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 (Robert) Hsu, *Senior Member, IEEE*

Abstract—Cloud computing is a new paradigm for 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 costs of power consumption, system congestion and server startup are all taken into consideration. The effect of energy-efficiency controls on response times, operating modes and incurred costs are all demonstrated. 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 under 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 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. 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, resource over-provisioning [3] is a common situation in a cloud system. However, most electricity-dependent facilities will inevitably suffer from idle times or low utilization 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 percent of the time on average [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 low-load 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. A penalty is given when a cloud provider violates performance guarantees in a SLA contract. 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 [9].

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

- *Y.-J. Chiang and Y.-C. Ouyang are with Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan. E-mail: yjchiang0320@gmail.com, ycouyang@nchu.edu.tw.*
- *C.-H. Hsu is with the Department of Computer Science and Information Engineering, Chung Hua University, Hsinchu 300, Taiwan. E-mail: chh@chu.edu.tw.*

Manuscript received 14 Mar. 2014; revised 18 June 2014; accepted 16 July 2014. Date of publication 20 Aug. 2014; date of current version 10 June 2015. Recommended for acceptance by R. Ranjan, L. Wang, A. Zomaya, D. Georgakopoulos, G. Wang, and X.-H. Sun.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCC.2014.2350492

- 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 in cloud systems. Systems with different 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 (VM) side by migrating VMs, applying consolidation or allocation algorithms, and on the data center infrastructure side through resource allocations, energy managements, etc.

2.1 Power-Saving in Virtual Machine

In [11], Huang et al. 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.

Yang et al. proposed an approximation algorithm (SEP-Pack) and two dynamic programming to consolidate virtual machines [13]. Two issues in energy conservation algorithm were addressed—the placement of virtual machine image and the characteristics of virtual machines. Despite that the dynamic programming could find the optimal solution, its time complexity was prohibitive in practice. In [14], the energy efficiency from the performance perspective was studied. 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 et al. 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. Unlike previous studies, our paper contributes to investigate an essential tradeoff between power consumption costs and system performances by applying different power-saving policies. 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.

3 Power Management in Clouds

3.1 ISN Policy

Briefly speaking, a distributed service system consists of lots of physical servers, virtual machines and a job dispatcher. The job dispatcher in our designed system is used to identify an arrival job request and forward it to a corresponding VM manager that can meet its specific requirements. When there has no job in a waiting queue or no job is being processed, a server becomes idle and it remains until a subsequent job has arrived. 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 modes 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. A server is allowed to stay in an idle mode for a short time when there has no job in the system, rather than switch abruptly into a sleep mode right away when the system becomes empty [21]. An idle mode is the only operating mode that connects to a sleep mode. A server doesn't end its sleep mode even if a job has arrived; it begins to work only when the number of jobs in a queue is more than the controlled N value. 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". Fig. 1 illustrates the step-by-step decision processes and job flows of the ISN policy.

Step 1. A server ends its busy mode when all current job requests have been finished.

Step 2. A server stays in an idle mode and waits for subsequently arriving jobs before switching into a sleep mode.

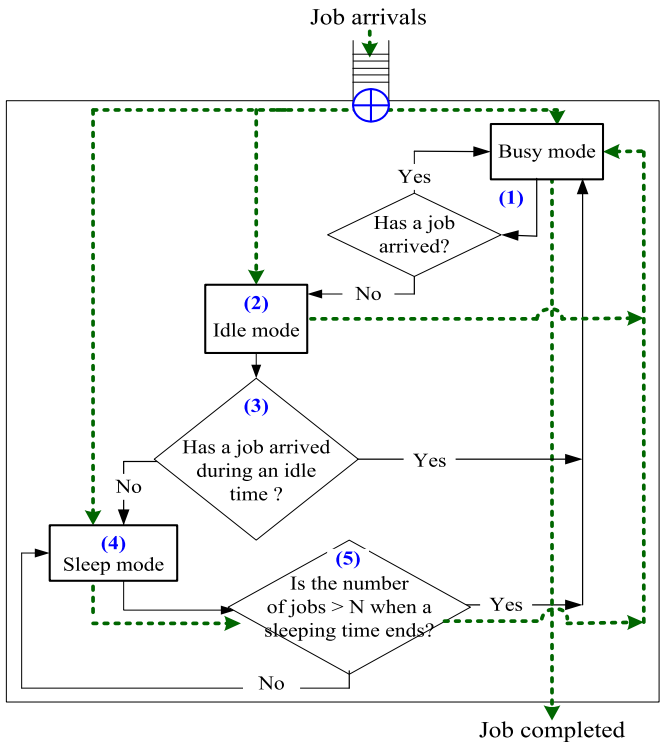

Fig. 1. Decision processes of the ISN policy.

