PERFORMANCE ANALYSIS AND SYSTEM OPTIMIZATION OF AN ENERGY-SAVING MECHANISM IN CLOUD COMPUTING WITH CORRELATED TRAFFIC

XUENA YAN
School of Information Science and Engineering
Yanshan University, Qinhuangdao 066004, China
Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province
Yanshan University, Qinhuangdao 066004, China
Langfang Yanjing Vocational Technical College, Langfang 065200, China

SHUNFU JIN*
School of Information Science and Engineering
Yanshan University, Qinhuangdao 066004, China
Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province
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 606-8501, Japan

(Communicated by Shoji Kasahara)

Abstract. Energy consumption is becoming a significant part of overall operational cost in cloud data centers. For the purpose of satisfying the Service Level Agreement (SLA) of cloud users while enhancing the energy efficiency in cloud computing systems, in this paper we propose an energy-saving mechanism with a sleep mode. Taking into consideration the traffic’s correlation and the stochastical behavior of data arrival requests in a random cloud environment with the proposed energy-saving mechanism, we model the system as a MAP/M/N/N+K queue with a synchronous multi-vacation. Then, we present a theoretical basis for analyzing and evaluating the system performance by taking a state transition rate matrix in the steady state. Next, we investigate the change trends for the energy saving rate of the system and the average latency of tasks by carrying out numerical experiments. Moreover, we give a

2020 Mathematics Subject Classification. Primary: 68M20, 60K25; Secondary: 90B22.
Key words and phrases. Cloud computing environment, energy-saving mechanism, service level agreement, Markovian arrival process, synchronous multi-vacation, intelligent optimization.

* Corresponding author: Shunfu Jin.
cost function to balance the system performance measures and optimize the performance of the system with the proposed energy-saving mechanism through intelligently utilizing the Chebyshev chaotic map and enriching the biological behavior of fireflies in an improved Firefly Algorithm (FA). Finally, we obtain the optimal sleep parameter and the minimal system cost using the improved FA.

1. **Introduction.** Cloud computing provides powerful and flexible computing capabilities, and will become the mainstream IT application mode for government, finance and important industry sectors in the future [17], [19]. In 2017, the size of the global Cloud Accounting Software market was 2.6 billion US dollars and it is expected to reach 4.25 billion US dollars by the end of 2025, with a Compound Annual Growth Rate (CAGR) of 6.2% during 2018-2025 [23]. As the construction size and power consumption of cloud data centers worldwide dramatically increases, a considerable amount of greenhouse gases will be emitted into the environment [1], [14]. Not surprisingly, the energy efficiency of these cloud data centers has been receiving significant attention recently [22].

How to improve the energy efficiency and construct green cloud computing has become an extremely challenging area of research [10]. An energy model was proposed to predict the energy consumption of a single machine in a cloud data center [18]. A comprehensive review of the state-of-the-art nature-inspired algorithms was presented to detail and discuss the energy issues in cloud data centers [27]. The nature-inspired algorithms were compared to explore better energy-efficient methods for operating cloud data centers. In [26], an Euclidean distance-based, multi-objective resources allocation was proposed in the form of Virtual Machines (VMs). The VM migration policy was designed to reduce energy consumption, minimize the wastage of resources, and avoid any Service Level Agreement (SLA) violation in the cloud data center. Capacity planning using a dynamic pricing algorithm was proposed in [36]. The dynamic pricing algorithm considered several factors that could reduce energy consumption in green data centers. All the strategies mentioned above were proposed to enhance energy efficiency in cloud computing.

Queueing theory provides a reliable and effective method for evaluating the energy-efficient strategies in cloud computing systems. In [2], a single-server vacation queue with exhaustive service was applied to model the task schedule in a heterogeneous cloud computing system, and a task scheduling algorithm based on traffic with similar service time was proposed to reduce the energy consumption. Different from single-server queues, some multi-server queues were applied to model a cloud computing system. In [4], for an energy-efficient cloud computing system, a Q-learning based task scheduling framework with two phases was proposed. In the first phase, an M/M/c queue is implemented by a centralized task dispatcher for assigning the arriving user requests to each server. In the second phase, all the requests are firstly prioritized by a Q-learning based scheduler on each server according to task laxity and task lifetime, then a continuously updating policy is used to assign tasks to VMs. In [30], an energy-saving VM allocation scheme with synchronous multi-sleep and sleep-delay was proposed to address the trade-off between providing higher quality of experience to users and reducing energy consumption. A multi-server queuing model was established to quantify the effects of changes to different sets of parameters, such as the sleep parameter and the sleep-delay parameter. Moreover, for resolving the deviation between the Nash equilibrium behavior and the socially optimal behavior, a pricing policy with an appropriate admission fee on tasks was presented to maximize the social profit.
In [12], a cloud architecture with a free service and a registration service was proposed. The cloud architecture was modeled and evaluated by an M/M/c queue with a second optional service. In [15], an M/G/c queue was constructed to analyze cloud computing centers, where the mean response time of a task for different offered loads was investigated using analytical and simulation results. In [37], a task scheduling strategy with a sleep-delay timer and a waking-up threshold was proposed in a cloud computing system, and an M/M/c synchronous vacation queue with a vacation-delay and an N-policy was constructed accordingly. In [28], aiming to determine and measure the Quality of Service (QoS) for cloud users, two kinds of queues were used to model a cloud computing system: A Poisson arrival process with a single server, and a Poisson arrival process with a multi-server. In this study, an open Jackson network of an M/M/1 queue and an M/M/c queue was built to provide cloud users with the best service option. In [11], considering a general arrival behavior of requests in a real cloud computing system, a GI/G/c queue was established to characterize cloud data centers, the average waiting time of requests was analyzed, and the total energy consumption was calculated. Numerical experiments show that the total energy consumption is associated with the arrival rate, the service rate and the quantity of servers in a cloud data center. In all the literature mentioned above, task arrivals were assumed to follow a Poisson process or a general process.

In cloud computing systems, some practical tasks are independent of each other, while other practical tasks are correlated [25]. The computing tasks initiated by person are usually uncorrelated. The arrival process with a Poisson distribution or a general independent distribution is appropriate for capturing the stochastical behavior of a cloud computing system with uncorrelated traffic [31]. The computing tasks, such as upload and download, initiated by a machine are correlated. A Markovian Arrival Process (MAP) is appropriate for capturing the stochastical behavior of a cloud computing system with correlated traffic [9]. For modeling correlated computing tasks in a cloud computing system, MAP is more appropriate.

There have been a lot of studies on MAP-based queues in recent years since a MAP is more universal than a Poisson distribution. In [16], a queueing model with arrivals following a MAP and having negative customers was studied. In this model, there were two classes of removal rules: the arrival of a negative customer meant that all the customers in the system were removed; the arrival of a negative customer resulted in only one customer from the head of the system being removed. In [3], a multi-server queueing model based on a MAP with constant impatient time was studied, and the loss probability, the mean waiting time, and the mean queue size were obtained.

In [38], a multi-server retrial queueing model based on a MAP with asynchronous single-vacation was constructed to calculate the loss probability, the blocking probability, the expected queue length, and the actual arrival rate of customers entering the servers. Numerical examples illustrated the relationship of the mean number of customers in the orbit and the mean number of customers under service using different fundamental arrival rates of a MAP and arrival processes. The size of the system capacity was supposed to be infinite in the research mentioned above.

Unlike those papers, [6] proposed a multi-server queueing model based on a MAP with finite capacity in a random environment to evaluate the performance of a call center. The Laplace-Stieltjes transformation for sojourn time of an arbitrary customer in the system was derived by using the method of collective marks. Although
there is a body of literature dedicated to MAP-based queues, there have been few applicable studies on MAP-based queueing models for evaluating the system performance of cloud computing. [33] evaluated an energy-saving mechanism in a random cloud environment by considering the correlation of cloud traffic. The authors assumed that the arrival of cloud data requests follows a MAP and established a MAP/M/N/N+K queue with synchronous multi-vacation to illustrate the change trends of the energy conservation level and the latency of data requests numerically.

