An Efficient Green Control Algorithm in Cloud Computing for Cost Optimization

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

Abstract— Cloud computing is a new paradigm for hosting and delivering remote computing resources through a network. However, achieving an energy-efficiency control and simultaneously satisfying a performance guarantee have become critical issues for cloud providers. In this paper, three power-saving policies are implemented in cloud systems to mitigate server idle power. The challenges of controlling service rates and applying the N-policy to optimize operational cost within a performance guarantee are first studied. A cost function has been developed in which the operational costs of power consumption, system congestion and mode-switching are all taken into consideration. The effect of energy-efficiency controls on response times, operating modes and incurred costs are demonstrated and compared. Our objectives are to find the optimal service rate and mode-switching restriction, so as to minimize cost within a response time guarantee under varying arrival rates. An efficient green control (EGC) algorithm is first proposed for solving constrained optimization problems and making costs/performances tradeoffs in systems with different power-saving policies. Simulation results show that the benefits of reducing operational costs and improving response times can be verified by applying the power-saving policies combined with the proposed algorithm as compared to a typical system with a same performance guarantee.

Index Terms— Cost optimization, energy-efficiency control, response time, power-saving policy

1. INTRODUCTION

Cloud computing is a new service model for sharing a pool of computing resources that can be rapidly accessed and released based on a converged infrastructure. In the past, an individual use or company can only use their own servers to manage application programs or store data. Thus, it may cause the dilemma of complex management and cost burden in "own-and-use" patterns. Nowadays, resources provided by Cloud allow users to get on-demand access with minimal management effort based on their needs. Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) are all existing service models. For example, Amazon Web Services is a well-known IaaS that lets users perform computations on the Elastic Compute Cloud (EC2). Google's App Engine and Salesforce are public clouds for providing PaaS and SaaS, respectively [1], [2].

To satisfy uncertain workloads and to be highly available for users anywhere at any time, providing more resources are required. Consequently, resource over-provisioning [3] and redundancy are common situations in a traditional operating system. However, most electricity-dependent facilities will inevitably suffer from idle times or under-utilized for some days or months since there usually have off-seasons caused by the nature of random arrivals. In fact, servers are only busy 10-30% of the time on average. An active server but let it remain in an idle mode still consumes more than 60% power at its peak power [4].

As cloud computing is predicted to grow, substantial power consumption will result in not only huge operational cost but also tremendous amount of carbon dioxide (CO_2) emissions [5], [6]. Therefore, an energy-efficient control, especially in mitigating server idle power has become a critical concern in designing a modern green cloud system. Ideally, shutting down servers when they are left idle during off-peak periods is one of the most direct ways to reduce power consumption. Unfortunately, some negative effects are caused under improper system controls.

First, burst arrivals may experience latency or be unable to access services. Second, there has a power consumption overhead caused by awakening servers from a power-off state too frequently. Third, the worst case is violating a Service Level Agreement (SLA) due to the fact that shutting down servers may sacrifice quality of service (QoS) [7], [8]. The SLA is known as an agreement in which QoS is a critical part of negotiation. To reach a consensus on service contents, negotiating the SLA contract is an essential process by all interested parties before service starts [9].

Performance levels (such as waiting time, queuing length) and billing are formulated in the SLA according to users' expectation and system's ability. A performance guarantee will be specified in terms of constraint, that is, when either party violates the contact, the SLA penalty is given based on the degree of severity. In short, reducing power consumption in a cloud system has raised several concerns, without violating the SLA constraint or causing additional power consumption are both important in system controls.

To avoid switching too often, a control approach called N policy, defined by Yadin and Naor [10] had been

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

extensively adopted in a variety of fields, such as computer systems, communication networks, wireless multimedia, etc. Queuing systems with the N policy will turn a server on only when items in a queue is greater than or equal to a predetermined N threshold, instead of activating a power-off server immediately upon an item arrival. However, it may result in performance degradation when a server stays in a power-saving mode too long under a larger controlled N value.

In this paper, the main contributions are summarized as follows.

- Three power-saving policies that (a) switching a server alternately between idle and sleep modes, (b) allowing a server repeat sleep periods and (c) letting a server stay in a sleep mode only once in an operation cycle are all considered for comparison. The main objective is to mitigate or eliminate unnecessary idle power consumption without sacrificing performances.
- The challenges of controlling the service rate and applying the N-policy to minimize power consumption and simultaneously meet a response time guarantee are first studied. To address the conflict issue between performances and power-saving, a tradeoff between power consumption cost and system congestion cost is conducted.
- An efficient green control (EGC) algorithm is proposed to optimize the decision-making in service rates and mode-switching within a response time guarantee by solving constrained optimization problems. As compared to a typical system without applying the EGC algorithm, more cost-saving and response time improvements can be achieved.

The remainder of this paper is structured as follows. Section 2 gives a brief overview of the problems related to the power management and cost-awareness in cloud systems. Systems with three power-saving policies and decision processes are given in Section 3. Optimization problem formulation and the EGC algorithm are presented in Section 4. In Section 5, the effectiveness of the proposed algorithm is proved and different power-saving policies are compared via experiments. Finally, the conclusions and future works are presented in Section 6.

2. Related Work

Power savings in cloud systems have been extensively studied on various aspects in recent years, e.g., on the virtual machine side by migrating VMs, applying consolidation or an allocation algorithm, and on the data center infrastructure side through a resource allocation, an energy management, etc.

2.1 Power-saving in Virtual Machine

In [11], Huang, Li and Qian studied the virtual machine placement problem with a goal of minimizing the total energy consumption. A multi-dimensional space partition model and a virtual machine placement algorithm were presented. When a new VM placement task arrived, their algorithm checked the posterior resource usage state for each feasible PM, and then chose the most suitable PM according to their proposed model to reduce the number of running PMs.

In [12], Nathuji et al. considered the problem of providing power budgeting support while dealing with many problems that arose when budgets virtualized systems. They managed power from a VM-centric point of view, where the goal was to be aware of global utility tradeoffs between different virtual machines (and their applications) when maintaining power constraints for the physical hardware on which they ran. Their approach to VM-aware power budgeting used multiple distributed managers integrated into the Virtual Power Management (VPM) framework.

Two issues in energy conservation algorithm were addressed in [13] – the placement of virtual machine image and the characteristics of virtual machines. Yang, Liu, and Wu proposed an approximation algorithm (SEP-Pack) and two dynamic programming to consolidate virtual machines. Despite that the dynamic programming could find the optimal solution, its time complexity was prohibitive in practice. As a result, an approximation algorithm was developed, which guaranteed that it could find a solution in time.

In [14], the energy efficiency from the performance perspective was studied. Firstly, Ye et al. presented a virtual machine based energy-efficient data center architecture for cloud computing. Then, they investigated the potential performance overheads caused by server consolidation and lived migration of virtual machine technology. The potential performances overheads of server consolidation were evaluated.

2.2 Power-saving in Computing Infrastructure

In [15], Duggan and Young presented a basic theoretical model and used it in building managing, micro-grids, and datacenter energy management. They analyzed these disparate energy management systems and defined a model for resource allocation that could be used for these and other energy management systems. The Datacenter Energy Management project was focused on modeling energy consumption in data centers, with a goal to optimize electricity consumption. Their project was focused on collecting data to define basic fuel consumption curves.

In [16], Mazzucco, Dyachuky, and Detersy addressed the problem of maximizing the revenues of Cloud providers by trimming down their electricity costs. Policies

were based on dynamic estimates of user demand, and system behavior models. Some approximations were used to handle the resulting models. They had demonstrated that decisions, such as how many servers were powered on, can have a significant effect on the revenue earned by the provider. However, no startup power draw or performance guarantees was considered.

In [17], Zhang et al. presented Harmony, a Heterogeneity-Aware Resource Monitoring and management system that was capable of performing dynamic capacity provisioning (DCP) in heterogeneous data centers. Using standard K-means clustering, they showed that the heterogeneous workload could be divided into multiple task classes with similar characteristics in terms of resource and performance objectives. The DCP was formulated as an optimization problem that considered machine and workload heterogeneity as well as reconfiguration costs.

A framework used to automatically manage computing resources of Cloud infrastructures was proposed in [18] to simultaneously achieve suitable QoS levels and to reduce the amount of energy used for providing services. Guazzone, Anglano and Canonico showed that via discrete-event system (DES) simulation, their solution was able to manage physical resources of a data center in such a way to significantly reduce SLO violations with respect to a traditional approach. The energy-efficiency of the infrastructure was defined as the amount of energy used to serve a single application request.