Step 3. If a job arrives during an idle period, a server can switch into a busy mode and start to work immediately. A server begins a next idle period until all job requests have been successfully completed.

Step 4. If there has no job arrival, a server switches into a sleep mode when an idle period expires.

Step 5. A server remains in a sleep mode if the number of jobs in the queue is fewer than the controlled N value. Otherwise, a server switches into a busy mode and begins to work.

Basically, there have 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 a waiting queue is more than the N value when a sleep period expires.

Although power is wasted in allowing a server to stay in the idle mode during a non-load period, the benefits are that an arrival job has more possibilities to get immediately service and the server startup cost can be reduced.

3.2 SN and SI Policies

To greatly reduce idle power consumption, non-idle mode operating is considered in another approach, where it only holds {Busy, Sleep} operating modes. Instead of entering into an idle mode, a server immediately switches into a sleep mode when the system becomes empty. Similarly, a server switches into a busy mode depending on the number of jobs in a waiting queue to avoid switching too often. 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. 2 illustrates the step-by-step decision processes and job flows of the SN policy.

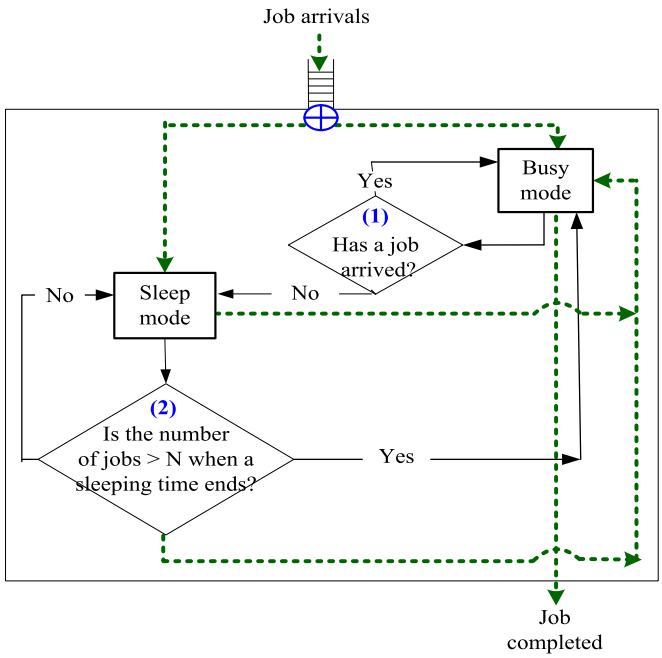

Fig. 2. Decision processes of the SN policy.

Step 1. A server switches into a sleep mode immediately when no job is in the system.

Step 2. 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. A server switches into a sleep mode right away rather than an idle mode when there has no job in the system. This is similar to the SN policy but a server only stays in a sleep mode for a given time. When a sleeping time expires, it will enter into an idle mode or a busy mode depending upon whether a job has arrived or not. According to the switching process (from **S**leep to **I**dle), we have called such an approach "SI policy". Fig. 3 illustrates the step-by-step decision processes and job flows of the SI policy.

Step 1. A server switches into a sleep mode immediately 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 arrival 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.

4 Objective Function

4.1 Queuing Models

Systems applied with these 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 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.