However, we note that the lower the average latency of tasks is, the higher the QoS is. Also, the higher the energy saving rate of the system is, the lower of the total operational spending for the system is. Therefore, the system should be optimally designed to minimize the latency and maximize the energy saving rate of the system using the system parameters. Additionally, we also note that in a MAP-based queueing model, the steady-state analysis is very complicated, so it is difficult to solve the system optimization problem using conventional methods. For this, intelligent optimization algorithms have greatly contributed to solving various complex optimization problems.

Particle Swarm Optimization (PSO), which is based on the swarming behaviors of bees and other insects, has become powerful in modern numerical optimization [7], [32]. The Firefly Algorithm (FA) is a type of PSO. In [34], a standard FA was proposed for multimodal optimization. In [21], an FA was proposed for use in an autonomous mobile robot navigation in an uncertain environment. The FA has been applied to almost all areas of science and engineering, but it has not always had a significant impact on search efficiency. In [8], twelve different chaotic maps were utilized to improve the reliability of the global optimality in the FA. In [29], through adding the capability of information exchange between the top fireflies, a type of FA was proposed for accelerating the global convergence speed. Swarm intelligence optimization is basically problem-dependent, and the details of algorithm processing will vary from problem to problem.

This paper is an expansion and improvement of our previous study in [33]. In this paper, we propose an improved FA to optimize the system performance by applying the Chebyshev chaotic map and increasing several biological behaviors of fireflies. Moreover, we discuss the optimal sleep parameter and the minimal system cost based on the results of numerical experiments. The main contributions of this paper are as follows:

(i) Considering that the arrival of data requests may be correlated, and that the parameters of the arrival process and the service process may change randomly, we use a MAP to describe the correlated traffic in a practical application over a random cloud environment.

(ii) We propose a MAP-based queueing model with a multi-server and a synchronous multi-vacation to capture the energy-saving mechanism in cloud computing. In order to analyze the established MAP-based queue in a random cloud environment, we design an irreducible continuous-time four-dimensional Markov chain and construct the infinitesimal generator.

(iii) Based on the model analysis, we derive performance measures in terms of the energy saving rate of the system and the average latency of tasks for the energy-saving mechanism in a random cloud environment. Moreover, we provide numerical results to illustrate the variation trends of the system performance measures.
To satisfy the SLA of cloud users and to enhance the energy efficiency in cloud data centers, we construct a cost function to balance the system performance measures. Additionally, we improve the FA by introducing the Chebyshev chaotic map and increasing several behaviors of fireflies to optimize the energy-saving mechanism. Furthermore, we carry out experiments to obtain the optimal sleep parameter.

The remainder of this paper is organized as follows. In Section 2, for the application of an energy-saving mechanism in a random cloud environment, we construct a MAP-based queueing model with a multi-server and a synchronous multi-vacation. In Section 3, we analyze the queueing model by constructing the infinitesimal generator of a four-dimensional Markov chain. In Section 4, the performance measures for the energy-saving mechanism in cloud computing are derived. In Section 5, we carry out numerical experiments to demonstrate the effect of the system parameters on the system performance. Then, in Section 6, we establish a cost function for the system and improve the FA to optimize the energy-saving mechanism. Finally, we make our conclusions in Section 7.

2. System model. Cloud computing provides dynamic and scale virtualization resources to cloud users through a network service. A cloud environment includes several random factors. Some random factors, such as available communication bandwidth, network congestion degree, traffic categories etc., affect the arrival process of tasks, while other random factors, such as temperature, traffic load, working frequency, etc., affect the service process of tasks. Thus, a cloud environment is also called a random cloud environment.

In order to allocate cloud resources more efficiently, cloud computing deploys multiple Virtual Machines (VMs) on one Physical Machine (PM) with the help of virtualization technology and provides services to users in the form of a resource pool [13]. With the aim of improving energy conservation of PMs in cloud computing, we investigate an energy-saving mechanism with a sleep mode in a random cloud environment.

In this energy-saving mechanism, if all the VMs on a PM are idle, all the VMs hosted on a PM together with the PM itself will simultaneously switch to a sleep period controlled by a sleep timer with a random length. If no task arrives before the sleep timer expires, the PM will switch to another sleep period with a new sleep timer. Otherwise, the PM will be awakened after the sleep timer expires.

Figure 1 demonstrates the state transition of a PM with the energy-saving mechanism.

Figure 1. State transition of a PM with the energy-saving mechanism.
We regard the tasks to be submitted to the cloud data center as customers, the VMs as servers, the buffer as a waiting space, and the sleep period as a vacation. A queueing model with a multi-server and a synchronous multi-vacation is established.

Let \( N \) be the VM-number hosted on one PM and \( K \) be the buffer size of a PM. When a task arrives at a sleeping PM: (i) if the buffer is not full, the arriving task will queue in the buffer; (ii) if there are no idle VMs and the buffer is not full, the arriving task will join the system and occupy one of the idle VMs immediately; (iii) if there are no idle VMs and the buffer is full, the arriving task will queue in the buffer; (iv) if there are no idle VMs and the buffer is full, the arriving task will be balked by the system.

Taking into account all the random factors in a cloud environment, we characterize the random cloud environment in several phases. Let \( R, R \geq 1 \) be number of the phases. \( R \) can be set as needed in cloud applications. Note that when \( R = 1 \), the arrival process and service process are independent of the random cloud environment.

In this paper, we describe the random cloud environment as a stochastic process \( \{r(t), t \geq 0\} \), which is a homogeneous irreducible continuous-time Markov chain with the state space \( E_r = \{1, 2, \ldots, R\} \). Let \( h_{r_1, r_2}, r_1, r_2 \in E_r \) be transition rates for the phase of the random cloud environment changing from \( r_1 \) to \( r_2 \). The infinitesimal generator \( H \) of \( \{r(t), t \geq 0\} \) is given as follows:

\[
H = \begin{bmatrix}
h_{1,1} & h_{1,2} & \cdots & h_{1,R} \\
h_{2,1} & h_{2,2} & \cdots & h_{2,R} \\
\vdots & \vdots & \ddots & \vdots \\
h_{R,1} & h_{R,2} & \cdots & h_{R,R}
\end{bmatrix}.
\]

In environment phase \( r, r \in E_r \), the arrival of tasks is supposed to follow a Markovian Arrival Process (MAP). The arrival of tasks is directed by a stochastic process \( \{\nu(t), t \geq 0\} \) with the state space \( E_\nu = \{0, 1, \ldots, W\} \). In environment phase \( r, r \in E_r \), the stochastic process \( \{\nu(t), t \geq 0\} \) is an irreducible continuous-time Markov chain. The time duration for the Markov chain \( \{\nu(t), t \geq 0\} \) sojourn in state \( \nu, \nu \in E_\nu \) is supposed to follow an exponential distribution with a positive parameter \( \lambda^{(r)}_\nu \). At the probability \( p^{(r)}_0(\nu, \nu') \), \( \nu, \nu' \in E_\nu, \nu \neq \nu', r \in E_r \), the Markov chain \( \{\nu(t), t \geq 0\} \) jumps from state \( \nu \) to state \( \nu' \) without any task arrival, and at the probability \( p^{(r)}_1(\nu, \nu') \), \( \nu, \nu' \in E_\nu, r \in E_r \), the Markov chain \( \{\nu(t), t \geq 0\} \) jumps from state \( \nu \) to state \( \nu' \) with one task arrival. The arrival pattern of tasks is characterized by the stochastic matrices \( D^{(r)}_0 \) and \( D^{(r)}_1 \), which are given by

\[
D^{(r)}_0 = \begin{bmatrix}
-\lambda^{(r)}_0 & \lambda^{(r)}_0 p^{(r)}_0(0, 1) & \cdots & \lambda^{(r)}_0 p^{(r)}_0(0, W) \\
\lambda^{(r)}_1 p^{(r)}_0(1, 0) & -\lambda^{(r)}_1 & \cdots & \lambda^{(r)}_1 p^{(r)}_0(1, W) \\
\vdots & \vdots & \ddots & \vdots \\
\lambda^{(r)}_W p^{(r)}_0(W, 0) & \lambda^{(r)}_W p^{(r)}_0(W, 1) & \cdots & -\lambda^{(r)}_W
\end{bmatrix},
\]