In [19], Amokrane et al. proposed Greenhead, a holistic resource management framework for embedding virtual data centers across geographically distributed data centers connected through a backbone network. The goal of Greenhead was to maximize the cloud provider's revenue while ensuring that the infrastructure was as environment-friendly as possible. They conducted extensive simulations of four data centers connected through the NSFNet topology. Choosing data center locations supplied by green energy sources could greatly reduce environmental pollution.

In [20], the objectives were to optimize the network performance, the CO2 emissions, the capital expenditures, and the operational expenditures. The objective of cloud computing was to minimize the power consumption of the network. Their proposed model allowed planners to evaluate different solutions and to make variations in the optimization priorities. Although power management in Cloud has attracted considerable research attention, few studies focused on effectively reducing idle server power draw. To the best of our knowledge, applying the N-policy for optimizing the mode-switching control and simultaneously achieving the minimum cost under a performance guarantee has not been considered before.

2.3 Cost optimization in Cloud

The problem of cost optimization is one of the major research topics in cloud computing; here, we refer to some efforts from different aspects [21, 22, 23, 24]. The InterCloud environment was proposed in [21] to support scaling of applications across multiple vendor clouds. Buyya, Ranjan, and Calheiros showed that cost associated with the processing increased significantly at higher rates. The vision, challenges, and architectural elements of InterCloud for utility-oriented federation of Cloud computing environments were presented. Results demonstrated that the federated Cloud computing model had immense potential as it offered cost saving under dynamic workload scenarios.

Leitner, Hummer, and Dustdar formalized the problem of finding the optimal set of adaptations, which minimized the total costs arising from SLA violations for the service provider and the adaptations to prevent them[22]. This problem of cost-effective had been modeled as a one-dimensional discrete optimization problem. A possible algorithms was presented to solve this complex optimization problem, and detailed an end-to-end system based on their earlier work on the PREvent (prediction and prevention based on event monitoring) framework, which clearly indicated the usefulness of their model.

In [23], Zhou, and He developed a cost model that guided planner to efficiently find an optimized transformation for a predefined goal (e.g., minimizing the monetary cost with a given performance requirement). An arbitrary performance and cost optimization process could be represented as a transformation plan (i.e., a sequence of basic transformation operations). All transformations formed a huge optimization space. The two components were designed to be extensible for user requirements on a performance and cost.

A resource allocation problem was considered in [24] that aimed to minimize the total energy cost of cloud computing system while meeting the specified client-level SLAs in a probabilistic sense. An efficient heuristic algorithm based on convex optimization and dynamic programming was presented to solve the aforesaid resource allocation problem. Simulation results demonstrated that considering the SLA with effective VM placement could help to minimize the operational cost in the cloud computing system. Unlike previous studies, our paper contributes to investigate an essential tradeoff between power consumption costs and system performances, so as to optimize operational cost in systems with different power-saving policies.

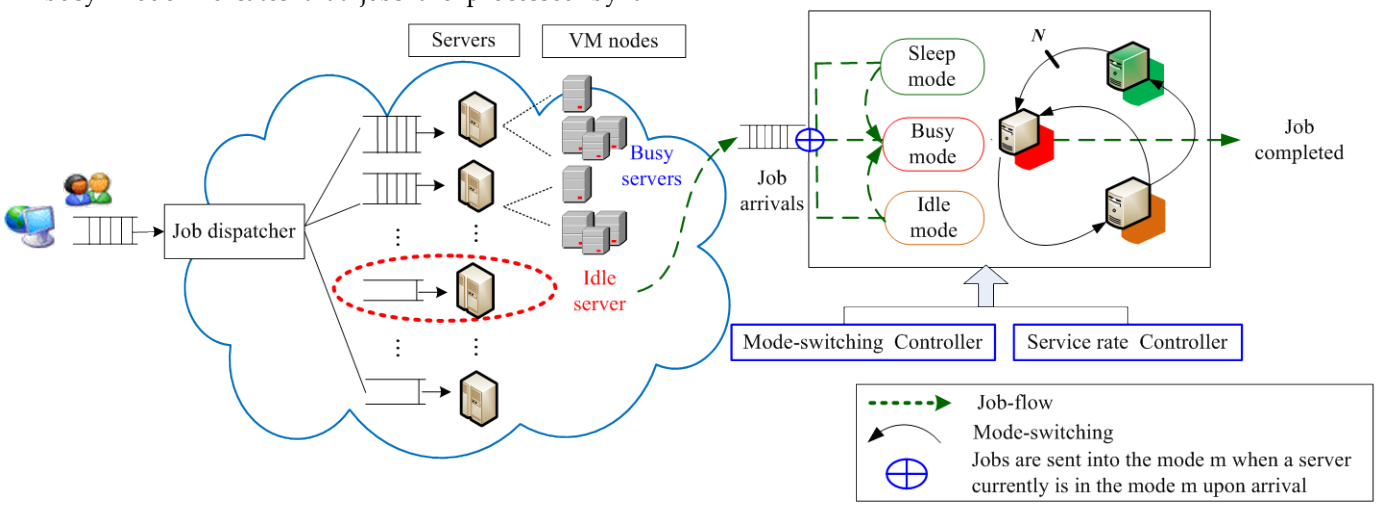
3 Power Management in Clouds

3.1. A cloud Service System and ISN policy

Briefly speaking, a distributed service system consists of lots of physical servers, virtual machines (VMs) and a job

dispatcher, as shown in Fig. 1 (a). The job dispatcher in our designed system is used to identify an arrival job request and forward it to a queue of a corresponding VM manager that can satisfy its QoS levels, meet its target web application or specific requirements. When there has no job in a queue or no job is being processed, a server becomes idle and it remains until a subsequent job has been sent to its processor node. Generally, a server operates alternately between a busy mode and an idle mode for a system with random job arrivals in a cloud environment.

A busy mode indicates that jobs are processed by a server running in one or more of its VMs'; and an idle mode indicates that a server remains active but no job is being processed at that time. To mitigate or eliminate idle power wasted, three power-saving policies with different energy-efficient controls, decision processes and operating mode configurations are presented. First, we try to make an energy-efficient control in a system with three operating modes m= { Busy, Idle, Sleep}, where a sleep mode would be responsible for saving power consumption, as shown in Fig. 1 (b). A job flow is indicated by the dashed line; and a server mode-switching is indicated by the solid line.



(a) A typical system (b) A system with the ISN policy

Fig. 1. (a) A typical cloud system (b) A system with the ISN policy.

A server is allowed to stay in an idle mode for a short time when there has no job in a queue or no job is being processed, rather than switching abruptly into a sleep mode right away when a system becomes empty [25]. An idle mode is the only operating mode that connects to a sleep mode. A server doesn't end its sleep mode even if there has a job arrival; it begins to work only when the number of jobs in a queue is more than the controlled N value. The energy-efficient control with the N policy is denoted by ***N*** in Fig. 1 (b). According to the switching process (from **I**dle to **S**leep) and the energy-efficient control (**N** policy), we have called such an approach the "ISN policy". In the following, Fig. 2 illustrates the step-by-step decision processes and job flows of the ISN policy.

Step1. Incoming jobs can obtain normal service while a server is in a busy mode. A server ends its busy mode when the current job requests have been finished and the queue becomes empty.

Step2. A server stays in an idle mode and waits for subsequently arriving jobs before switching into a sleep mode.

Step3. If a job arrives during an idle period, a server can switch into a busy mode and start to work immediately. Then, a server begins a next idle period until all job requests have been successfully completed.

Step4. If there has no job arrival, a server switches into a sleep mode when an idle period expires.

Step5. A server remains in a sleep mode if the number of jobs in the queue is less than the controlled N value. Otherwise, a server switches into a busy mode and begins to work.

Basically, there are two cases of starting a busy mode:

Case 1: starting a busy mode when a job arrives in an idle mode;

Case 2: starting a busy mode if the number of jobs in the queue is more than the N value when a sleep period expires.

Although power is wasted in allowing a server to stay in idle modes in a non-load period, there have two main benefits of adopting the ISN policy. First, it gives arrival jobs more possibilities of getting immediately service without latency. Second, the total startup cost can be reduced since a server still remains active in an idle mode at its regular service rate.

