.

If a visitor leaves the system after receiving the experience service, the system level decreases by one, but both the system stage and the system phase remain unchanged, and the transition rate can be calculated as \((j - k)\bar{q} \mu_1\), where \( j \) is the number of active VMs in the cloud system and \( k \) is the number of visitors experiencing the enrollment service. If leaves the system after receiving the enrollment service, both the system level and the system phase decrease by one, whereas the system stage remains unchanged. The transition rate can then be calculated as \( k \mu_2 \).

Thus, the sub-matrix \( Q_{x,x-1} \) with an order of \( \left\lfloor \frac{1}{2} (c+1)(c+2) \right\rfloor \) is given as follows:

\[
Q_{x,x-1} = \begin{pmatrix}
0 & 0 \\
\bar{q} \mu_1 & \mu_2 \\
& & 0 \\
0 & 2\bar{q} \mu_1 & \mu_2 \bar{q} \mu_1 \\
& & 2\mu_2 \end{pmatrix}.
\]
(2) System level remains unchanged:

\( x = 0 \) indicates that there are no visitors in the cloud system and none of the VMs are active. Provided that no visitors arrive within the current sleep period, the system level, the system stage and the system phase will all remain unchanged, so the transition rate can be calculated as \(-\lambda\). Thus, the sub-matrix \( Q_{0,0} \) is given by

\[
Q_{0,0} = -\lambda.
\]

\( x \geq 1 \) indicates that at least one visitor exits the cloud system. In the case where the number of VMs operating normally is less than \( \min(c, x) \), there is at least one VM asleep. Given that no visitors arrive within the current sleep period and no visitors select or finish the enrollment service, the system level, the system stage and the system phase will all remain unchanged, so the transition rate can be calculated as

\[
-\left(\lambda + (c - j)\theta + (j - k)q\mu_1 + k\mu_2\right),
\]

where \( j \) is the number of active VMs and \( k \) is the number of visitors being enrolled as members. In the case where the number of VMs operating normally is equal to \( \min(c, x) \), if there are more visitors than VMs, i.e., \( \min(c, x) = c \), all the VMs are working normally without sleep. If the number of VMs is greater than the number of visitors, i.e., \( \min(c, x) = x \), the system buffer will be empty. Hence, the sleeping VMs will enter another sleep period when current sleep period ends. If no visitors arrive and no visitors select or finish the enrollment service, then the system level, the system stage and the system phase will all remain unchanged, so the transition rate can be calculated as

\[
-\left(\lambda + (c - j)\theta + (j - k)q\mu_1 + k\mu_2\right),
\]

(3) System level increases:
With the arrival of one visitor, the system level will increase, whereas the system stage and the system phase will be fixed, so the transition rate can be calculated as $\lambda$.

Thus, the sub-matrix $Q_{x,x+1}$ with an order of $\left\lfloor \frac{1}{2}(\min(x, c) + 1)(\min(x, c) + 2) \right\rfloor \times \left\lfloor \frac{1}{2}(\min(x, c) + 1)(\min(x, c) + 2) \right\rfloor$ is given as follows:

$$Q_{x,x+1} = \begin{pmatrix} \lambda & 0 & \cdots & 0 \\ \lambda & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda \end{pmatrix}.$$  

Based on the discussions above, we conclude that the sub-matrices $Q_{x,x}$ and $Q_{x,x+1}$ repeat the starting system level $c$, and the sub-matrices $Q_{x,x-1}$ repeat the starting system level $(c + 1)$. Therefore, we write the transition rate matrix $Q$ as

$$Q = \begin{pmatrix} Q_{0,0} & Q_{0,1} & \cdots & 0 \\ Q_{1,0} & Q_{1,1} & \cdots & Q_{1,2} \\ \vdots & \ddots & \ddots & \vdots \\ Q_{c-1,c-2} & Q_{c-1,c-1} & \cdots & Q_{c-1,c} \\ 0 & \cdots & 0 & Q_{c,c} \end{pmatrix}.$$  

From the structure of the transition rate matrix $Q$, we find that the state transitions of the Markov chain under consideration occur only between adjacent levels. Referencing [16], the Markov chain $\{ N(t), Y(t), S(t), t \geq 0 \}$ is in fact a Quasi Birth-and-Death (QBD) process. Therefore, we resort to the method of matrix-geometric solution to analyze the system model.