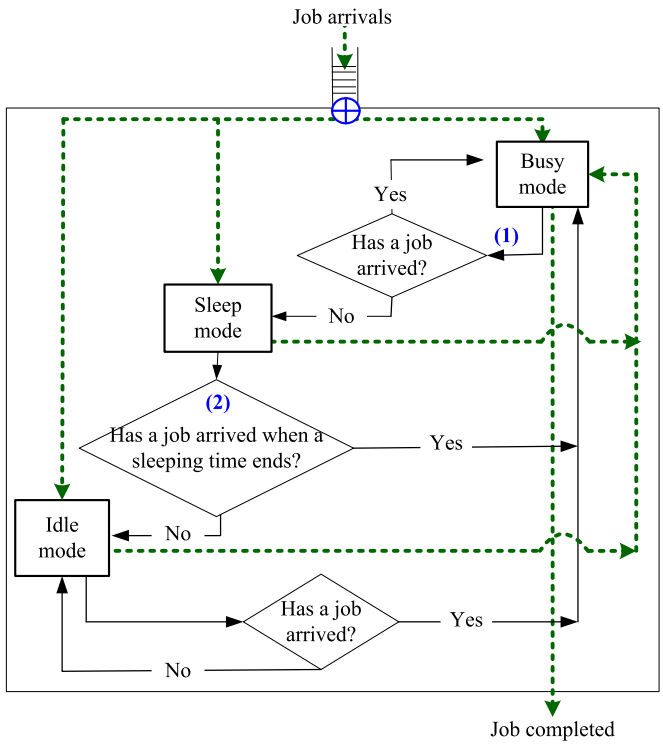

Fig. 3. Decision processes of the SI policy.

Idle times that these 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 $S = \{(n, m), 0 \leq n < \infty, m = \{0, 1, 2\}\}$ where n denotes the number of jobs in the system, and m denotes the operating mode. The state-transition-rate diagram for a queuing system with the ISN-1 policy is shown in Fig. 4. 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.

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

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

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

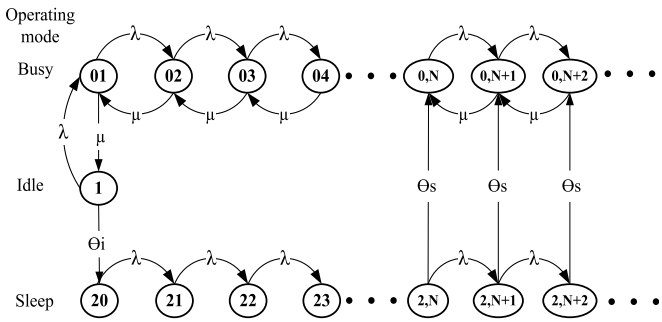

Fig. 4. The state-transition-rate diagram for a queuing model with the ISN policy.

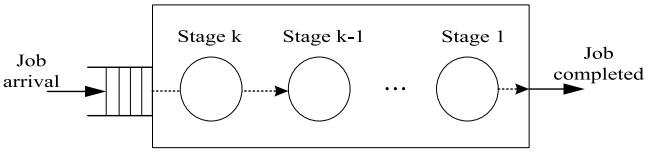

Fig. 5. The Erlang-k service model.

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

Based on Fig. 4, the following balanced equations can be given:

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

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

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

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 [21] can be obtained as follows:

$$\begin{cases} P_B = \sum_{n=1}^{\infty} P_{0n} \\ P_I = P_1 = \frac{\lambda\theta_s(1-\rho)}{\lambda\theta_i + \lambda\theta_s + K\theta_d\theta_s}, \\ P_S = \sum_{n=0}^{\infty} P_{2n}. \end{cases} \quad (4)$$

Similarity, by using the balanced equations in a queuing system with the ISN-2 policy and 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 [21].

$$\begin{cases} P_B = \sum_{n=1}^{\infty} p_{0n} = \sum_{n=1}^{\infty} p_1 \left(\frac{\lambda}{\mu}\right)^n + \sum_{n=1}^{\infty} \sum_{i=0}^{n-1} \left(\frac{\lambda}{\mu}\right)^{n-i} p_{2i,} \\ P_I = \frac{(1-\rho)\theta_s(e^{\lambda/\theta_d}-1)}{\lambda-\theta_s+e^{\lambda/\theta_d}\theta_s+N\theta_s}, \\ P_S = \sum_{n=0}^{\infty} p_{2n} = \sum_{n=0}^{K-1} \frac{(1-\rho)\theta_s}{\lambda-\theta_s+e^{\lambda/\theta_d}\theta_s+N\theta_s} + \sum_{n=K}^{\infty} \left(\frac{\lambda}{\lambda+\theta_s}\right)^{j-n+1} p_{0,2}. \end{cases} \quad (5)$$

In the real-world, observed in an application might not exactly be processed by a single service node. There may have 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 in our work. It is assumed that the service times follow an Erlang k-type distribution with mean $1/(k\mu)$ for each phase [22]. 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. 5.