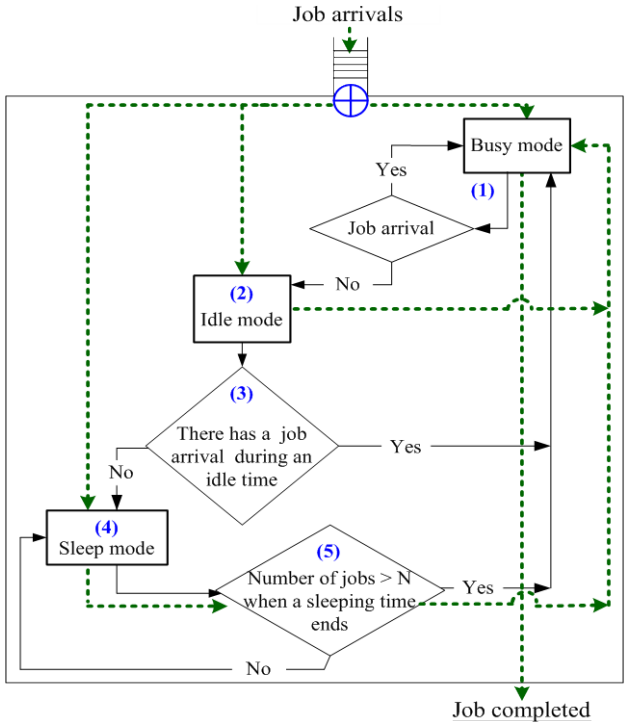


Fig. 2. Decision processes of the ISN policy.

3.2 SN and SI policies

To greatly reduce idle power consumption, non-idle mode operating is designed in another approach, where it only holds {Busy, Sleep} operating modes, as shown in Fig. 3 (a). Instead of entering into an idle mode, a server immediately switches into a sleep mode when system becomes empty. Similarly, a server switches into a busy mode depending on the number of jobs in the queue to avoid switching too often, the switching restriction is denoted by *N* in Fig.3 (a). According to the switching process (directly to Sleep) and the energy-efficient control (N policy), we have called such an approach the "SN policy". Fig. 3 (b) illustrates the step-by-step decision processes and job flows of the SN policy.

Step1. A server switches into a sleep mode immediately when no job is in the system.

Step2. A server stays in a sleep mode if the number of jobs in the queue is less than the N value; otherwise, a server switches into a busy mode and begins to work.

For comparison, the other policy is designed with no mode-switching restriction and performed under the other energy-efficient control, as shown in Fig. 4 (a). A server switches into a sleep mode right away rather than an idle mode when there has no job in a system. This is similar to the SN policy but a server only stays in a sleep mode for a given time.

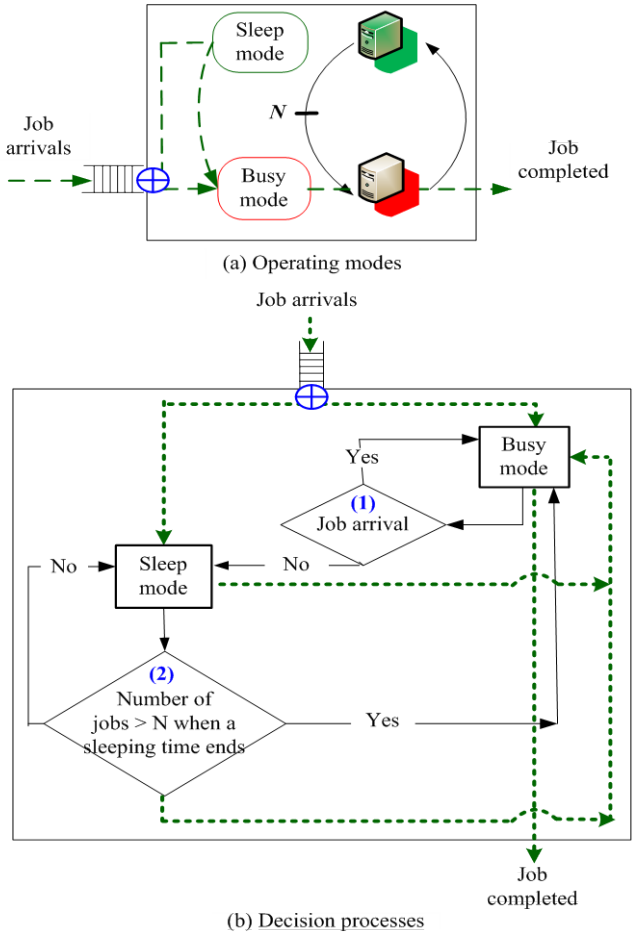
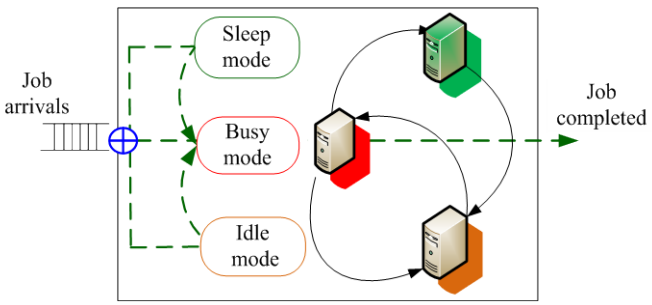


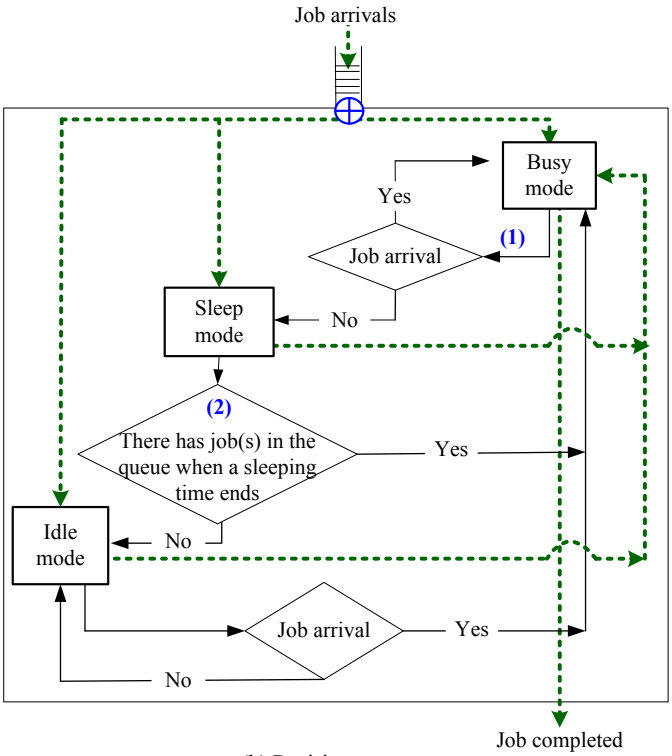
Fig. 3. (a) Operating modes (b) Decision processes with the SN policy.

When a sleeping time expires, it will enter into an idle mode or a busy mode depending upon whether a job has arrived in the queue or not. According to the switching process (from Sleep to Idle), we have called such an approach "SI policy". Fig. 3 (b) illustrates the step-by-step decision processes and job flows of the SI policy.

Step 1: A server immediately switches into a sleep mode instead of an idle mode when there has no job in the system.

Step 2: A server can stay in a sleep mode for a given time in an operation period. If there has no job in the queue when a sleeping time expires, a server will enter into an idle mode. Otherwise, it switches into a busy mode without any restriction and begins to work.





(b) Decision processes

Fig. 4. (a) Operating modes (b) Decision processes of the SI policy.

4 Objective Function

4.1 Queuing Models

Three power-saving policies follow the identical assumptions as follows. It is assumed that job request arrivals follow a Poisson process with parameter λ and they are served in order of their arrivals, that is, the queue discipline is based on the First Come First Served (FCFS). All service times are independent and exponentially distributed with mean $1/\mu$ and the system utilization is $\rho=\lambda/\mu$, which is required to be less than one for a stable state.

Idle times that those follow the exponential distribution with mean $1/\Theta_i$ and follow a fixed (deterministic) time with mean $1/\Theta_d$ are both considered in the ISN policy, denoted by ISN-1 and ISN-2, respectively. A sleep length is exponentially distributed with mean $1/\Theta_s$ and both aforementioned variables are independent of each other. Here the state space is settled by $S = \{(n, m), 0 \leqq n < \infty, m=\{0, 1, 2\}\}$ where n denotes the number of jobs in the system, and m denotes that the system is in the mode m. The state-transition-rate diagram for a queuing system with the ISN-1 policy is shown in Fig. 5. State (0, 1) denotes that the system is in an idle mode when there has no job in the system; state (n, 0), $n \geq 1$ indicates that the system is in a regular busy mode when there have n jobs in the system; state (n, 2), $n \geq 0$ indicates that the system is in a sleep mode when there have n jobs in the system.