3.2. Derivation of the probability distribution. By solving the matrix quadratic equation $R^2 Q_{x+1,c} + R Q_{x,c} + Q_{x,c+1} = 0$, we analyze the QBD process. In order to guarantee $R^2 Q_{x+1,c} + R Q_{x,c} + Q_{x,c+1} = 0$ has a minimal non-negative solution $R$ and a spectral radius $SP(R) < 1$, we apply the consistency technique formula to tackle the one-step transition rate matrix $Q$ as

$$Q' = Q/U,$$  

where $U$ is the absolute value of the minimum element in sub-matrix $Q_{x,c}$. The matrix quadratic equation can be modified as $R^2 Q'_{x+1,c} + R Q'_{x,c} + Q'_{x,c+1} = 0$, where $Q'_{x+1,c} = Q_{x+1,c}/U$, $Q'_{x,c} = Q_{x,c}/U$ and $Q'_{x,c+1} = Q_{x,c+1}/U$. To give the rate matrix $R$ numerically, we propose an iterative algorithm shown in Table 1.

With the rate matrix $R$, we establish a square matrix $B[R]$ as

$$B[R] = \begin{pmatrix} Q_{0,0} & Q_{0,1} & \cdots & 0 \\ Q_{1,0} & Q_{1,1} & \cdots & Q_{1,2} \\ \vdots & \ddots & \ddots & \vdots \\ Q_{c-1,c-2} & Q_{c-1,c-1} & \cdots & Q_{c-1,c} \\ 0 & \cdots & 0 & Q_{c,c} \\ 0 & \cdots & 0 & R Q_{c+1,c} + Q_{c,c} \end{pmatrix}.$$  

(6)
Table 1. Iterative algorithm to calculate the rate matrix $R$.

**Step 1:** Setting the error precision $\varepsilon$ (for example, $\varepsilon = 10^{-8}$). Initialize $c$, $\lambda$, $\mu_1$, $\mu_2$, $\theta$ and $q$ as needed. Initialize the rate matrix $R = 0$ with an order of $m \times m$, where $m = \left(\frac{1}{2}(c + 1)(c + 2)\right)$.

**Step 2:** Tackle $Q$ by using the consistency technique formula and get $Q'_{c+1}^{c+1}$, $Q'_{c}$ and $Q'_{c+1}$.

$Q' = Q/U$, $Q'_{c+1} = Q_{c+1}/U$, $Q'_{c} = Q_{c}/U$, $Q'_{c+1} = Q_{c+1}/U$.

**Step 3:** Calculate $R^*$ by

$R^* = R^2 \times Q'_{c+1} + R \times (I + Q'_{c}) + Q'_{c+1}$. $I$ is an identity matrix.

**Step 4:** While $\|R - R^*\|_{\infty} > \varepsilon$

$\|R - R^*\|_{\infty} = \max \{ \sum_{i=1}^{m} \sum_{j=1}^{m} |r_{i,j} - r'_{i,j}| \}$, where $r_{i,j}$ and $r'_{i,j}$ are elements in $R$ and $R^*$ respectively.

$R = R^*$, $R^* = R^2 \times Q'_{c+1} + R \times (I + Q'_{c}) + Q'_{c+1}$.

Endwhile

**Step 5:** $R = R^*$,

**Step 6:** Output $R$.

And then, we give the following equations:

\[
\begin{cases}
(\pi_0, \pi_1, \ldots, \pi_c) B[R] = 0 \\
(\pi_0, \pi_1, \ldots, \pi_{c-1}) e + \pi_c (I - R)^{-1} e_1 = 1
\end{cases}
\]  

where $e$ is a ones vector with an order of $\left(\frac{1}{2}(c + 1)(c + 2)\right) \times 1$, and $e_1$ is a ones vector with an order of $\left(\frac{1}{2}(c + 1)(c + 2)\right) \times 1$.

Using the Gauss-Seidel method [13] to solve Eq. (7), we get $\pi_0, \pi_1, \pi_2, \ldots, \pi_c$. Following the matrix-geometric form, we obtain $\pi_i$ ($i = c + 1, c + 2, c + 3, \ldots$) by

$\pi_i = \pi_c R^{i-c}, \ i \geq c$.  

Substituting the solution of $\pi_c$ obtained from Eq. (7) into Eq. (8), we further obtain the solution of $\pi_i$ ($i = c + 1, c + 2, c + 3, \ldots$).