\[
D^{(r)}_1 = \begin{bmatrix}
\lambda^{(r)}_0 p^{(r)}_1(0, 0) & \lambda^{(r)}_0 p^{(r)}_1(0, 1) & \cdots & \lambda^{(r)}_0 p^{(r)}_1(0, W) \\
\lambda^{(r)}_1 p^{(r)}_1(1, 0) & \lambda^{(r)}_1 p^{(r)}_1(1, 1) & \cdots & \lambda^{(r)}_1 p^{(r)}_1(1, W) \\
\vdots & \vdots & \ddots & \vdots \\
\lambda^{(r)}_W p^{(r)}_1(W, 0) & \lambda^{(r)}_W p^{(r)}_1(W, 1) & \cdots & \lambda^{(r)}_W p^{(r)}_1(W, W)
\end{bmatrix}.
\]
The matrix $D^{(r)} = D^{(r)}_0 + D^{(r)}_1$ represents the infinitesimal generator of Markov chain $\{\nu(t), t \geq 0\}$. The average arrival rate $\lambda^{(r)}$ of tasks can be calculated by

$$\lambda^{(r)} = \theta^{(r)} D^{(r)} e$$

where $\theta^{(r)}$ is the invariant vector of the Markov chain $\{\nu(t), t \geq 0\}$ with generator matrix $D^{(r)}$. The vector $\theta^{(r)}$ can be calculated by $\theta^{(r)} D^{(r)} = 0$ and $\theta^{(r)} e = 1$. Here $e$ is a column vector with $W+1$ elements and all elements of the vector are equal to 1, and 0 is a row vector with $W+1$ elements and all elements of the vector are equal to 0.

The squared coefficient of variation $c^{(r)}_{\text{var}}$ of intervals successive arrivals can be calculated as follows:

$$c^{(r)}_{\text{var}} = 2 \lambda^{(r)} \theta^{(r)} (D^{(r)}_0)^{-1} e - 1. \tag{1}$$

The coefficient of correlation $c^{(r)}_{\text{cor}}$ of two successive intervals between arrivals can be calculated as follows:

$$c^{(r)}_{\text{cor}} = \frac{1}{c^{(r)}_{\text{var}}} \left( \lambda^{(r)} \theta^{(r)} (D^{(r)}_0)^{-1} D^{(r)}_1 (D^{(r)}_0)^{-1} e - 1 \right). \tag{2}$$

In environment phase $r$, $r \in E_r$, the service time of a task is supposed to follow an exponential distribution with a positive parameter $\mu_r$. When $\mu_1 = \mu_2 = \cdots = \mu_R$, the system model is not affected by the random cloud environment.

The time duration of a sleep period is supposed to be exponentially distributed with a positive parameter $\alpha$.

The arrival epoch of a task, the service time of a task, and the time duration of a sleep period are supposed to be independent of each other.

3. Model analysis. Let $x(t), y(t) \in \{0, 1\}$ be the state of VMs at epoch $t, t \geq 0$. $x(t) = 0$ means that the VMs are asleep, and $x(t) = 1$ means that the VMs are awake. Let $y(t), y(t) \in \{0, 1, \ldots, N + K\}$ be the number of tasks in the queueing system, including the tasks being served on VMs and the tasks waiting in the buffer, at epoch $t, t \geq 0$. For convenience, we call $y(t)$ the system level. Let $h(t), h(t) \in E_r$ be the phase of the random cloud environment at epoch $t, t \geq 0$, and $z(t), z(t) \in E_r$ be the state of the MAP at epoch $t, t \geq 0$. The system model under consideration can be described in terms of the regular irreducible continuous-time four-dimensional Markov chain $\{x(t), y(t), h(t), z(t), t \geq 0\}$ with infinitesimal generator matrix $Q$. The state space $\Omega$ of the four-dimensional Markov chain $\{x(t), y(t), h(t), z(t), t \geq 0\}$ is given as

$$\Omega = \{(j, i, r, \nu) \mid j = 0, 1, i \in \{0, 1, \ldots, N + K\},$$

$$r \in E_r, \nu \in E_r \}.$$  

The stationary distribution of the four-dimensional Markov chain $\{x(t), y(t), h(t), z(t), t \geq 0\}$ is defined as follows:

$$\pi_{j,i,r,\nu} = \lim_{t \to \infty} P \{x(t) = j, y(t) = i, h(t) = r, z(t) = \nu\}, \quad (j, i, r, \nu) \in \Omega.$$  

Let $\pi_j$ be the stationary probability vector for the state of the VMs being at $j$. When $j = 0$:

$$\pi_0 = (\pi_{0,0,0}, \pi_{0,1,0}, \ldots, \pi_{0,K,0})$$
where $\pi_{0,i}$ is the stationary probability vector for the state of the VMs being at $j = 0$ and the system level being at $i, i = 0, 1, \ldots, K$. $\pi_{0,i}$ is given as

$$\pi_{0,i} = \left(\pi_{0,i,1,0}, \pi_{0,i,1,1}, \ldots, \pi_{0,i,1,W}, \pi_{0,i,2,0}, \pi_{0,i,2,1}, \ldots, \pi_{0,i,2,W}, \ldots, \pi_{0,i,R,0}, \pi_{0,i,R,1}, \ldots, \pi_{0,i,R,W}\right),$$

$i = 0, 1, \ldots, K$.

When $j = 1$:

$$\pi_{1} = \left(\pi_{1,1}, \pi_{1,2}, \ldots, \pi_{1,N+K}\right)$$

where $\pi_{1,i}$ is the stationary probability vector for the state of the VMs being at $j = 1$ and the system level being at $i, i = 1, 2, \ldots, N + K$. $\pi_{1,i}$ is given as

$$\pi_{1,i} = \left(\pi_{1,i,1,0}, \pi_{1,i,1,1}, \ldots, \pi_{1,i,1,W}, \pi_{1,i,2,0}, \pi_{1,i,2,1}, \ldots, \pi_{1,i,2,W}, \ldots, \pi_{1,i,R,0}, \pi_{1,i,R,1}, \ldots, \pi_{1,i,R,W}\right),$$

$i = 1, 2, \ldots, N + K$.

Then, the stationary probability vector $\Pi$ of the four-dimensional Markov chain $\{x(t), y(t), h(t), z(t), t \geq 0\}$ is shown as follows:

$$\Pi = (\pi_{0}, \pi_{1}).$$

The stationary distribution $\Pi$ can be obtained by solving the linear equations:

$$\Pi Q = 0$$

subject to the condition of $\Pi e = 1$. Here $e$ is a column vector with $(2K + N + 1) \times (W + 1)$ elements and all elements of the vector are equal to 1.

The infinitesimal generator matrix $Q$ can be given in a $2 \times 2$ block-structure form as follows:

$$Q = \begin{bmatrix} Q_{0,0} & Q_{0,1} \\ Q_{1,0} & Q_{1,1} \end{bmatrix}$$

where $Q_{b,c}, b, c = 0, 1$ indicates that the state of the VMs changes from $b$ to $c$.