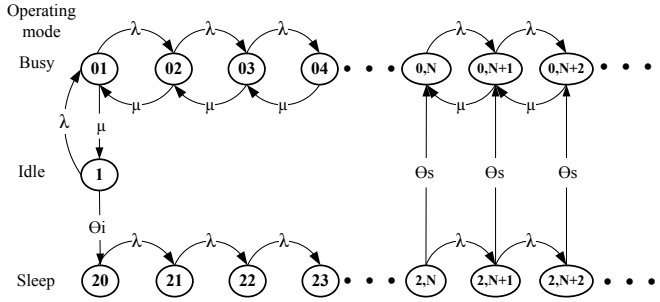


Fig. 5 The state-transition-rate diagram for a queuing model with the ISN policy.

Let P_{mn} denote the steady-state probabilities at state (n, m), then the following notations are used in a steady state:

P_{0n}≡ Probability that there have n jobs in the system when a server is in a busy mode;

P_1 ≡ Probability that there has no job in the system when a server is in an idle mode;

P_{2n}≡ Probability that there have n jobs in the system when a server is in a sleep mode.

Based on Fig. 5, the following balanced equations can be given.

$$m=0\begin{cases}(\lambda+\mu)P_{01}=\lambda P_{10}+\mu P_{02} & k=1\\ (\lambda+\mu)P_{0n}=\lambda P_{0,n-1}+\mu P_{0,n+1} & (k=2,\ldots,K-1)\\ (\lambda+\mu)P_{0n}=\lambda P_{0,n-1}+\mu P_{0,n+1}+\theta_s P_{2n} & (k=N,N+1,\cdots)\end{cases} \quad (1)$$

$$m=1\quad (\lambda+\theta_i)P_{10}=\mu P_{01} \quad (2)$$

$$m=2\begin{cases}\lambda P_{20}=\theta_s P_{10} & k=0\\ \lambda P_{2k}=\lambda P_{2,k-1} & (k=1,2,\ldots,N-1)\\ (\lambda+\theta_s)P_{2k}=\lambda P_{2,k-1} & (k=N,N+1,\cdots)\end{cases} \quad (3)$$

Let P_B, P_I and P_S denote the probabilities that a server is in a busy, idle and sleep mode, respectively. With the normalizing equation $\sum_{n=1}^{\infty}P_{0n}+P_1+\sum_{n=0}^{\infty}P_{2n}=1$, the solutions of these equations [25] can be obtained as follows.

$$\begin{cases}P_B=\sum_{n=1}^{\infty}p_1(\frac{\lambda}{\mu})^n+\sum_{n=1}^{\infty}\sum_{i=0}^{n-1}(\frac{\lambda}{\mu})^{n-1}p_{2i}\\ P_I=\frac{\lambda\theta_s(1-\rho)}{\lambda\theta_i+\lambda\theta_s+K\theta_d\theta_s}\\ P_S=\sum_{n=0}^{K-1}\frac{\theta_i}{\lambda}p_1+\sum_{n=K}^{\infty}(\frac{\lambda}{\lambda+\theta_s})^{n-K+1}p_{20}\end{cases} \quad (4)$$

The balanced equations for a queuing model with the ISN-2 policy are given as follows.

$$m=0\begin{cases}(\lambda+\mu)p_{01}=\lambda\int_0^{1/\theta_d}p_{10}(x)dx+\mu p_{01}\\ (\lambda+\mu)p_{0n}=\lambda p_{0,n-1}+\mu p_{0,n+1} & (n=2,3,\ldots,N-1)\\ (\lambda+\mu)p_{0n}=\lambda p_{0,n-1}+\mu p_{0,n+1}+\theta_s p_{2n} & (n=N,N+1,\ldots)\end{cases} \quad (5)$$

$$m=1 \begin{cases} p_{10}(0)=\mu p_{01} \\ \dfrac{d}{dx}p_{10}=-\lambda p_{10} \quad 0<x<\dfrac{1}{\theta_d} \end{cases} \tag{6}$$

$$m=2 \begin{cases} \lambda p_{02}=\dfrac{1}{\theta_d}p_{10} \\ \lambda p_{2n}=\lambda p_{2,n-1} \quad (n=1,\ 2,\ 3,\ldots,\ N-1) \\ (\lambda+\theta_s)p_{2n}=\lambda p_{2,n-1} \quad (n=N,\ N+1,\ldots) \end{cases} \tag{7}$$

By using the normalizing equation $\sum_{n=1}^{\infty}p_{0n}+\int_0^{1/\theta_d}p_{10}(x)dx+\sum_{n=0}^{\infty}p_{2n}=1$, the solutions can be obtained as follows. Besides, the derivation of the mean queuing length and the response time can be calculated by using the Laplace-Stieltjes (LS) transforms and the Little's formula [25].

$$\begin{cases} P_B=\sum_{n=1}^{\infty}p_1(\frac{\lambda}{\mu})^n+\sum_{n=1}^{\infty}\sum_{i=0}^{n-1}(\frac{\lambda}{\mu})^{n-i}p_{2i} \\ P_I=\dfrac{(1-\rho)\theta_s(e^{\lambda/\theta_d}-1)}{\lambda-\theta_s+e^{\lambda/\theta_d}\theta_s+N\theta_s} \\ P_S=\sum_{n=0}^{K-1}\dfrac{(1-\rho)\theta_s}{\lambda-\theta_s+e^{\lambda/\theta_d}\theta_s+N\theta_s}+\sum_{n=K}^{\infty}\left(\dfrac{\lambda}{\lambda+\theta_s}\right)^{j-n+1}p_{0,2} \end{cases} \tag{8}$$

In the real-world, observed in an application might not exactly be processed by a single service node. There may be some job requests that need to be performed serially at multiple service stages. Then, applying phase-type distributions allow us to consider a more general situation. Let k be the number of phases in the service station. To represent a queuing model in which the offered service is a series of k identical phases, the Erlang-k service model is adopted and controlled by the SN policy. It is assumed that the service times follow an Erlang K-type distribution with mean 1/(kμ) for each phase. In the Erlang-k service model, a job request is sent into the first stage of the service (say stage k), then carried out through the remaining stages. The system outputs a job until the last service stage (say stage 1) has been completed, as shown in Fig 6.

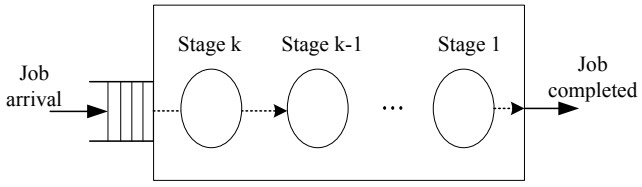


Fig. 6 The Erlang-k service model

The state of the system is described by the triplet (i, j, n), i= 0, 1, 2,…, j = 0, 1, …, k and n= 0, 1, where i denotes the number of jobs in the system, and j denotes that the job is in the service stage j, and n denotes that a server is in the mode n. The state transition-rate diagram of the multi-stage queuing system model with the SN policy is shown in Fig. 7. The steady-state probability distribution of a non-empty system (i, j, n) is denoted by P^n_{ij} and the following notations are used:

P^0_{00} ≡ Probability that there has no job in the system and no stage begins service when a server is in a sleep mode;

P^0_{ik} ≡ Probability that there have i jobs in the system and a job is processed at the stage k when a server is in a sleep mode, where i=1, 2, …, N–1;

P^1_{ij} ≡ Probability that there have i jobs in the system and a job is processed at the stage j when a server is in a busy mode, where i=1, 2, …, and j=1, 2, …, k.

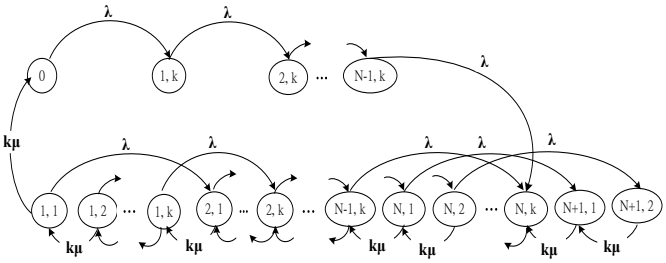


Fig. 7. The state-transition-rate diagram of a multi-stage queuing model with the SN policy.