With the mathematical analysis above, the probability distribution $\Pi = (\pi_0, \pi_1, \pi_2, \ldots)$ of the system model can be given numerically.

4. **Performance criteria.** We derive the mean delay and the energy conservation level for numerically evaluating the system performance of the proposed cloud architecture.

The delay of a visitor is defined as the time period from the moment a visitor arrives at the system to the moment the visitor leaves the system. That will be the sum of the time taken by a visitor waiting in the system buffer plus the time spent receiving service.
Based on the probability distribution obtained in Section 3, we give the average number $E[L]$ of visitors in the cloud system as follows:

$$E[L] = \sum_{i=c}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} (i - c) \pi_{ijk}.$$  

(9)

According to Little’s formula, we give the mean delay $E[W]$ of visitors by

$$E[W] = \frac{1}{\lambda} E[L] = \left( \sum_{i=c}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} \frac{(i - c) \pi_{ijk}}{\lambda} \right).$$  

(10)

If a visitor does not select the enrollment service, the average service time is $\frac{1}{\mu_1}$. The mean delay $E[T_1]$ of the visitors who do not select the enrollment service is

$$E[T_1] = \left( \sum_{i=c}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} \frac{(i - c) \pi_{ijk}}{\lambda} \right) + \frac{1}{\mu_1}.$$  

(11)

If a visitor selects the enrollment service after getting the experience service, the average service time is $\frac{1}{\mu_1} + \frac{1}{\mu_2}$. The mean delay $E[T_2]$ of the visitors who select the enrollment service is then given by

$$E[T_2] = \left( \sum_{i=c}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} \frac{(i - c) \pi_{ijk}}{\lambda} \right) + \frac{1}{\mu_1} + \frac{1}{\mu_2}.$$  

(12)

The energy conservation level is defined as the energy savings per unit time due to the introduction of the sleeping mechanism. When a VM is asleep, power is conserved, so energy consumption can be reduced. However, when a VM wakes up, there will be extra energy consumption.

We let $V_{a1}$ denote the necessary power required for a VM to provide the experience service, $V_{a2}$ denote the necessary power required for a VM to provide the enrollment service, $V_e$ denote the necessary power required for a sleeping VM, and $V_t$ denote the energy consumed when a VM is woken up. The energy conservation level $E[S]$ of the system is given by

$$E[S] = (V_a - V_s) \sum_{i=0}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} (c - j) \pi_{ijk} - V_t \sum_{i=1}^{\infty} \sum_{j=0}^{c} \sum_{k=0}^{j} (c - j) \pi_{ijk} \times \theta.$$  

(13)

where $V_a = (1 - q)V_{a1} + q(V_{a1} + V_{a2})$.

5. **System experiments.** We carry out system experiments to evaluate the system performance of the proposed cloud architecture. The analysis results are presented in Matlab 2010a on Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz 3.60 GHz, 6.00 GB RAM. The simulation results are obtained under the platform of MyEclipse 2014. We create a VISITOR class with properties of NOTARR, WAITING, RUNING, DEPART and KIND to keep track of the visitor state. Moreover, we create a VM class with properties of ASLEEP and ACTIVE to keep track of the server state.

Using parameters of $c = 10$, $\mu_1 = 0.5 \text{ s}^{-1}$ and $\mu_2 = 0.2 \text{ s}^{-1}$, we demonstrate the dependent relationship between the mean delay $E[T_2]$ of visitors selecting the enrollment service and the arrival rate $\lambda$ of visitors with different enrollment probabilities $q$ and different sleeping parameters $\theta$ in Fig. 2.
As can be seen from Fig. 2, for all the enrollment probabilities \( q \) and all the sleeping parameters \( \theta \), the mean delay \( E[T_2] \) increases as the arrival rate \( \lambda \) of visitors increases. As the arrival rate increases, there will be more visitors queueing in the system buffer. Also, a newly arriving visitor, regardless of whether they select the enrollment service or not, will wait longer before accessing a VM.

For a lower arrival rate (such as \( \lambda < 1.2 \)) and the same sleeping parameter, the enrollment probability will have little effect on the average latency \( E[T_2] \), since a VM is more likely to be asleep with a lower arrival rate. The sleeping parameter will therefore be the main factor affecting the waiting time of visitors in the queue.