In order to analyze the non-zero sub-blocks of $Q_{b,c}$, we introduce the following notations:

- $I_W$: An identity matrix of $W + 1$ dimension
- $I_R$: An identity matrix of $R$ dimension
- $I_{R \times W}$: An identity matrix of $R \times (W + 1)$ dimension
- $0$: A zero matrix of appropriate dimension
- $\otimes$: Symbol of Kronecker’s product
- $D_l = \text{diag} \left\{ D_l^{(r)}, r \in E_l \right\}, l = 0, 1$
- $A = \text{diag} \{ \mu_r, r \in E_r \}$

Submatrix $Q_{0,0}$ can be given in a $(K + 1) \times (K + 1)$ block-structure form as follows:

$$Q_{0,0} = \begin{bmatrix} L_{0,0} & L_{0,1} & L_{0,2} & \cdots & L_{0,K-1,K} \\ L_{1,1} & L_{1,2} & \cdots & \cdots & L_{1,K-1,K} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ L_{K-1,1} & L_{K-1,2} & \cdots & L_{K-1,K-1} & L_{K-1,K} \\ L_{K,K} \end{bmatrix}$$
where $L_{s,d}, s, d \in \{0, 1, \ldots, K\}$ represents the transition rate submatrix from system level $s$ to $d$ when the VMs stay in the sleep state. It follows that

\[
L_{0,0} = \tilde{D}_0 + H \otimes I_W,
\]

\[
L_{s,s} = \tilde{D}_0 + (H - \alpha I_R) \otimes I_W, \quad s \in \{1, 2, \ldots, K - 1\},
\]

\[
L_{K,K} = \tilde{D}_0 + \tilde{D}_1 + (H - \alpha I_R) \otimes I_W,
\]

\[
L_{s,s+1} = \tilde{D}_1, \quad s \in \{0, 1, \ldots, K - 1\}.
\]

Submatrix $Q_{1,1}$ can be given in an $(N + K) \times (N + K)$ block-structure form as follows:

\[
Q_{1,1} = \begin{bmatrix}
U_{1,1} & U_{1,2} & & & \\
U_{2,1} & U_{2,2} & U_{2,3} & & \\
& \ddots & \ddots & \ddots & \\
& U_{N+K-1,N+K-2} & U_{N+K-1,N+K-1} & U_{N+K-1,N+K} & U_{N+K,N+N} \\
& U_{N+K,N+K} & U_{N+K,N+K-1} & & \\
\end{bmatrix}
\]

where $U_{s,d}, s, d \in \{1, 2, \ldots, N + K\}$ represents the transition rate submatrix from system level $s$ to $d$ when the VMs remain awake. It follows that

\[
U_{s,s} = \tilde{D}_0 + (H - s A) \otimes I_W, \quad s \in \{1, 2, \ldots, N\},
\]

\[
U_{s,s} = \tilde{D}_0 + (H - N A) \otimes I_W, \quad s \in \{N+1, N+2, \ldots, N+K-1\},
\]

\[
U_{N+K,N+K} = \tilde{D}_0 + \tilde{D}_1 + (H - N A) \otimes I_W,
\]

\[
U_{s,s-1} = s A \otimes I_W, \quad s \in \{2, 3, \ldots, N\},
\]

\[
U_{s,s-1} = N A \otimes I_W, \quad s \in \{N+1, \ldots, N+K\},
\]

\[
U_{s,s+1} = \tilde{D}_1, \quad s \in \{1, 2, \ldots, N+K-1\}
\]

where $\otimes$ is the symbol of Kronecker’s product.

Submatrix $Q_{0,1}$ represents the transition rate submatrix for the system being awakened from a sleep period. $Q_{0,1}$ can be given in a $(K + 1) \times (N + K)$ block-structure form as follows:

\[
Q_{0,1} = \begin{bmatrix}
0 & \alpha I_{R \times W} & & & \\
& \alpha I_{R \times W} & \ddots & \ddots & \\
& & \ddots & \alpha I_{R \times W} & \\
& & & \alpha I_{R \times W} & \\
\end{bmatrix}
\]

Submatrix $Q_{1,0}$ represents the transition rate submatrix for the system going into a sleep state from an awake state. $Q_{1,0}$ can be given in an $(N + K) \times (K + 1)$ block-structure form as follows:

\[
Q_{1,0} = \begin{bmatrix}
A \otimes I_W & 0 & & & \\
& 0 & \ddots & & \\
& & 0 & 0 & \\
\end{bmatrix}
\]

There are mainly two methods for computing the stationary distribution of finite state Markov chains: the direct method and the iterative method. If the dimensions
of the infinitesimal generator $Q$ are small, it can be easily solved using the direct method. Otherwise, it can be solved using the iterative method. In this paper, we use the Gauss-Seidel method, one of the iterative methods, to calculate the stationary distribution $\Pi$ of the four-dimensional Markov chain $\{x(t), y(t), h(t), z(t), t \geq 0\}$.

4. Performance measures. In order to evaluate the energy-saving mechanism, we derive some performance measures in terms of the energy saving rate of the system and the average latency of tasks.

We define the energy saving rate of the system as the energy conservation per unit time for the VMs in cloud computing with correlated traffic. In the cloud computing system considered in this paper, an idle VM does not stop working completely, but continues to do some auxiliary works, such as system monitoring. Therefore, the energy consumption of an idle VM is much higher than that of a sleeping VM. In contrast, the difference between the energy consumption of an idle VM and that of a busy VM is not so evident. Therefore, the energy consumption per unit time of a busy VM is supposed to be the same as that of an idle VM. Let $C_b$ be the energy consumption per unit time when the system is in an awake state, $C_v$ be the energy consumption per unit time when the system is in a sleep period, and $C_x$ be the additional energy consumption for the system transition from the sleep state to the awake state. The energy saving rate $\omega$ of the system can be calculated as follows:

$$\omega = (C_b - C_v) \sum_{i=0}^{K} \pi_{0,i} e - C_x \sum_{i=1}^{K} \pi_{0,i} e \alpha \sum_{i=1}^{K} \pi_{0,i} e$$

where $\pi_0$ is the stationary probability vector that the VMs are asleep, $e$ is a column vector with $(W+1) \times R$ elements and all elements of the vector are equal to 1. In the scheme considered in this paper, the energy consumption $C_b$ per unit time when the system is in an awake state is greater than the energy consumption $C_v$ per unit time when the system is in a sleep period, i.e. $C_b > C_v$.

We define the average latency of tasks as the sum of the average waiting time of tasks in the buffer and the average service time of tasks on the VMs. Using Little’s law, the average latency $\sigma$ of tasks can be calculated as follows:

$$\sigma = \frac{1}{\lambda_{\text{out}}} \left( \sum_{i=1}^{K} i \pi_{0,i} e + \sum_{i=1}^{N+K} i \pi_{1,i} e \right)$$

where $\lambda_{\text{out}}$ is the effective arrival rate of the tasks, and $e$ is a column vector with $(W+1) \times R$ elements and all elements of the vector are equal to 1. $\lambda_{\text{out}}$ is calculated as

$$\lambda_{\text{out}} = \sum_{i=1}^{N+K} \min \{i, N\} \pi_{1,i} (A \otimes I_{\Pi}) e.$$

5. Numerical results. In order to numerically evaluate the system performance of the energy-saving mechanism with correlated traffic, we carry out numerical experiments under MATLAB R2016a. We provide numerical results to demonstrate the relationship between the sleep parameter and the performance measures, such as the energy saving rate of the system and the average latency of tasks.

In our numerical experiments, one part of the experimental parameters is set by examples, and the other parts are set by referring to the related literature.

The phase number of the random cloud environment: $R = 2$. 
The state number of the directing process: \( W = 2 \).
The infinitesimal generator of the random cloud environment is as follows:

\[
H = \begin{bmatrix}
-1 & 1 \\
2 & -2
\end{bmatrix}.
\]

The parameter of the service process of tasks at the first environmental phase: \( \mu_1 = 1.3 \). The parameter of the service process of tasks at the second environmental phase: \( \mu_2 = 1.8 \).
The buffer size: \( K = 27 \).
The VM-number: \( N = 6, 7, 8 \).
The energy consumption per unit time when the system is in an awake state: \( C_b = 20 \text{ mW} \). The energy consumption per unit time when the system is in a sleep period: \( C_s = 2 \text{ mW} \). The additional energy consumption for the system transition from the sleep state to the awake state: \( C_x = 10 \text{ mJ} \).

Under the constraint of the fixed average arrival rate of tasks \( \lambda^{(r)} \) and the fixed coefficient of correlation \( c_{\text{cor}}^{(1)} = c_{\text{cor}}^{(2)} = 0 \), we set the stochastical matrices \( D_0^{(r)} \) and \( D_1^{(r)} \).