Based on Fig. 7, the following balanced equations can be given as follows.

$$P^0_{ij}=\begin{cases} \lambda P^0_{00}=\lambda P^0_{1k}, \\ k\mu P^1_{11}=\lambda P^0_{00} \\ \lambda P^0_{ik}=\lambda P^1_{i-1,k}, \quad 2\le i\le N-1 \end{cases} \tag{9}$$

$$P^1_{ij}=\begin{cases} (\lambda+k\mu)P^1_{1j}=k\mu P^1_{1,j+1} \quad 1\le j\le k \\ (\lambda+k\mu)P^1_{1k}=k\mu P^1_{21}, \\ (\lambda+k\mu)P^1_{ij}=\lambda P^1_{i-1,j}+k\mu P^1_{i,j+1}, \quad 2\le i\le N, 1\le j\le k-1 \\ (\lambda+k\mu)P^1_{ik}=\lambda P^1_{i-1,j}+k\mu P^1_{i+1,1} \quad 2\le i\le N-1 \\ (\lambda+k\mu)P^1_{Nk}=\lambda P^1_{N-1,k}+k\mu P^1_{N+1,1}+\lambda P^0_{N-1,k}, \\ (\lambda+k\mu)P^1_{ij}=\lambda P^1_{i-1,k}+k\mu P^1_{i,j+1} \quad i\ge N+1,\ 1\le j\le k-1 \\ (\lambda+k\mu)P^1_{ik}=\lambda P^1_{i-1,k}+k\mu P^1_{i+1,1} \quad i\ge N+1. \end{cases} \tag{10}$$

The solution of P^0_{00} and P^0 can be computed from Eq. 9 by applying the probability generating function (p,g,f) technique, the normalizing equation $P^0_{00}+\sum_{i=1}^{N-1}P^0_{ik}+\sum_{i=1}^{\infty}\sum_{j=1}^{k}P^1_{ik}=1$ and the L'Hopital's Rule. Then, the solutions of the busy and idle probabilities can be given as follows.

$$\begin{cases} P_I=P^0=1-\rho \\ P_B=\rho \end{cases} \tag{11}$$

Let L_s and L_b denote the mean number of jobs in the system when a server is in a sleep and a busy mode, respectively. By applying the L'Hopital's Rule, the mean number of jobs can be obtained as follows [26].

$$L=L_s+L_b=\sum_{i=1}^{N-1}iP^0_{ik}+\sum_{i=1}^{\infty}\sum_{j=1}^{k}iP^1_{ij}. \tag{12}$$

Finally, the SI policy is applied to a queuing model that has Poisson arrivals and general service-time distributions for comparison. It's assumed that the sleep duration follows an exponential distribution with a parameter Θ.

Solution derivation of an M/G/1 queuing model has been studied extensively in previous works since 1975 [27], hence, it's omitted here since it's not our research topic [28], [29], [30].

4.2 Optimization Problem Formulation

In general, a larger controlled N value can gain more power saving but result in excessive delay. Conversely, a smaller controlled N value can reduce delay times but lead to a shorter operational cycle. Therefore, the power consumption overhead due to server startup cannot be ignored. The operational costs and system congestion cost considered in our cost function include power consumption (service rates, operating modes and server startup) and performance degradation (congestion management cost and delay cost). The corresponding cost notations are defined and listed in Table 1.

Table1.
Cost Notations

Notation	Description
C_0	Power consumption cost when a server is in a busy mode per unit time;
C_1	Power consumption cost when a server is in an idle mode per unit time;
C_2	Power consumption cost per service rate per unit time;
C_3	Power consumption cost when a server is in a sleep mode per unit time;
C_4	Server startup cost incurred by activating a server;
C_5	Cost incurred by jobs waiting in a system per unit time;
C_6	Cost incurred by congestion management cost per unit time;

The waiting time cost C_5 mainly indicates the performance penalty cost that is used to compensate for user delay experienced. On the other hand, the congestion management cost (also known as the holding cost in a queuing system) is spent to manage arrival jobs according to a service discipline (e.g., FCFS, LCFS) and avoid a waiting queue growing without bound. Besides, the mean length of operational period (can be found in [25], [26], [30]), denoted by E[C], is also considered in our cost function to estimate startup cost. Since system performances and operational cost strongly depend on the service rate and mode-switching restriction, a cost objective function per unit time is developed in which both the service rate and the controlled N value are the main decision variables to address a tradeoff problem.

Furthermore, it is known that a response time guarantee is regarded as one of the most important performance concerns in designing a green cloud system since no customer wants to suffer from long delay caused by power conservation. Therefore, the SLA constraint is focused on the response time guarantee, which indicates the time from a job arrival in a buffer to the time that a job request has been processed and completed. That is, both the waiting time in the queue and the job execution time are considered. The cost minimization problem can be stated mathematically as:

$$\text{Minimize } Fc$$

$$\text{Where } Fc = Fc(\mu, N)$$
$$= C_0P_B + C_1P_I + C_2\mu + C_3P_S + C_4/E[C] + C_5W + C_6L \qquad (13)$$

Subject to

$$0 \le \rho \le 1 \qquad \text{(i)}$$
$$W \le x \qquad \text{(ii)}$$

The differences in operating procedures, service times and idle time distributions between different polices are listed in table 2.

Table 2
Comparing Different policies

Policies / Descriptions	ISN-1	ISN-2	SN	SI
Differences in operating procedures				
A server goes into a sleep mode immediately when a server becomes empty			√	√
Switching into a busy mode depends on the number of jobs in a queue	√	√	√	
An idle server is not allowed in a system			√	
Applying the mode-switching control	√	√	√	
Distributions of service times				
Having exponential service times	√	√		
Having Erlang-k service times			√	
Having general service times				√
Distributions of idle times				
Having exponential distributions	√		-	√
Having deterministic (constant) idle times		√		

4.3 Performance Comparisons and ECG control Algorithm

To gain more insight into systems with different power-saving policies, experiments are conducted to (i) illustrate

the relationship between the mode-switching restriction and traffic-load intensity on power consumption cost and system congestion cost; (ii) examine the idle and sleep probability distributions under different service rates and (iii) compare response times and total operational costs with a typical system, where it doesn't have any energy-efficient control in operating.

For an idle server in a data center, power waste is compounded by not only the server itself, but also the power distribution losses and air conditioning power usage, which increase power consumption requirements by 50-100% [31]. Therefore, an idle cost parameter is setted within the reference range and the cost matrix is assumed as [C_0, C_1, C_2, C_3, C_4, C_5, C_6] = [500, 400, 1, 8, 2, 3, 3] in simulations. All computational programs are coded by using MATLAB. The traffic-load intensities are assigned values of 0.5 and 0.8 in experiments. These values give reasonable insight into the situations during an off-season and a peak-load period, which will result in relatively low and high traffic intensities, respectively for a cloud provider, such as Amazon.

Fig. 8 shows the power consumption cost for systems with the ISN-1 and the ISN-2 policies under different N values, which is made variable from 1 to 40, while the arrival rate is assumed as λ=960 request/min and $\Theta_i = \Theta_d = \Theta_s = \Theta = 8$. As can be seen, power consumption costs decrease as the N value increases but the costs drop slowly as the N value further increases. The cost differences between systems with both policies become virtually undetectable when the N value is large. Conversely, system congestion costs are roughly proportional to the N value. The operational costs incurred in a system with the ISN-1 policy are higher than the ISN-2 policy regardless of whether there has a lower or higher traffic intensity, as shown in Fig. 9.

In additional to the N value, the service rate also has a significant influence on the operational cost and performances. Fig. 10 demonstrates the idle probability distribution as compared to a system without applying any power-saving policy under different service rates, while the arrival rate is the same as used in Fig. 8. The μ value is made variable from 18 to 50, while the N=5. As can be seen, reductions in idle probability between systems with different power-saving policies are not the same. Most of their idle times can be reduced or eliminated by switching into sleep modes for systems with the power-saving policies, as shown in Fig. 11. Comparison of results show that although adopting the ISN-1 policy has a larger probability for a server to stay in a sleep mode under varying service rates, it causes higher power consumption than adopting the ISN-2 policy under either higher or lower traffic intensity when the N value is low.

The response time distribution is shown in Fig. 12, as could be expected, systems with power-saving policies will result in higher response time. One may also notice that all of them decrease and stabilize as the service rate further increase. However, total operational cost rapidly increases as the service rate increases for a system without applying power-saving policies, as shown in Fig. 13. From observation, it can be noted that the differences in cost reduction are slight between these power-saving policies when the gap between the startup cost and the system congestion cost is small.

In the following, larger differences between the startup cost and the system congestion cost are studied and compared between systems with different power-saving policies. Two situations where one has a larger system congestion cost with a lower startup cost (assumed as [C_4, C_5, C_6] = [2, 20, 20]) and the other has the converse situation (assumed as [C_4, C_5, C_6] = [40, 2, 2]) are illustrated, as shown in Fig. 14 (a) and (b). As can be seen, when startup cost > system congestion cost (UC > CC), it will lead to higher costs under a larger service rate; conversely, lowering service rate results in a higher cost when UC < CC (see Fig. 14 (b)).