For a higher arrival rate (such as \( \lambda > 1.2 \)) and the same sleeping parameter, an increase in the enrollment probability results in a greater mean delay \( E[T_2] \). With a higher arrival rate, a newly arriving visitor is more likely to queue longer in the system buffer before accessing a VM. The larger the enrollment probability is, the more visitors there will be who select the enrollment service. So, the visitors will be delayed longer in the system buffer, and the mean delay of the visitors who select the enrollment service will increase.

For a lower arrival rate (such as \( \lambda < 1.8 \)) and the same enrollment probability, the mean delay \( E[T_2] \) decreases as the sleeping parameter increases. With a lower arrival rate, it is less likely that all of VMs will be active. With a larger sleeping parameter, a sleep period will be shorter, i.e., the visitors who arrive within the sleep period are more likely to get service earlier. Thus, the mean delay of the visitors who select the enrollment service will decrease.

For a higher arrival rate (such as \( \lambda > 1.8 \)) and the same enrollment probability, it is less likely that all the VMs will be asleep, so the sleeping parameter will have little effect on the mean delay \( E[T_2] \).

Using the parameters of \( V_{a1} = 5 \text{ mW}, V_{a2} = 4 \text{ mW}, V_s = 0.5 \text{ mW} \) and \( V_t = 6 \text{ mW} \), we demonstrate the dependent relationship between the energy conservation level \( E[S] \) of the system and the arrival rate \( \lambda \) of visitors with different enrollment probabilities \( q \) and different sleeping parameters \( \theta \) in Fig. 3.
As can be seen from Fig. 3, for all the enrollment probabilities $q$ and all the sleeping parameters $\theta$, the energy conservation level $E[S]$ shows a downtrend decreasing to 0 as the arrival rate $\lambda$ increases. With an increase in the arrival rate, the VMs are more likely to be active. Since no energy is saved during the active state, the energy conservation level of the system shows a downtrend. When the arrival rate reaches a certain value (such as $\lambda \geq 2.5$ for $\theta = 0.8$ and $q = 0.4$), all the VMs will be more likely to remain awake. Thus, the energy conservation level tends to 0.

For the same arrival rate and the same sleeping parameter, an increase in the enrollment probability leads to a smaller energy conservation level. With a greater enrollment probability, a visitor is more likely to select the enrollment service, which forces the VMs to operate for a longer time, i.e., sleep for a shorter time. So the energy conservation level of the system shows a downtrend.

For the same arrival rate and the same enrollment probability, an increase in the sleeping parameter leads to a smaller energy conservation level. With a greater sleeping parameter, a sleep period will be shorter. If a visitor arrives before the sleep timer expires, the VMs will be more likely to enter an active state earlier. So the energy conservation level of the system shows a downtrend.

A higher arrival rate means a longer mean delay and a smaller energy conservation profit, but a greater profit for the application provider. A lower arrival rate means a shorter mean delay and a greater energy saving profit, but a smaller profit for the application provider. Regulating the arrival behavior of visitors is therefore the key factor in optimizing our proposed cloud architecture.

6. Enrollment fee. In this section, we discuss the Nash equilibrium and the socially optimal strategies for visitors. Moreover, we formulate a pricing policy and impose an appropriate enrollment fee on the visitors who upgrade to member status.

6.1. Nash equilibrium and socially optimal strategies. With our proposed cloud architecture, each visitor independently decides whether to join the cloud
system or not. Visitors are inherently selfish. However, a compromise needs to be reached between all stakeholders in a cloud system. For this, we will investigate both the Nash equilibrium and the socially optimal strategies of visitors. To this end, we make following assumptions.

(1) The reward for a visitor after receiving experience service is $R_1$, the reward of a visitor after upgrading to a VIP user is $R_2$, the profit of the cloud provider due to any energy saving is $\psi$ for each milliwatt. A visitor’s cost from remaining in the system is $\varepsilon$ per unit of time.

(2) The welfare of all the visitors is alike, and we can add these welfares.

(3) In order to ensure the visitors selecting the enrollment service to gain some profit, under the condition that the system buffer is empty, the reward a newly arriving visitor who completes their experience service and enrollment service should be greater than the cost for a delay in the system, i.e., $R_1 > \varepsilon(1/\mu_1 + 1/\theta)$ and $R_2 > \varepsilon/\mu_2$. Otherwise, all the visitors, even when the system buffer is empty, would not join the system.