For the coefficient of correlation \( c_{\text{cor}}^{(1)} = c_{\text{cor}}^{(2)} = 0 \), we set the stochastical matrices as follows:

\[
D_0^{(1)} = \begin{bmatrix}
-10.80 & 2.16 & 2.16 \\
3.00 & -15.00 & 3.00 \\
1.92 & 1.92 & -9.60
\end{bmatrix},
D_1^{(1)} = \begin{bmatrix}
2.16 & 2.16 & 2.16 \\
3.00 & 3.00 & 3.00 \\
1.92 & 1.92 & 1.92
\end{bmatrix},
\]

\[
D_0^{(2)} = \begin{bmatrix}
-13.8 & 2.76 & 2.76 \\
3.72 & -18.6 & 3.72 \\
1.68 & 1.68 & -8.40
\end{bmatrix},
D_1^{(2)} = \begin{bmatrix}
2.76 & 2.76 & 2.76 \\
3.72 & 3.72 & 3.72 \\
1.68 & 1.68 & 1.68
\end{bmatrix}.
\]

By referring to [5], for the coefficient of correlation \( c_{\text{cor}}^{(1)} = 0.1, c_{\text{cor}}^{(2)} = 0.2 \), we set the stochastical matrices as follows:

\[
D_0^{(1)} = \begin{bmatrix}
-18.22 & 0.80 & 0.84 \\
0.95 & -3.34 & 0.58 \\
0.93 & 0.57 & -2.23
\end{bmatrix},
D_1^{(1)} = \begin{bmatrix}
15.78 & 0.50 & 0.30 \\
0.53 & 1.18 & 0.11 \\
0.39 & 0.06 & 0.29
\end{bmatrix},
\]

\[
D_0^{(2)} = \begin{bmatrix}
-23.09 & 0.89 & 0.87 \\
0.76 & -3.36 & 0.69 \\
0.88 & 0.83 & -2.88
\end{bmatrix},
D_1^{(2)} = \begin{bmatrix}
20.78 & 0.44 & 0.12 \\
0.16 & 1.51 & 0.24 \\
0.13 & 0.29 & 0.75
\end{bmatrix}.
\]

Figure 2 demonstrates the change trend for the energy saving rate \( \omega \) of the system versus the sleep parameters \( \alpha \).

From Fig. 2, we observe that for the same VM-number \( N \), the energy saving rate \( \omega \) of the system will decrease as the sleep parameter \( \alpha \) increases. When the sleep parameter increases, the VMs will be awakened earlier from a sleep state, so the VMs are less likely to be asleep. Thus, the energy consumption of the VMs will increase, and the energy saving rate of the system will decrease.

In addition, it appears also from Fig. 2 that for the same sleep parameter \( \alpha \), the energy saving rate \( \omega \) of the system will increase as the VM-number \( N \) increases. The more VMs there are deployed in a PM, the stronger the system service ability is, so the quicker the system will become empty. Consequently, the system will be more likely to be asleep, and the energy saving rate of the system will increase.

From Fig. 2, we can also observe that for the same VM-number \( N \) and sleep parameter \( \alpha \), the energy saving rate \( \omega \) of the system for the MAP with the coefficient
Figure 2. Change trend for the energy saving rate $\omega$ of the system.

The energy saving rate $\omega$ of the system is shown in Figure 2. The energy saving rate is plotted against the sleep parameter $\alpha$. The energy saving rate decreases as the sleep parameter $\alpha$ increases. The system with the coefficient of correlation $c_{\text{cor}}^{(1)} = 0, c_{\text{cor}}^{(2)} = 0$ has a lower energy saving rate compared to the system with the coefficient of correlation $c_{\text{cor}}^{(1)} = 0.1, c_{\text{cor}}^{(2)} = 0.2$.

Figure 3 shows the change trend for the average latency $\sigma$ of tasks versus the sleep parameter $\alpha$.

Figure 3. Change trend for the average latency $\sigma$ of tasks.

From Fig. 3, we find that for the same VM-number $N$, the average latency $\sigma$ of tasks will decrease as the sleep parameter $\alpha$ increases. We note that the tasks...
arriving while the system is in the sleep state have to wait for the expiration of the
sleep timer before getting service. The greater the sleep parameter is, the shorter
the sleep period is, and the earlier the VMs are awakened. Thus, the shorter the
time the tasks have to wait during the sleep state, and the average latency of arrival
tasks will decrease.

On the other hand, from Fig. 3, we also observe that for the same sleep parameter
$\alpha$, the average latency $\sigma$ of tasks will decrease as the VM-number $N$ increases. This
is because the more VMs there are deployed in a PM, the stronger the system
service ability is. Consequently, the shorter time a task will wait in the buffer, and
the average latency of tasks will decrease.

From Fig. 3, we can also observe that for the same VM-number $N$ and sleep
parameter $\alpha$, the average latency $\sigma$ of tasks for the MAP with the coefficient of
correlation $c_{cor}^{(1)} = 0.1$, $c_{cor}^{(2)} = 0.2$ is much longer than that with the coefficient of
correlation $c_{cor}^{(1)} = c_{cor}^{(2)} = 0$.

6. **System optimization.** Comparing Figs. 2 and 3, we see that there is a trade-
off between the energy saving rate $\omega$ of the system and the average latency $\sigma$ of
tasks when setting the sleep parameter in the energy-saving mechanism. In order
to satisfy the Service Level Agreement (SLA) of cloud users, and reduce operating
costs, in this paper, a cost function is established to balance the energy saving rate
$\omega$ of the system and the average latency $\sigma$ of tasks. Considering that the higher the
energy saving rate $\omega$ of the system is, the less the cloud provider will pay in turn,
we use the reciprocal of the energy saving rate $\omega$ of the system in the cost function
$\phi(\alpha)$ [24]. The cost function $\phi(\alpha)$ taking the sleep parameter $\alpha$ as the argument is
given as follows:

$$\phi(\alpha) = C_1 \frac{1}{\omega} + C_2\sigma$$  \hspace{1cm} (6)

where $C_1$ is the weight factor for the energy saving rate of the system and $C_2$ is the
weight factor for the average latency of tasks. $C_1$ and $C_2$ can be set as needed in
practice, and the change trend for results of numerical experiments does not depend
on the values of $C_1$ and $C_2$. In practical applications, we set the values of the weight
factor $C_1$ for the energy saving rate of the system and the weight factor $C_2$ for the
average latency of tasks according to the traffic type. For energy sensitive traffic,
we would set a bigger $C_1$, and for delay sensitive traffic, we would set a bigger $C_2$.
In real-time cloud computing services, we will set a smaller weight for the energy
saving rate of the system and a bigger weight for the average latency of tasks. In
non-real-time cloud computing services, we will set a bigger weight for the energy
saving rate of the system and a smaller weight for the average latency of tasks.

Using the same parameters as those in Section 5, and setting VM-number $N = 6, 7, 8$, buffer size $K = 24, 27, 30, 33$, and the weight factors $C_1 = 11/15$, $C_2 = 4/15$
as an example, we carry out numerical experiments and show the change trends for
the cost function $\phi(\alpha)$ versus the sleep parameter $\alpha$ for different VM-number $N$
and buffer size $K$ in Fig. 4.

It can be observed that all the curves of the cost function $\phi(\alpha)$ experience two
stages in Fig. 4. In the first stage, the cost function $\phi(\alpha)$ decreases with an increase
in the sleep parameter $\alpha$. In the second stage, the cost function $\phi(\alpha)$ increases with
an increase in the sleep parameter $\alpha$. As shown in Figs. 2 and 3, when the sleep
parameter is smaller, the average latency of tasks is the main factor affecting the cost
function. When the sleep parameter increases, the average latency of tasks decreases
and the cost function decreases. As the sleep parameter continues to increase, the impact of the average latency of tasks on the cost function is weakened, and the energy consumption efficiency of the system becomes the main factor affecting the cost function. As the sleep parameter increases, the energy consumption efficiency of the system decreases and the cost function increases. Therefore, as the sleep parameter increases, all the curves of the cost function firstly decrease and then increase. Thus, there is a minimal system cost \( \phi(\alpha^*) \) when the sleep parameter \( \alpha \) is set to be the optimum \( \alpha^* \).