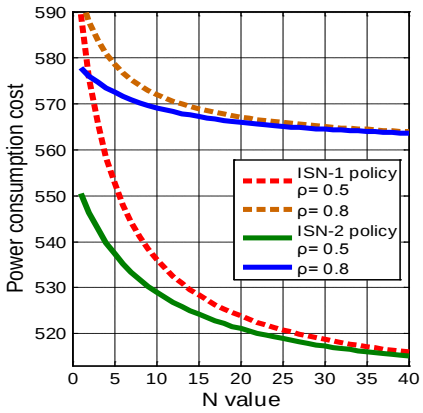


Fig. 8. Power consumption cost under various N values.

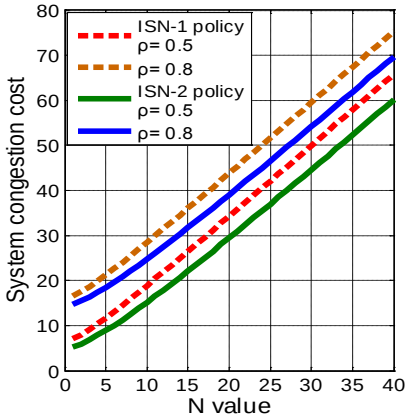


Fig. 9. System congestion cost under various N values.

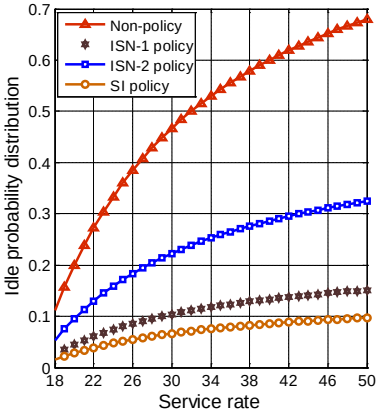


Fig.10. Idle probability distribution under various service rates.

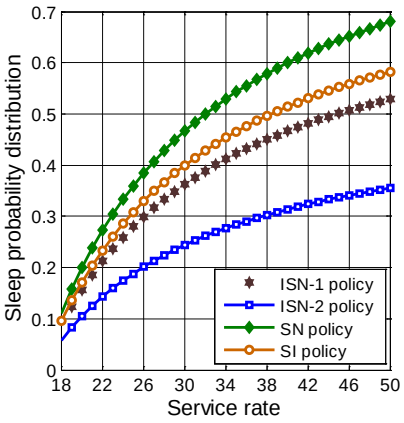


Fig. 11. Sleep probability distribution under various N values.

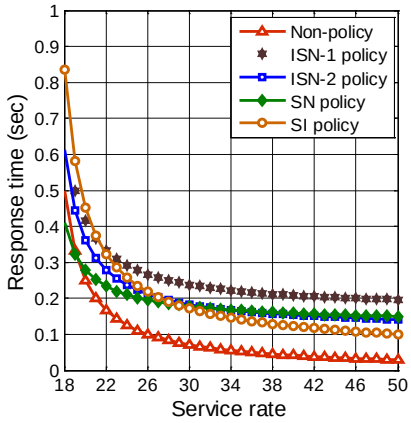


Fig. 12. Response time under various various service rates.

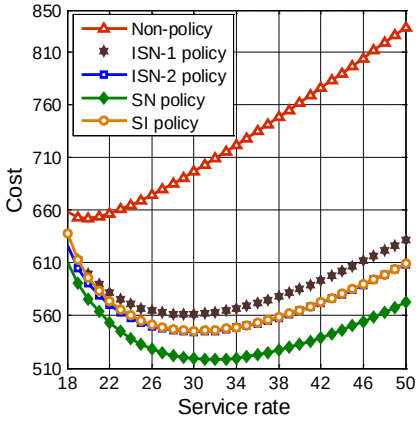


Fig. 13. Cost distribution under various service rates.

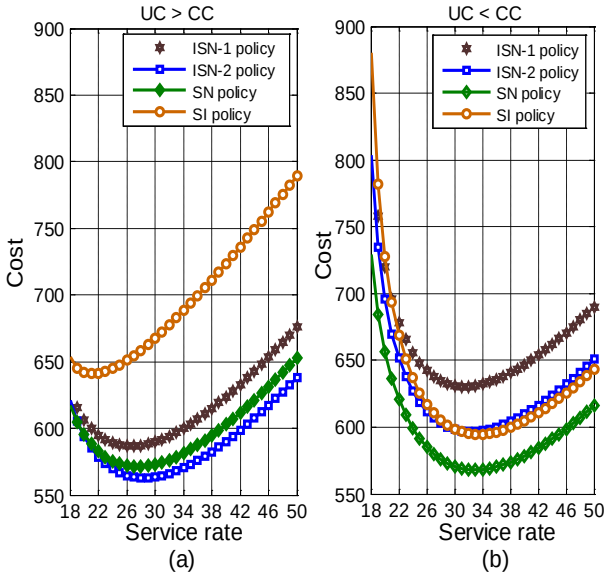


Fig. 14. Cost distributions under (a) UC>CC situation (b) UC<CC situation.

Basically, it isn't worthwhile for a server to be activated from the sleep mode immediately upon a job arrival if UC > CC. As can be seen, lower cost can be obtained when UC < CC for a system with the SI policy; however, it will incur the highest cost when UC > CC. This behavior is due to the fact that, mode-switching frequency for a system with the SI policy is higher than other power-saving policies; thus, it will lead to a higher startup cost. However, the SI policy ensures that a job can get service without long delay, hence, it can achieve the lowest cost than others when a system has a lower startup cost (see Fig. 14 (b)).

Results show that to choose and implement the most suitable power-saving policy among diverse approaches, a cloud provider should take not only the operating mode configurations, but also the decision processes and incurred costs into consideration since these have non-negligible impacts on system performances and cost distributions. Here, our goal is to optimize an operational cost and of course, obey a response time guarantee via the optimal service rate and the controlled N value, say (μ^*, N^*). However, it is extremely difficult to obtain the minimum cost and the analytical results of the optimal solution due to the fact that the objective function is complex and highly nonlinear under either the μ or N value; besides, there has two constraints that need to be satisfied (i) $0 \le \rho \le 1$ and (ii) $W \le x$. Instead, an EGC algorithm is presented to solve the nonlinear constrained optimization problem effectively. Meeting a SLA constraint has the highest priority, followed by a cost minimization in deciding the optimal solution (μ^*, N^*).

EGC Algorithm

Input:
1. An arrival rate λ.
2. Upper bound of the server rate and the waiting buffer, denoted by μ_u and N_b
3. Cost parameters [C_o, C_1, …, C_6]
4. A response time guarantee x.
5. System parameters {Θ_i, Θ_d, Θ_s} used by the ISN policy
6. System parameter {k} used by the SN policy
7. System parameters {Θ, N=1} used by the SI policy

Output: μ^*, N^* and F_c (μ^*, N^*)

Step1. For i = 1; i = u; i++
  Set $\mu_i \leftarrow$ a current service rate;
Step 2. For j = 1; j = b; j++
  Set $N_j \leftarrow$ a current N parameter;
Step 3. Calculate the system utilization.
  If the current test parameters satisfy the constraint of (i) $0 \le \rho \le 1$, then
    Calculate the response time;
  Else
    Return to step 1 and begin to test a next index;
  End
Step 4. If the current test parameters satisfy the constraint of (ii) $W \le x$, then
    Record the current joint values of (μ_i, N_j) and identify it as the approved joint parameters;
  Else
    Return to step 1 and begin to test a next index;

End

Step5. When all the test parameters have been done, then
{(μ_{i+a}, N_{j+a}),…, (μ_u, N_b)} ← current set of the approved parameters;
Bring cost parameters into the objective function using Eq. 13 and test all approved joint parameters;

Step6. If the joint values of (μ_k, N_k) can obtain the minimum cost value in all testing, then,
Output (μ_k, N_k) and $F_c(\mu_k, N_k)$
Else
Return to step 5 and begin to test a next approved parameter.
End

5 Experimental Results

5.1 Numerical Validation

Experiments are conducted to (i) validate that the optimal solution to minimize cost within a response time constraint can be obtained by applying the power-saving policies with the EGC algorithm. (ii) Cost reduction and response time improvement can be achieved at the optimal solution as compared to a general policy. Systems with different power-saving policies need to comply with the same response time guarantee, which is assumed to be within 0.5 second in the SLA constraint, denoted by SLA ($W \leqq 0.5$). Fig. 15, Fig. 16, and Fig. 17 demonstrate the cost distribution for systems with the ISN-1, the ISN-2 and the SN policies, respectively by assuming the λ=1200/min and using the same cost and system parameters in Fig. 13.