(4) In order to make sure the system is stable, the maximum arrival rate $\lambda_{max}$ of visitors should be set.

We define the individual welfare $G_{ind}(\lambda)$ of a visitor as follows:

$$G_{ind}(\lambda) = R_1 + qR_2 - \varepsilon(qE[T_1] + (1 - q)E[T_2])$$

where $E[T_1]$ is given in Eq. (11) and $E[T_2]$ is given in Eq. (12).

An arriving visitor will decide to either join or balk the system. If $G_{ind}(0) \geq 0$, even if every visitor joins the system, the welfare for each visitor can not be negative. Thus, the equilibrium strategy is to join the system, i.e., $\lambda_e = \lambda_{max}$. If $G_{ind}(\lambda_{max}) \leq 0$, even if no other visitors join the system, the welfare of a visitor who joins is non-positive. Therefore, balking the system is an equilibrium strategy, i.e., $\lambda_e = 0$. If $G_{ind}(0) < G_{ind}(\lambda) < G_{ind}(\lambda_{max})$. There is a unique equilibrium strategy, where the Nash equilibrium arrival rate $\lambda_e$ of visitors is subject to $G_{ind}(\lambda_e) = 0$.

Given that no pricing policy is imposed on visitors, by adding the welfare of the system to the energy conservation and the welfare for all the visitors, the social welfare $G_{soc}(\lambda)$ can be obtained as follows:

$$G_{soc}(\lambda) = \lambda(R_1 + qR_2 - \varepsilon(qE[T_1] + (1 - q)E[T_2])) + \psi E[S]$$

The socially optimal arrival rate $\lambda^*$ of visitors is then given as follows:

$$\lambda^* = \text{argmax}_{0 \leq \lambda \leq \lambda_{max}} \{G_{soc}(\lambda)\}$$

where “arg max” represents the argument of the maximum. In other words, the set of points from “arg max” ensures the social welfare $G_{soc}(\lambda)$ attains its maximum value.

With system experiments, we illustrate the trends of $G_{ind}(\lambda)$ and $G_{soc}(\lambda)$ for different enrollment probabilities $q$ and sleeping parameters $\theta$. Besides the parameters given in Section 5, we set $R_1 = 50$, $R_2 = 80$, $\varepsilon = 10$ and $\psi = 0.1$ as examples in the system experiments.

Figure 4 illustrates how the individual welfare $G_{ind}(\lambda)$ changes with the arrival rate $\lambda$ of visitors.

As can be seen from Fig. 4, when the enrollment probabilities and the sleeping parameters are given, when the arrival rate of visitors increases, the individual welfare $G_{ind}(\lambda)$ continually decreases and tends to a negative value. A higher arrival rate leads to a greater sojourn time, hence the individual welfare is lower.
When the arrival rate reaches a certain value (such as $\lambda \geq 2.3$ for $\theta = 0.8$ and $q = 0.4$), the system tends towards becoming unstable, being delayed longer is inevitable, this results in a negative welfare of visitors. Looking at each curve of the individual welfare, we find that there is a unique arrival rate subject to $G_{\text{ind}}(\lambda) = 0$, where the arrival rate is the Nash equilibrium arrival rate $\lambda_e$.

We demonstrate the dependent relationship between the social welfare $G_{\text{soc}}(\lambda)$ and the arrival rate $\lambda$ of visitors in Fig. 5.

In Fig. 5, we observe that as the arrival rate of visitors increases, the social welfare firstly increases, and then decreases. For a lower arrival rate, any increase
in the arrival rate has little effect on the mean delay. The main factor affecting
the social welfare is reward rather than cost. As the arrival rate increases, more
visitors will earn rewards, resulting in a greater social welfare. When the arrival
rate is higher, the main factor affecting the social welfare is cost rather than reward.
With an increase in the arrival rate, the visitors will be delayed longer, resulting
in a smaller social welfare. In summary, the social welfare shows a concave trend.
Therefore, at the peak value of the social welfare we obtain the socially optimal
arrival rate $\lambda^\ast$.

Obviously, when the enrollment probabilities and the sleeping parameters are
given, the arrival rate $\lambda^\ast$ of visitors under the socially optimal strategy is always
lower than the arrival rate $\lambda_e$ of visitors under the Nash equilibrium strategy. In
other words, less visitors would receive the experience service and upgrade as mem-
bers under the socially optimal strategy. In this event, we formulate a pricing policy
to impose an appropriate enrollment fee on the visitors who select the enrollment
service.