We note that when \( C_1 \) is set bigger than that in Fig. 4, the optimal sleep parameter becomes smaller; when \( C_2 \) is set bigger than that in Fig. 4, the optimal sleep parameter becomes bigger. As a special case, if \( C_1 = 0 \), i.e., the influence of the energy saving rate of the system on the cost function is neglected, the curves of the cost functions become monotone, and the upper bound of the sleep parameter is the optimal one; whereas if \( C_2 = 0 \), i.e., the influence of the average latency of tasks on the cost function is neglected, the curves of the cost functions become monotone, and the lower bound of the sleep parameter is the optimal one.
However, when optimizing the energy-saving mechanism, it is sometimes difficult
to secure an explicit solution for the optimal sleep parameter $\alpha^*$, as well as an
explicit solution for the minimal system cost $\phi(\alpha^*)$. For this, in this paper, we
present an intelligent optimization algorithm to optimize the system parameters of
cloud computing. The Firefly Algorithm (FA) was proposed firstly based on the
biological behavior of fireflies. Compared with other metaheuristic algorithms, such
as PSO, the FA is much more efficient in finding the global optima with higher
success rates [34]. Previous studies [35] show that the proposed FA is superior to
existing metaheuristic algorithms. In this paper, we choose the FA to explore the
optimal sleep parameter.

Here we improve the FA, calling it an “improved FA”, in the following aspects to
optimize the system performance. (i) The improved FA introduces the Chebyshev
chaotic map to initialize the position of the fireflies. (ii) Besides being attracted
to a brighter, adjacent firefly, the improved FA increases certain behaviors in the
fireflies. The increased behaviors include avoidance of collision between individuals,
following (keeping pace with) other individuals, finding a food source and evading
a predator. In the improved FA, there are three idealized assumptions:
(i) The brightness of a firefly is associated with the position of the firefly itself.
The stronger the brightness of the firefly is, the better its position is, the
most advantageous the position of the food in relation to the firefly is, but the
position of the predator is the least advantageous.
(ii) The brightness of a firefly is determined by the return value of the encoded
objective function.
(iii) A firefly can only visually perceive the brightness of other fireflies within a
radius around itself (perception radius).

We now outline the improved FA proposed in this paper.

The family of a Chebyshev map is defined as follows:

$$x_{n+1} = \cos(g \arccos(x_n)), x_n \in [-1, 1]$$  (7)

where $g$ is the order of the map function family.

Let $X$ be the position of the current firefly, $S_r$ be the perception radius of the
current firefly, $N_v$ be the number of fireflies within the visual perception radius $S_r$
of the current firefly. Let $V_j, j = 1, 2, \ldots, N_v$ be the position of the $j$th firefly.

Then we can calculate mathematically four behaviors of fireflies: “Attraction”,
“Collision avoidance”, “Following” and “Predator evasion”, denoted by $Atr$, $Col$, $Fol$ and $Pre$, respectively, using in this paper as follows:

“Attraction” $Atr$ is calculated by

$$Atr = X_+ - X$$  (8)

where $X_+$ is the position of a neighboring firefly within the perception radius around
the current firefly.

“Collision avoidance” $Col$ is calculated by

$$Col = -\frac{1}{N_v} \left( \sum_{j=1}^{N_v} V_j - X \right).$$  (9)

“Following” $Fol$ is calculated by

$$Fol = \frac{1}{N_v} \sum_{j=1}^{N_v} \Delta V_j$$  (10)
where $\Delta V_j$ is the span of fireflies in position $V_j$, $j = 1, 2, \ldots, N_v$. The “Following” will be keeping pace with other individuals.

“Predator evasion” $Pre$ is calculated by

$$Pre = -(X + X_{wor})$$

(11)

where $X_{wor}$ is the position of the predator.

The behavior of a firefly is assumed to be a combination of the four behaviors mentioned above and the “Fly inertia” of the firefly itself. To update the position of a firefly in a search space by simulating the movement of a firefly, we introduce a span $\Delta X$ of the current firefly in position $X$. The span $\Delta X$ of a firefly is calculated as follows:

$$\Delta X_{t+1} = c_t \ast Col + f_t \ast Fol + a_t \ast Atr + p_t \ast Pre + w_t \ast \Delta X_t$$

(12)

where $t$ is the current fly count of the firefly, $t + 1$ is the next fly count of the firefly. $c_t$ is the weight coefficient of the “Collision avoidance”, $f_t$ is the weight coefficient of the “Following”, $a_t$ is the weight coefficient of the “Attraction”, $p_t$ is the weight coefficient of the “Predator evasion”, and $w_t$ is the weight coefficient of the “Fly inertia”, respectively. These weight coefficients are related to the fly process of fireflies. As discussed above, a firefly finds a food source or evades a predator. A firefly tends to follow while maintaining proper separation with neighbor fireflies within the perception radius around the firefly itself in a dynamic swarm. When a firefly is finding a food, the behavior of “Following” is weaker and the behavior of “Collision avoidance” is stronger to attack prey. Therefore, the weight coefficient $f_t$ of the “Following” is bigger, while the weight coefficient $c_t$ of the “Collision avoidance” is smaller. When a firefly is being found by a predator, the behavior of “Following” is stronger and the behavior of “Collision avoidance” is weaker to escape from the predator. Therefore, the weight coefficient $f_t$ of the “Following” is bigger, while the weight coefficient $c_t$ of the “Collision avoidance” is smaller. Updating the visual perception radius $S_r$, we can calculate the weight coefficients $c_t$, $f_t$, $p_t$ and $w_t$ shown in Eq. (12) by Algorithm 1 below in Step 5. Moreover, the weight coefficient $a_t$ of the “Attraction” is given by using the attraction function in reference [34] as follows:

$$a_t = I_0 e^{-\gamma r^2}$$

(13)

where $I_0$ is the maximum attractiveness, $r$ is the distance between two fireflies and $\gamma$ is an absorption coefficient for a fixed brightness.

To increase the global search capability of the improved FA, the visual perception semi-diameter will increase as the fly count of a firefly increases. If there is not any information about other fireflies in the visual perception semi-diameter of a firefly, the firefly will fly around the search space according to the Lévy flight. By referring to [20], Lévy flight $Lev$ is calculated as follows:

$$Lev = 0.01 \times \frac{r_1}{|r_2|^{\frac{1}{\beta}}} \left( \frac{\Gamma (1 + \beta) \sin \left( \frac{\pi \beta}{2} \right)}{\Gamma \left( \frac{1+\beta}{2} \right) \beta 2^{(\frac{2-\beta}{2})}} \right)^{\frac{1}{\beta}}$$
where \( r_1, r_2 \) are two random numbers in \([0,1]\), \( \beta \) is a Lévy flight parameter, and \( \Gamma(n) = (n-1)! \). Thus, the fly span is calculated as follows:

\[
\Delta X_{t+1} = 0.01 \times r_1 \left( \frac{\Gamma(1 + \beta)}{r_2 \Gamma \left( \frac{1 + \beta}{2} \right) \beta^{2 \left( \frac{n-1}{2} \right)}} \right)^{\frac{1}{\beta}} X_t. \tag{14}
\]

The position of a firefly can be updated after the firefly flies away, and the new position of the firefly is calculated by

\[
X_{t+1} = X_t + \Delta X_{t+1}. \tag{15}
\]

The fly count of a firefly is called the iteration as an abbreviation. The main steps of the improved FA are listed in Algorithm 1.

---

**Algorithm 1** The improved FA for finding \( \alpha^* \) and \( \phi(\alpha^*) \).

**Input:** The maximal iteration \( MaxG \), the fireflies population \( N_{fir} \), the search region \([l_d, u_d]\), the maximal run times \( RunN \).