As can be seen, the optimal solutions solved by the EGC algorithm can be obtained to optimize cost in systems with different power-saving policies. A system with the SN policy can obtain a lower cost than the ISN-1 and the ISN-2 policies. The values are 584.86, 571.62 and 548.15 at the optimal solution of (μ, N) = (32, 12), (32, 8) and (33, 3) for the ISN-1, the ISN-2, the SN policies, respectively. The corresponding response times to satisfy the SLA ($W \leq 0.5$) are 0.4016, 0.2228 and 0.1048, respectively, as shown in Fig. 18, Fig. 19 and Fig. 20. It's known that both the service rate and the N value have non-negligible impact on the cost distribution (see Fig. 15, Fig.16, and Fig.17), but the effect of the N value on the response time is more obvious than the service rate, especially for the SN policy.

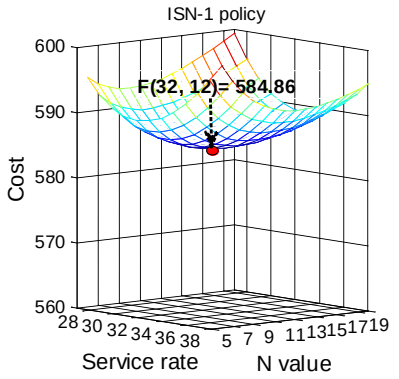


Fig. 15. Cost distribution for the system with the ISN-1 policy.

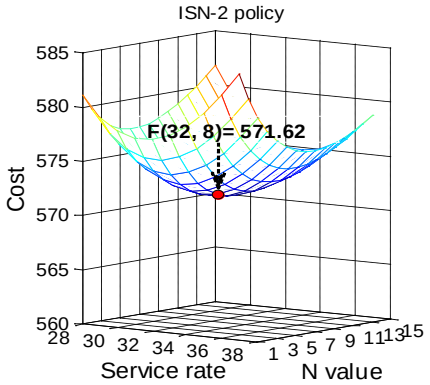


Fig. 16. Cost distribution for the system with the ISN-2 policy.

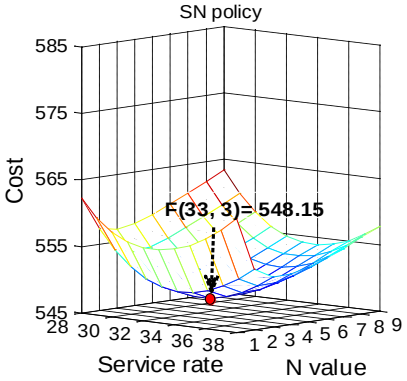


Fig. 17. Cost distribution for the system with the SN policy.

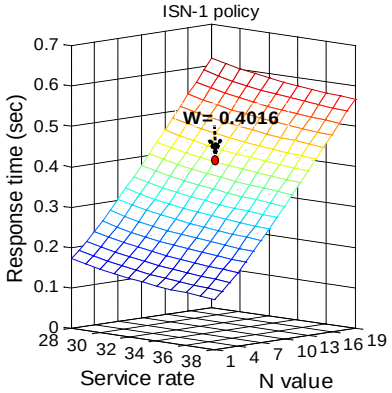


Fig. 18. Response time distribution for the system with the ISN-1 policy.

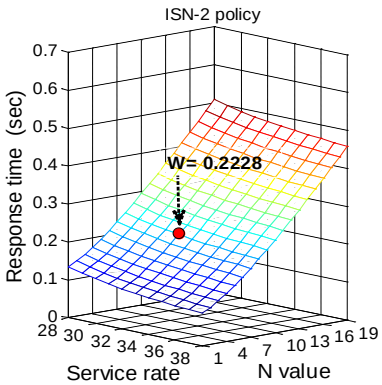


Fig. 19. Response time distribution for the system with the ISN-2 policy.

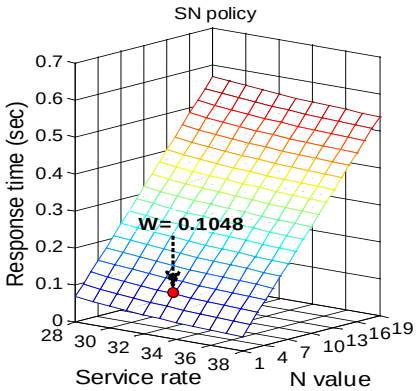


Fig. 20. Response time distribution for the system with the SN policy.

5.2 Comparison of Results

The proposed power-saving policies combined with the EGC algorithm are evaluated on the basis of comparisons with a general policy. For a general policy, it implies that a solution is given only by considering an absolute performance guarantee (in most existing approaches [32, 33, 34] to manage power consumption) in which no energy-efficient control or cost evaluation is analyzed. Different arrival rates ranging from 360 to 1200 request/min are demonstrated in order to investigate a wide range of traffic load intensities from an off-season to a peak-load period. In order to compare the power-saving policies with a general policy, we mainly control the service rate in systems to obey the same performance guarantee of SLA(W $\leqq$ 0.5) under various arrival rates with the fixed N value of 5.

Comparisons of the idle probability in a system with the general policy and sleep probabilities in systems with the power-saving policies are shown in Fig. 21. Sleep probabilities and the idle probability will be reduced since a server has more probability to work and stay in a busy mode when the arrival rate increases. It is also noted that sleep probabilities are obvious larger than the idle probability in a system with the general policy due to the fact that our proposed algorithm tries to enhance process efficiency in busy modes to satisfy the response time guarantee and reduce system delay cost.

In other words, service rates in systems with power-saving policies are controlled at higher values and their idle times can be reduced by switching into sleep modes. Conversely, the general policy focuses only on a performance guarantee and reduces the service rate as low as possible for the purpose of saving operational cost. On the other hand, since a server switches into a sleep mode only once in an operation cycle for the SI policy, the variation among sleep probabilities is slight as the arrival rate increases.

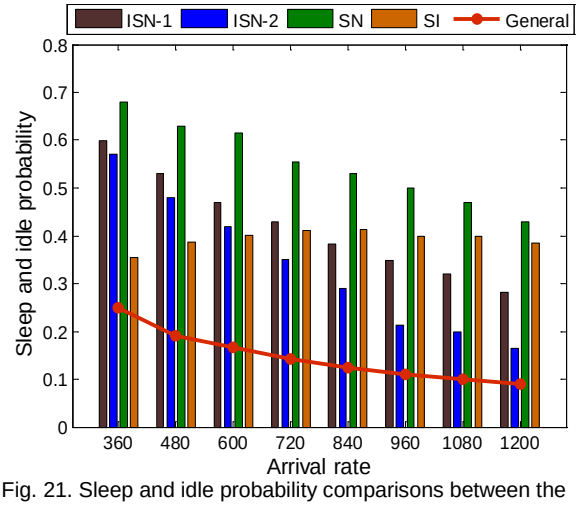


Fig. 21. Sleep and idle probability comparisons between the power-saving policies and the general policy.

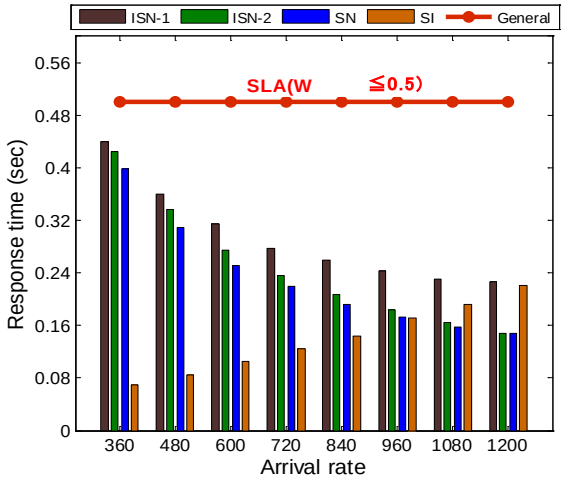


Fig. 22. Response time comparisons between power-saving policies and the general policy.

Fig. 22 shows the response time under various arrival rates. As can be seen, the response time variations are affected by the probability that a server stays in the sleep mode. The system with the SI policy doesn't reduce the sleep probability as arrival rate increase; hence, it results in higher response time than other power-saving policies with the higher arrival rate. For the general policy, the response times are all kept at the 0.5 sec guarantee. Although the general policy tries to keep the service rate as low as possible, it still results in higher cost than other policies, as shown in Fig. 23.

The proposed power-saving policies can effectively reduce cost, especially when the arrival rate is low. Finally, we measure the performance and cost improvement ratios, which calculate the relative value of improvements to the original value instead of an absolute value, the results are shown in Fig. 24 (a) and Fig. 24 (b). Results show that applying the SI policy can get a better response time reduction when the arrival rate is low. Nevertheless, for a cloud provider who focuses on reducing cost, implementing the SN policy is a better choice to deal with a wide range of arrival rates.