6.2. Pricing policy. In order to prevent too many visitors accessing the cloud
system and to control the scale of membership, we charge an appropriate enrollment
fee $f$ to the visitors who receive the experience service and select the enrollment
service. Given an enrollment fee $f$, we modify the individual welfare $G_{\text{ind}}'(\lambda)$ as

$$G_{\text{ind}}'(\lambda) = R_1 + q(R_2 - f) - \varepsilon(qE[T_1] + (1 - q)E[T_2]).$$

(16)

A cloud system is composed of all visitors and the application provider. The
enrollment fee is transferred from the visitors to the application provider. Therefore,
the enrollment fee has no effect on the social welfare.

However, it is difficult to give an exact expression and to explain the monotonic-
ity for the social welfare $G_{\text{soc}}'(\lambda)$. For this, we resort to an intelligent searching
algorithm, called a “Bat algorithm”, based on the echolocation behavior of bats to
derive the socially optimal arrival rate $\lambda^\ast$.

To avoid any local maximum problems and to effectively accelerate the conver-
gence seep, we improve the Bat algorithm by gradually decreasing the step span.

The main steps of the improved Bat algorithm proposed in this paper are given
in Table 2.

Adopting the parameters in Subsection 6.1 to the improved Bat algorithm, we
carry out numerical experiments on the socially optimal arrival rate $\lambda^\ast$.

Putting the socially optimal arrive rate $\lambda^\ast$ obtained above into Eq. (16) and
setting $G_{\text{ind}}'(\lambda) = 0$, we get the enrollment fee $f$ by

$$f = 1/q \times (R_1 + qR_2 - \varepsilon(qE[T_1] + (1 - q)E[T_2])) \bigg|_{\lambda = \lambda^\ast}. $$

(17)

For different enrollment probabilities $q$ and sleeping parameters $\theta$, we present
numerical results for the enrollment fee $f$ in Table 3.

From Table 3, it is observed that when the sleeping parameter $\theta$ is given, the
enrollment fee $f$ will decrease as the enrollment probability $q$ increases. The increase
in the enrollment probability leads to a longer mean delay and a greater sojourn
cost of visitors. Under this situation, the users’ motivation to access the cloud
system is lower. To motivate more visitors to receive the experience service and
upgrade as members, the enrollment fee $f$ should be set lower. It is also observed
that when the enrollment probability $q$ is given, the enrollment fee $f$ will increase as
the sleeping parameter $\theta$ increases. An increase in the sleeping parameter leads to
a smaller mean delay and a lower sojourn cost of visitors. In this case, more visitors
Table 2. Improved Bat algorithm to obtain $\lambda^*$ and $G_{soc}(\lambda^*)$.

**Step 1:** Set the number $N$ of bats, loudness $A_0$, pulse rate $R_0$, the maximum search frequency $f_{\text{max}}$, the minimum search frequency $f_{\text{min}}$, upper search bound $U_b$, lower search bound $L_b$, the minimum moving step $\text{step}_{\text{min}}$, volume attenuation coefficient $\eta$, searching frequency enhancement factor $\phi$. Set the initial number of iterations as $\text{iter} = 1$, the maximum iterations as $\text{iter}_{\text{max}}$.

**Step 2:** Initialize the position, the loudness and the pulse rate for each bat.

For $i = 1 : N$

- $\lambda_i = L_b + (U_b - L_b) \times \text{rand}$
  
- $A_i = A_0$
  
- $r_i = R_0$

Endfor

**Step 3:** Calculate the fitness for each bat.

$G_{soc}(\lambda_i) = \lambda_i (R_1 + qR_2 - \varepsilon (qT_1 + (1 - q)T_2)) + \psi E[S]$, $i \in \{1, 2, \ldots, N\}$.

$\lambda^* = \arg\max_{i \in \{1, 2, \ldots, N\}} \{G_{soc}(\lambda_i)\}$ $\% \lambda^*$ is present optimal position.

**Step 4:** Calculate the position and the fitness for each bat.

For $i = 1 : N$

- $f_i = f_{\text{min}} + (f_{\text{max}} - f_{\text{min}}) \times \text{rand}$
- $v_i = v_i + (\lambda_i - \lambda^*) f_i$
- $\lambda_i = \lambda_i + v_i$
  
- If $r_i < \text{rand}$
  
- $\lambda_i = \lambda^* + (1/(2 \times \text{iter}) + \text{step}_{\text{min}}) \times \text{randn}$
  
- $\% \text{randn}$ returns a sample from the “standard normal” distribution.