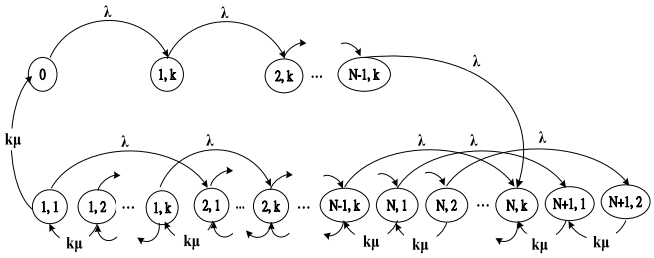

Fig. 6. The state-transition-rate diagram of a multi-stage queuing model with the SN policy.

The state of the system is described by the triplet $(i, j, n), i = 0, 1, 2, \ldots, j = 0, 1, \ldots, k$ and $n = 0, 1$, where i denotes the number of jobs in the system, j denotes the service stage, and n denotes the operating mode. The state transition-rate diagram of the multi-stage queuing system with the SN policy is shown in Fig. 6.

The steady-state probability distribution of a non-empty system (*i, j, n*) is denoted by P_{ij}^n and the notations in the following are used:

$P_{00}^0 \equiv$ Probability that there has no job in the system and no stage begins working when a server is in a sleep mode;

$P_{ik}^0 \equiv$ 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 $i = 1, 2, \ldots, N-1$;

$P_{ij}^1 \equiv$ 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 $i = 1, 2, \ldots,$ *and* $j = 1, 2, \ldots, k$.

Based on Fig. 6, the balanced equations can be obtained and the solutions of P_{00}^0 and the idle probabilities P^0 can be computed by applying the probability generating function (p,g,f) technique, the normalizing equation $P_{00}^0 + \sum_{i=1}^{N-1} P_{ik}^0 + \sum_{i=1}^{\infty} \sum_{j=1}^{k} P_{ik}^1 = 1$ and the L'Hopital's Rule. 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 derivations of an M/G/1 queuing model have been studied extensively in previous works since 1975 [23], hence, it's omitted here since it's not our research topic [24], [25], [26].

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.

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

TABLE 1
Cost Notations

Notation	Description
C_0	Power consumption cost when a server is in a busy mode per unit time;
$C1$	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;
$C3$	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 per unit time;

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 [21], [22], [26]), denoted by E[C], is also considered in our cost function to estimate startup cost. Since system performances and operational costs 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 trade-off 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 by considering both the queuing delay and the job execution time. The cost minimization problem can be stated mathematically as:

$$\begin{aligned} &\text{Minimize } F \\ &\text{Where } F = F(\mu, N) \\ &\quad = C_0 P_B + C_1 P_I + C_2 \mu + C_3 P_S + C_4 / E[C] + C_5 W + C_6 L \end{aligned} \tag{6}$$

Subject to

$$\begin{aligned} 0 \le \rho \le 1 \quad &\text{(i)} \\ W \le x \quad &\text{(ii)}. \end{aligned}$$

The differences in decision processes, 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 decision processes				
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			√	
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 the 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.

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 percent [27]. Therefore, an idle cost parameter is setted within the reference range and the cost matrix is assumed to be $[C_0, C_1, C_2, C_3, C_4, C_5, C_6] = [500, 400, 1, 8, 2, 3, 3]$ in simulations. All computational programs are coded by 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. 7 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 to be $\lambda = 960$ 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 they 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 costs incurred in a system with the ISN-1 policy are higher

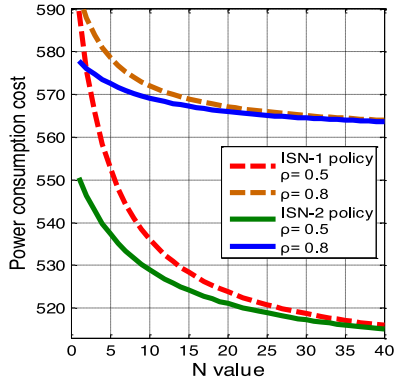

Fig. 7. Power consumption cost under various N values.