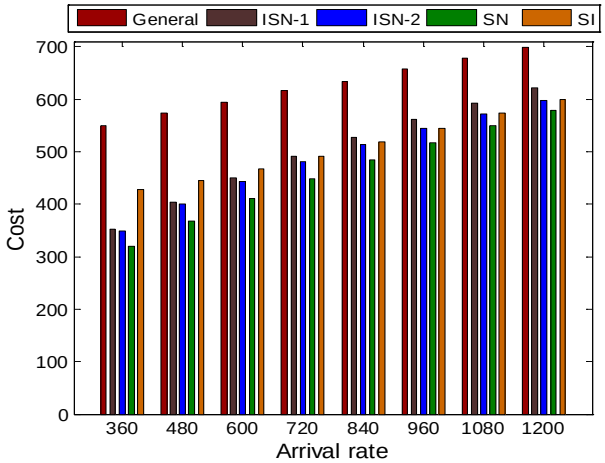


Fig. 23. Cost comparisons between the power-saving policies and the general policy.

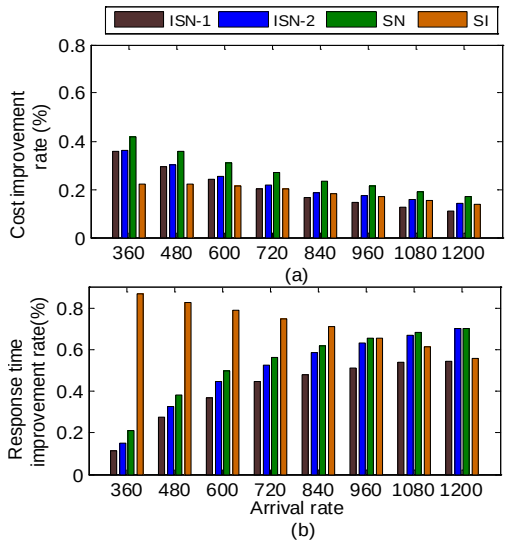


Fig. 24. (a) (b) Cost and response time improvement rates.

6 Conclusion

The growing crisis in power shortages has brought a concern in existing and future cloud system designs. To mitigate or eliminate unnecessary idle power consumption, three power-saving policies with different decision processes and mode-switching controls are considered. The issue of choosing the most suitable policy among diverse power management policies to reach a relatively high effectiveness has been examined based on the variations of arrival rates and incurred costs. Experimental results show that a system with the SI policy can reach a greater cost-effectiveness than other policies when there has a lower startup cost. It also can significantly improve the response time in a low arrival rate situation. On the other hand, applying others policies can obtain more benefits in the converse situation.

Our proposed algorithm allows cloud providers to optimize the decision-making in service rate and mode-switching restriction, so as to minimize the operational cost without sacrificing a SLA constraint. As compared to a general policy, the benefits of cost savings and response time improvement can be verified. In future, we plan to analyze more key factors that will influence the power consumption and system performances in cloud environments. Looking into finite user populations, traffic load controls, etc. will be another direction of extension to achieve energy-efficiency through a comprehensive system control.

References

[1] G. Wang, and T. E. Ng, "The impact of virtualization on network performance of amazon ec2 data center," IEEE Proceedings in INFOCOM, pp.1-9, 2010.

[2] R. Ranjan, et al, "Peer-to-peer cloud provisioning: Service discovery and load-balancing," Springer London in Cloud Computing, pp. 195-217, 2010.

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

[4] Server virtualization has stalled, despite the hype, at http://www.infoworld.com/print/146901

[5] Y. C. Lee, and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," The Journal of Supercomputing, vol. 60, no. 2, pp. 268-280, 2012.

[6] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," Advances in Computers, vol. 82, pp. 47–111, 2011.

[7] R. Buyya, et al. "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation computer systems, vol.25, no. 6, pp. 599-616, 2009.

[8] L. Wang, et al. "Cloud computing: a perspective study," New Generation Computing, vol. 28, no. 2, pp. 137-146, 2010.

[9] R. Ranjan, R. Buyya, and M. Parashar, "Special section on autonomic cloud computing: technologies, services, and applications," Concurrency and Computation: Practice and Experience, vol. 24, no. 9, pp. 935-937, 2012.

[10] M. Yadin, M, and P. Naor, "Queueing systems with a removable service station," Operations research quarterly 14, pp. 393-405, 1963.

[11] W. Huang, X. Li and Z. Qian, "An Energy Efficient Virtual Machine Placement Algorithm with Balanced Resource Utilization," Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp. 313-319, 2013.

[12] Nathuji R, et al., "VPM tokens: virtual machine-aware power budgeting in datacenters," Cluster computing, vol. 12, no. 2, pp.189-203, 2009.

[13] J. S. Yang, P. Liu, and J. J. Wu, "Workload characteristics-aware virtual machine consolidation algorithms," IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 42-49, 2012.

[14] K. Ye, et al, "Virtual machine based energy-efficient data center architecture for cloud computing: a performance perspective," IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, pp.171-178, 2010.

[15] G. P. Duggan and P. M. Young, "A Resource Allocation Model for Energy Management Systems," IEEE International Systems Conference (SysCon), pp. 1-3, 2012.

[16] M. Mazzucco, D. Dyachuky, and R. Detersy, "Maximizing Cloud Providers Revenues via Energy Aware Allocation Policies," IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 131-138, 2010.

[17] Q. Zhang, et al, Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud, IEEE Transactions on Cloud Computing, vol. 2, no.1, pp.14-28, 2014.

[18] M. Guazzone, C. Anglano and M. Canonico, "Energy-Efficient Resource Management for Cloud Computing Infrastructures," IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 424-431, 2011.

[19] A. Amokrane, et al, "Greenhead: Virtual Data Center Embedding across Distributed Infrastructures," IEEE Transactions on Cloud Computing, vol. 1, no.1, pp. 36-49, 2013.

[20] F. Larumbe, and B. Sanso, "A Tabu Search Algorithm for the Location of Data Centers and Software Components in Green Cloud Computing Networks," IEEE Transactions on Cloud Computing, vol. 1, no. 1, pp. 22-35, 2013.

[21] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," Springer Berlin Heidelberg in Algorithms and architectures for parallel processing, pp.13-31, 2010.

[22] P. Leitner, W. Hummer, and S. Dustdar, "Cost-based optimization of service compositions," IEEE Transactions on Services Computing, vol. 6, no. 2, pp. 239-251, 2013.

[23] A. C. Zhou, and B. He, "Transformation-based monetary cost optimizations for workflows in the cloud," IEEE Transactions on Cloud Computing, pp. 1-1, 2013.

[24] H. Goudarzi, M. Ghasemazar, and M. Pedram, "Sla-based optimization of power and migration cost in cloud computing," IEEE/ACM International Symposium on Cloud and Grid Computing (CCGrid) in Cluster, pp. 172-179, 2012.

[25] Y. Deng, W. J. Braun, and Y. Q. Zhao, "M/M/1 queueing system with delayed controlled vacation," OR Transactions, vol. 3, pp. 17-30, 1999.

[26] K. H. Wang and H. M. Huang, "Optimal control of an M/E_k/1 queueing system with a removable service station," Journal of the operational research society," vol. 46, pp.1014 -1022, 1995.

[27] Y. Levy and U. Yechiali , "Utilization of Idle Time in an M/G/1 Queueing System," Management Science, vol. 22, no2, pp.202-211, 1975.

[28] T. Naishuo, Z. Daqing, C. Chengxuan, "M/G/1 queue with controllable vacations and optimization of vacation policy," Acta Mathematicae Applicatae Sinica, pp. 363-373, vol. 7, mo. 4, 1991.

[29] D. A. Wu, D. A., and H. Takagi, "M/G/1 queue with multiple working vacations," Performance Evaluation," vol. 63, no. 7, 654-681, 2006.

[30] M. Zhang, and Z. Hou, "M/G/1 queue with single working vacation," Journal of Applied Mathematics and Computing, vol. 39, no. 1-2, 221-234, 2012.

[31] D. Meisner, B. T. Gold and T. F. Wenisch, "PowerNap: Eliminating Server Idle Power," In proceedings of the 14th international conference on Architectural support for programming languages and operating systems, 205-216, 2009.

[32] H. Mi, et al, "Online Self-reconfiguration with Performance Guarantee for Energy-efficient Large-scale Cloud Computing Data Centers," IEEE International Conference on Services Computing, pp. 514-521, 2010.

[33] H. AbdelSalam, et al, "Towards Energy Efficient Change Management in a Cloud Computing Environment," Springer Berlin Heidelberg Scalability of Networks and Services, pp. 161-166, 2009.

[34] J. Song, et al, "Study on energy-consumption regularities of cloud computing systems by a novel evaluation model," pp. 1-19, 2013.