Endif

$G_{soc}(\lambda_i) = \lambda_i (R_1 + qR_2 - \varepsilon (qT_1 + (1 - q)T_2)) + \psi E[S]$

If $G_{soc}(\lambda_i) > G_{soc}(\lambda_i)$ and $(A_i > \text{rand})$

$G_{soc}(\lambda_i) = G_{soc}(\lambda_i)$

$A_i = \eta A_i$

$r_i = R_0 (1 - \exp(-\phi \times \text{iter}))$

Endif

Endfor

**Step 5:** Select the optimal position among all the bats.

$\lambda^* = \arg\max_{i \in \{1, 2, \ldots, N\}} \{G_{soc}(\lambda_i)\}$.

**Step 6:** Check iterations.

If $\text{iter} < \text{iter}_{\text{max}}$

- $\text{iter} = \text{iter} + 1$, go to Step 4

Endif

**Step 7:** Output the optimal position $\lambda^*$ and the maximum fitness $G_{soc}(\lambda^*)$.

may very well access the cloud system. In order to limit the risk of performance deterioration due to overcrowding, a higher enrollment fee is necessary.

Compared with no sleep state for the VMs, for the same enrollment probability, the enrollment fee is lower. When there is a sleep state for the VMs, the users’ QoS with the mean delay of visitors is lower, and then the sojourn cost becomes bigger. So the enrollment fee imposed on the visitors who upgrade as members should be set lower.
Table 3. Numerical results for the enrollment fee.

| Sleeping parameter (θ) | Enrollment probability (q) | Socially optimal arrival rate (λ*) | Maximum social welfare (G_{soc}(λ*)) | Enrollment fee (f) |
|------------------------|---------------------------|-----------------------------------|-------------------------------------|--------------------|
| no sleep               | 0.3                       | 2.1256                            | 73.0759                             | 114.5963           |
| no sleep               | 0.4                       | 1.8489                            | 68.6578                             | 92.8360            |
| no sleep               | 0.5                       | 1.6465                            | 65.3641                             | 79.3976            |
| 0.8                    | 0.3                       | 2.0560                            | 65.4861                             | 105.0306           |
| 0.8                    | 0.4                       | 1.7981                            | 61.9026                             | 85.2437            |
| 0.8                    | 0.5                       | 1.6020                            | 59.2212                             | 73.3388            |
| 0.2                    | 0.3                       | 1.8647                            | 49.7374                             | 78.4550            |
| 0.2                    | 0.4                       | 1.6420                            | 47.9301                             | 69.9957            |
| 0.2                    | 0.5                       | 1.4728                            | 46.6180                             | 60.8772            |

7. Conclusions. In this paper, with the aim of reducing the high energy consumption in a cloud system and establishing a loyal and stable client base, we proposed a sleeping mechanism-based cloud architecture, in which experience service and enrollment service are deployed on one VM. Regarding the experience service as the first essential service, the enrollment service as the selectable extra service, and the sleep period as a vacation, we established a queue with an asynchronous multi-vacation and a selectable extra service. Combining analysis and simulation results, we revealed how the system performance is influenced by the system parameters within our proposed cloud architecture. We investigated the Nash equilibrium arrival rate of visitors based on an individual welfare function. Furthermore, we presented a type of Bat algorithm with an adaptive step, and obtained the socially optimal arrival rate of visitors. Experimental results show that the arrival rate of visitors under the Nash equilibrium strategy is higher than that under the socially optimal strategy. We then presented a pricing policy with an appropriate enrollment fee that coincides with the Nash equilibrium and the socially optimal arrival rates.

Acknowledgments. This research is supported in part by grants from NSF (Grant Nos. 61872311 and 61472342), and Hebei Province NSF (Grant No. F2017203141) of China, and is supported in part by MEXT and JSPS KAKENHI (Grant Nos. JP17H01825 and JP26280113) of Japan. The authors would also like to thank the reviewers for their invaluable comments on this paper.