**Output:** The optimal sleep parameter and the minimal system cost.

**Step 1:** Initialize the current run times \( i_{run} = 1 \).

**Step 2:** Initialize the span \( \Delta X_k, k = 1, 2, \ldots, N_{fir} \), the current best position \( X_{best} = -\infty \), the current strongest brightness \( Food_{fit} = 0 \), the current worst position \( X_{worst} = +\infty \), the current weakest brightness \( Enemy_{fit} = 0 \), the firefly individual \( i = 1 \) and the iteration \( \text{iter} = 1 \).

\[
\Delta X_k = (u_d - l_d) / N_{fir}, \quad k = 1, 2, \ldots, N_{fir}
\]

**Step 3:** Initialize the position of each firefly by Chebyshev chaotic map.

\[
p_o = -1 + 2 \times \text{rand} \quad \% \text{ rand returns a sample in the interval (0,1) following the uniform distr}
\%
\]

\[
\text{for } k = 1 : N_{fir} \quad \% \text{ g is a positive integer.}
\]

\[
p_o = \cos(g \times \arccos(p_o)) \\
X_k = l_d + (1 + p_o) \times (u_d - l_d) / 2
\]

**endfor**

**Step 4:** Search the current best position \( X_{best} \) and the current worst position \( X_{worst} \), record the current strongest brightness \( Food_{fit} \) and the current weakest brightness \( Enemy_{fit} \).

\[
\text{for } k = 1 : N_{fir} \quad \% \text{ Collision avoidance, Following, Predator evasion}
\]

\[
\text{if } -\phi(X_k) > Food_{fit} \quad \% \text{ Collision avoidance}
\]

\[
X_{best} = X_k \\
Food_{fit} = -\phi(X_k)
\]

**endif**

\[
\text{if } -\phi(X_k) < Enemy_{fit} \quad \% \text{ Predator evasion}
\]

\[
X_{worst} = X_k \\
Enemy_{fit} = -\phi(X_k)
\]

**endif**

**endfor**

**Step 5:** Update the visual perception radius \( S_r \), calculate the weight coefficient \( c_t \) of the “Collision avoidance”, the weight coefficient \( f_t \) of the “Following”, the weight coefficient \( p_t \) of the “Predator evasion” and the
weight coefficient \( w_t \) of the “Fly inertia”.

\[
S_r = \frac{(u_d - l_d)}{4} + (u_d - l_d) \times \left( \frac{\text{iter}}{\text{MaxG} \times 4} \right)^3
\]

\[
c_t = \text{rand} \times (k_1 - \text{iter} \times (k_1 - k_2)) / \text{MaxG}
\]

\[
f_t = \text{rand} \times (k_3 - \text{iter} \times (k_3 - k_4)) / \text{MaxG}
\]

\[
p_t = \text{rand} \times (k_5 - \text{iter} \times (k_5 - k_6)) / \text{MaxG}
\]

\[
w_t = \text{rand} \times (k_7 - \text{iter} \times (k_7 - k_8)) / (\text{MaxG}/2)
\]

% \( k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8 \) are constants in (0, 1) and \( k_1 > k_2, k_3 > k_4, k_5 > k_6, k_7 > k_8 \).

**Step 6:** Calculate the number \( N_i \) and the position \( V_{ij}, j = 1, \ldots, N_i \), of fireflies within the visual perception radius \( S_r \) of the \( i \)th firefly.

**Step 7:** Update the fly span of the \( i \)th firefly.

calculate the “Collision avoidance” \( Col \) by Eq. (9)

calculate the “Following” \( Fol \) by Eq. (10)

calculate the “Predator evasion” \( Pre \) by Eq. (11)

if \( \text{abs}(X_i - X_{\text{best}}) \leq S_r \)

calculate the weight coefficient \( a_t \) of the “Attraction” \( (X_i \text{ vs. } X_{\text{best}}) \) by Eq. (13)

calculate the “Attraction” \( Atr \) by Eq. (8)

update the span \( \Delta X_i \) of the \( i \)th firefly by Eq. (12)

elseif \( N_i \geq 1 \)

for \( j = 1 : N_i \)

calculate the weight coefficient \( a_t \) of the “Attraction” \( (X_i \text{ vs. } V_{ij}) \) by Eq. (13)

calculate the “Attraction” \( Atr \) by Eq. (8)

update the span \( \Delta X_i \) of the \( i \)th firefly by Eq. (12)

endif

else

update the span \( \Delta X_i \) of the \( i \)th firefly by Eq. (14)
endif

**Step 8:** Update the position of the \( i \)th firefly.

\( X_i = X_i + \Delta X_i \)

**Step 9:** Check the firefly individual.

\( i = i + 1 \)

if \( i \leq N_{\text{fir}} \) goto **Step 5**
endif

**Step 10:** Check the iteration.

\( \text{iter} = \text{iter} + 1 \)

if \( \text{iter} < \text{MaxG} \) goto **Step 4**
endif

**Step 11:** Check the run times.

if \( i_{\text{run}} < \text{RunN} \)

\( X_B(i_{\text{run}}) = X_{\text{best}} \)

\( i_{\text{run}} = i_{\text{run}} + 1 \)

goto **Step 2**
endif

**Step 12:** Calculate the optimal sleep parameter \( \alpha^* \).
\[ \alpha^* = \text{average}(XB(i_{run})) \]

**Step 13:** Output the optimal sleep parameter \( \alpha^* \) and the minimal system cost \( \phi(\alpha^*) \).

The parameters of the improved FA are set as follows by referencing [34]: \( I_0 = 1.0, \gamma = 1.5, g = 3, \beta = 1.5, \text{MaxG} = 250, N_{fir} = 12, l_d = 0.1, u_d = 2.0, k_1 = 0.1, k_2 = 0, k_3 = 0.1, k_4 = 0, k_5 = 0.02, k_6 = 0, k_7 = 0.9, k_8 = 0.5. \)

Using \( D_0^{(r)} \) and \( D_1^{(r)} \) with \( c_{co}^{(1)} = 0.1, c_{co}^{(2)} = 0.2, \) we execute the improved FA to optimize the proposed energy-saving mechanism. The optimal sleep parameter \( \alpha^* \) and the minimal system cost \( \phi(\alpha^*) \) for different VM-number \( N \) and buffer size \( K \) are given in Table 1.

| Buffer size \( K \) | VM-number \( N \) | Optimal sleep parameter \( \alpha^* \) | Minimal system cost \( \phi(\alpha^*) \) |
|---------------------|-----------------|-------------------------------|---------------------------------|
| 6                   | 0.2860          | 0.8414                        |
| 24                  | 0.4127          | 0.6951                        |
| 27                  | 0.4407          | 0.7224                        |
| 30                  | 0.4746          | 0.7462                        |
| 33                  | 0.5141          | 0.7666                        |

From Table 1, we observe that for the same buffer size \( K \), the minimal system cost \( \phi(\alpha^*) \) will decrease as the VM-number \( N \) increases. We note that as the VM-number increases, the energy saving rate \( \omega \) of the system will increase whereas the average latency \( \sigma \) of tasks will decrease. For both of these two cases, the system cost will decrease.

On the other hand, from Table 1, for the same buffer size \( K \), we also observe that the change trend for the minimal system cost \( \phi(\alpha^*) \) becomes slower as the VM-number \( N \) increases. The reason is that as the VM-number increases, the changes for both of the energy saving rate \( \omega \) of the system and the average latency \( \sigma \) of tasks are smaller. Therefore, the downtrend for the system cost is gradual.

Comparing the optimization results in Table 1 with those in Fig. 4, we find that the optimal sleep parameters obtained by the improved FA are consistent with the global optimal points observed in Fig. 4. Therefore, we can verify that for the cloud computing system considered in this paper, the optimal results obtained by the improved FA are the global optimal values.
7. Conclusion. In this paper, we evaluated and optimized the energy-saving mechanism considering the correlation in cloud traffic. We considered the correlated arrival of cloud data requests affected by the random cloud environment to follow a Markovian Arrival Process (MAP), and established a MAP/M/N/N+K queue with a synchronous multi-vacation accordingly to analyze the system performance. Based on the system analysis in the steady state, two performance measures, namely, the energy saving rate of the system and the average latency of tasks with the energy-saving mechanism were derived. For the correlated traffic in a random cloud environment, numerical experiments were carried out to illustrate the change trends for the average latency for tasks and the energy saving rate of the system. A cost function taking sleep parameters as an argument was constructed to balance different performance measures. Furthermore, we designed an improved FA to optimize the energy-saving mechanism. Optimization results show that an appropriate increase in the VM-number deployed in a PM will minimize the system cost.