REFERENCES

[1] S. Ahn, J. Lee, S. Park, S. Newaz and J. Choi, Competitive partial computation offloading for maximizing energy efficiency in mobile cloud computing, IEEE Access, 6 (2018), 899–912.
[2] R. Buyya, A. Beloglazov and J. Abawajy, Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges, Eprint Arxiv, 12 (2010), 6–17.
[3] R. Dhanwate and V. Bhagat, Improving energy efficiency on android using cloud based services, International Journal of Advance Research in Computer Science and Management Studies, 3 (2015), 75–79.
[4] B. Doshi, Queueing systems with vacations—A survey, Queueing Systems, 1 (1986), 29–66.
[5] W. Gary, P. Wang and M. Scott, A vacation queueing model with service breakdowns, Applied Mathematical Modelling, 24 (2000), 391–400.
[6] M. Ghorbani-Mandolakani and M. Rad, ML and Bayes estimation in a Two-Phase tandem queue with a second optional service and random feedback, Communications in Statistics-Theory and Methods, 45 (2016), 2576–2591.
[7] Z. Gui, J. Xia, N. Zhou and Q. Huang, How to Choose Cloud Services: Toward a Cloud Computing Cost Model, CRC Press, 2013.
[8] Z. Guo, M. Song and Q. Wang, Policy-based market-oriented cloud service management architecture, Proc. of the International Conference on Information and Management Engineering, Wuhan, China, (2011), 284–291.
[9] J. Hu, J. Deng and J. Wu, A green private cloud architecture with global collaboration, Telecommunication Systems, 52 (2013), 1269–1279.
[10] S. Hussein, Y. Alkabani and H. Mohamed, Green cloud computing: Datacenters power management policies and algorithms, Proc. of the 9th IEEE International Conference on Computer Engineering and Systems, Cairo, Egypt, (2015), 421–426.
[11] A. Jain and M. Jain, Multi-server machine repair problem with unreliable server and two types of spares under asynchronous vacation policy, International Journal of Mathematics in Operational Research, 10 (2017), 286–315.
[12] S. Jin, H. Wu and W. Yue, Pricing policy for a cloud registration service with a novel cloud architecture, Cluster Computing, 22 (2019), 271–283.
[13] S. Jin, X. Ma and W. Yue, Energy-saving strategy for green cognitive radio networks with an LTE-advanced structure, Journal of Communications and Networks, 18 (2016), 610-618.
[14] Z. Ma, P. Wang and W. Yue, Performance analysis and optimization of a pseudo-fault Geo/Geo/1 repairable queueing system with N-policy, setup time and multiple working vacations, Journal of Industrial and Management Optimization, 13 (2017), 1467–1481.
[15] K. Madan, An M/G/1 queue with second optional service, Queueing Systems, 34 (2000), 37–46.
[16] M. Neuts, Matrix-Geometric Solutions in Stochastic Models, Johns Hopkins University Press, 1981.
[17] P. Shi, H. Wang, X. Yue, S. Yang, X. Fu and Y. Peng, Corporation architecture for multiple cloud service providers in jointcloud computing, Proc. of the 37th International Conference on Distributed Computing Systems Workshops, Atlanta, USA, (2017), 294–298.
[18] C. Singh, M. Jain and B. Kumar, Queueing model with state-dependent bulk arrival and second optional service, International Journal of Mathematics in Operational Research, 3 (2011), 322–340.
[19] A. Tarabia, Transient and steady state analysis of an M/M/1 queue with balking, catastrophes, server failures and repairs, Journal of Communications and Networks, 7 (2017), 811–823.
[20] C. Wei, L. Cai and J. Wang, A discrete-time Geom/G/1 retrial queue with balking customers and second optional service, Opsearch, 53 (2016), 344–357.
[21] H. Wu, S. Jin, W. Yue and Y. Takahashi, Performance evaluation for a registration service with an energy efficient cloud architecture, Proc. of the International Conference on Queueing Theory and Network Applications, Tsukuba City, Japan, (2018), 133–141.
[22] K. Ye, D. Huang, X. Jiang, H. Chen and S. Wu, Virtual machine based energy-efficient data center architecture for cloud computing: A performance perspective, Proc. of the IEEE/ACM International Conference on Green Computing and Communications, Hangzhou, China, (2010), 171–178.

Received October 2018; revised January 2019.

E-mail address: jsf@ysu.edu.cn
E-mail address: wuhaixing1120163.com
E-mail address: yue@konan-u.ac.jp
E-mail address: takahashi@i.kyoto-u.ac.jp