Restricted by our experimental conditions, in this paper, one part of the experimental parameters is set by examples, and the other parts are set by referring to the related literature. This is the limitation of our study. In our future work, we will try to set the experimental parameters based on the real data from practical cloud systems.

Acknowledgments. This work was supported in part by National Science Foundations (Nos. 61872311, 61973261) and Natural Science Foundation of Hebei Province (F2017203141), China, and was supported in part by MEXT and JSPS KAKENHI Grant (No. JP17H01825), Japan.

REFERENCES

[1] P. Bertoldi, M. Avgerinou and L. Castellazzi, Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency, Technical Report, Publications Office of the European Union, Luxembourg, 2017.
[2] C. Cheng, J. Li and Y. Wang, An energy-saving task scheduling strategy based on vacation queuing theory in cloud computing, Tsinghua Science and Technology, 20 (2015), 28–39.
[3] B. D. Choi, B. Kim and D. Zhu, MAP/M/c queue with constant impatient time, Mathematics of Operations Research, 29 (2004), 309–325.
[4] D. Ding, X. Fan, Y. Zhao, K. Kang, Q. Yin and J. Zeng, Q-learning based dynamic task scheduling for energy-efficient cloud computing, Future Generation Computer Systems, 108 (2020), 361–371.
[5] S. A. Dudin and O. S. Dudina, Call center operation model as a MAP/PH/N/R – N system with impatient customers, Problems of Information Transmission, 47 (2011), 364–377.
[6] O. Dudina and S. Dudin, Queuing system MAP/M/N/N+K operating in random environment as a model of call center, in BWWQT, Minsk, Belarus, 2013, 83–92.
[7] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, in MHS’95, Nagoya, Japan, 1995, 39–43.
[8] A. H. Gandomi, X.-S. Yang, S. Talatahari and A. H. Alavi, Firefly algorithm with chaos, Communications in Nonlinear Science and Numerical Simulation, 18 (2013), 89–98.
[9] Q.-M. He, Fundamentals of Matrix-Analytic Methods, Springer, New York, 2014.
[10] T. Hirai, H. Masuyama, S. Kasahara and Y. Takahashi, Performance optimization of parallel-distributed processing with checkpointing for cloud environment, Journal of Industrial and Management Optimization, 14 (2018), 1423–1442.
[11] X. Huang, D. Wu and N. Zhao, Study of performance measures and energy consumption for cloud computing centers based on queueing theory, Journal of Physics: Conference Series, 1631 (2020), 25–26.
[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, S. Hao, X. Qie and W. Yue, A virtual machine scheduling strategy with a speed switch and a multi-sleep mode in cloud data centers, Journal of Systems Science and Systems Engineering, 28 (2019), 194–210.

[14] S. Jing, S. Ali, K. She and Y. Zhong, State-of-the-art research study for green cloud computing, The Journal of Supercomputing, 65 (2013), 445–468.

[15] H. Khazaei, J. Mišić and V. B. Mišić, Performance analysis of cloud computing centers, in QShine, Houston, USA, 2010, 251–264.

[16] Q.-L. Li and Y. Q. Zhao, A MAP/G/1 queue with negative customers, Queueing Systems, 47 (2004), 5–43.

[17] Y. Liu, L. Wang, X. Wang, X. Xu and P. Jiang, Cloud manufacturing: Key issues and future perspectives, International Journal of Computer Integrated Manufacturing, 32 (2019), 858–874.

[18] L. Luo, W. Wu and F. Zhang, Energy modeling based on cloud data center, Journal of Software, 25 (2014), 1371–1387.

[19] A. Manzoor, Cloud Security: Concepts, Methodologies, Tools, and Applications, IGI Global, Hershey, PA, 2019.

[20] S. Mirjalili, Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Computing and Applications, 27 (2016), 1053–1073.

[21] B. K. Patle, A. Pandey, A. Jagadeesh and D. R. Parhi, Path planning in uncertain environment by using firefly algorithms, Defence Technology, 14 (2018), 691–701.

[22] T. Phung-Duc and K. Kawanishi, Multiserver retrial queue with setup time and its application to data centers, Journal of Industrial and Management Optimization, 15 (2019), 15–35.

[23] QYResearch, Global Cloud Accounting Software Market Size, Status and Forecast 2025, Technical Report, Albany, NY, 2018.

[24] J. Shaler Stidham, Optimal Design of Queueing Systems, Chapman and Hall, New York, 2009.

[25] G. Shao and J. Chen, A load balancing strategy based on data correlation in cloud computing, in UCC, Shanghai, China, 2016, 364–368.

[26] N. Sharma and R. Guddeti, Multi-objective energy efficient virtual machines allocation at the cloud data center, IEEE Transactions on Services Computing, 12 (2019), 158–171.

[27] M. J. Usman, A. S. Ismail, G. Abdul-Salaam, H. Chizari, O. Kaiwartya, A. Y. Gital, M. Abdulllahi, A. Aliyu and S. I. Dishing, Energy-efficient nature-inspired techniques in cloud computing datacenters, Telecommunication Systems, 71 (2019), 275–302.

[28] J. Vilaplana, F. Solsona, I. Teixidó, J. Mateo, F. Abella and J. Rius, A queuing theory model for cloud computing, The Journal of Supercomputing, 69 (2014), 492–507.

[29] G.-G. Wang, L. Guo, H. Duan and H. Wang, A new improved firefly algorithm for global numerical optimization, Journal of Computational & Theoretical Nanoscience, 11 (2014), 477–485.

[30] X. Wang, J. Zhu, S. Jin, W. Yue and Y. Takahashi, Performance evaluation and social optimization of an energy-saving virtual machine allocation scheme within a cloud environment, Journal of the Operations Research Society of China, 8 (2020), 561–580.

[31] Y. C. Wang, J. S. Wang and F. H. Tsai, Analysis of discrete-time space priority queue with fuzzy threshold, Journal of Industrial and Management Optimization, 5 (2009), 467–479.

[32] Z.-Q. Wu and X.-B. Zhao, Frequency $H_2/H_\infty$ optimizing control for isolated microgrid based on IPSO algorithm, Journal of Industrial and Management Optimization, 14 (2018), 1565–1577.

[33] X. Yan, S. Jin, W. Yue and Y. Takahashi, A MAP-based performance analysis on an energy-saving mechanism in cloud computing, in QTNA, Ghent, Belgium, 2019, 369–378.

[34] X.-S. Yang, Firefly algorithms for multimodal optimization, in SAGA, Sapporo, Japan, 2009, 169–178.

[35] X.-S. Yang, Firefly algorithm, stochastic test functions and design optimisation, International Journal of Bio-Inspired Computation, 2 (2010), 78–84.

[36] H. Yeganeh, A. Salahi and M. A. Pourmina, A novel cost optimization method for mobile cloud computing by capacity planning of green data center with dynamic pricing, Canadian Journal of Electrical and Computer Engineering, 42 (2019), 41–51.

[37] W. Zhao, X. Wang, S. Jin, W. Yue and Y. Takahashi, An energy efficient task scheduling strategy in a cloud computing system and its performance evaluation using a two-dimensional continuous time Markov chain model, Electronics, 8 (2019), 775–790.
[38] Z. Zhou and Z. Zhou, A MAP/M/N retrial queueing model with asynchronous single vacations, in ICVRIIS, Changsha, China, 2018, 245–249.

Received September 2020; 1st revision January 2021; 2nd revision April 2021.

E-mail address: yanxuena@126.com
E-mail address: jsf@ysu.edu.cn
E-mail address: yue@konan-u.ac.jp
E-mail address: takahashi@i.kyoto-u.ac.jp