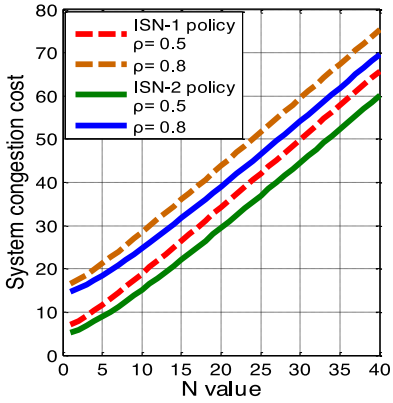

Fig. 8. System congestion cost under various N values.

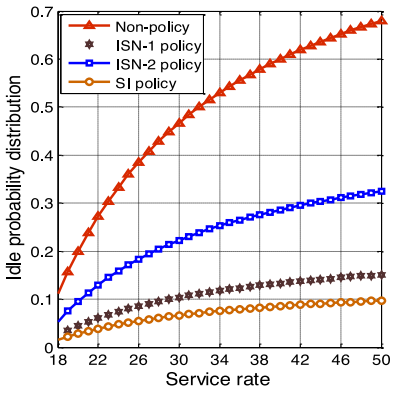

Fig. 9. Idle probability distribution under various service rates.

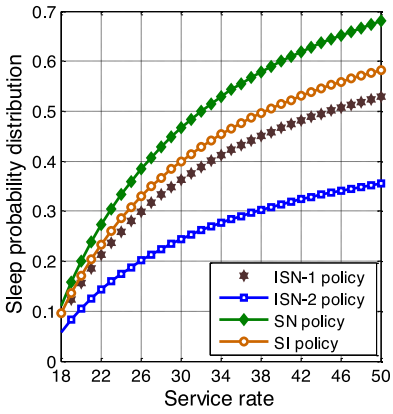

Fig. 10. Sleep probability distribution under various service rates.

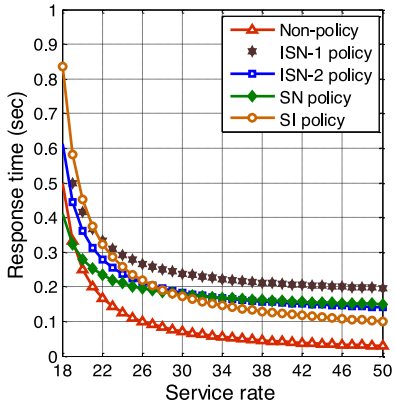

Fig. 11. Response time under various service rates.

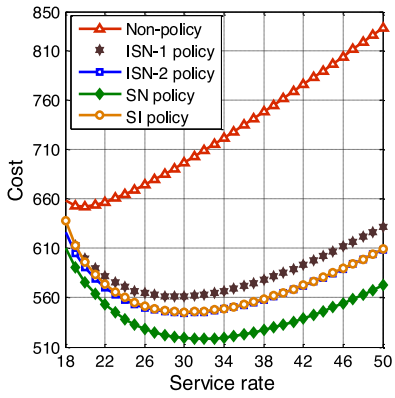

Fig. 12. Cost distribution under various service rates.

than the ISN-2 policy regardless of whether there has a lower or a higher traffic intensity, as shown in Fig. 8.

In additional to the N value, the service rate also has a significant influence on the operational cost and performances. Fig. 9 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. 7. 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. 10. 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 a higher or a lower traffic intensity when the N value is low.

The response time distribution is shown in Fig. 11, as could be expected, systems with power-saving policies will result in a higher response time. One may also notice that all of them decrease as the service rate further increases. However, the operational cost rapidly increases as the service rate increases for a system without applying power-saving policies, as shown in Fig. 12. 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 to be $[C_4, C_5, C_6] = [2, 20, 20]$) and the other has the converse situation (assumed to be $[C_4, C_5, C_6] = [40, 2, 2]$) are illustrated, as shown in Figs. 13a and 13b. As can be seen, when the 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. 13b). Basically, it isn't worthwhile for a server to be activated from the sleep mode immediately when a job request has arrived if $UC > CC$. As can be seen, it will incur

the highest cost for a system with the SI policy when $UC > CC$ due to the fact that its mode-switching frequency is higher than others.

However, the SI policy ensures that a job can get services without long delay, hence, it can achieve a lower cost than others when a system has a lower startup cost (see Fig. 13b). Results show that to choose and implement the most suitable power-saving policy among diverse approaches, a cloud provider should take not only energy-efficient conrtrols but also incurred costs into consideration since these have non-negligible impacts on system performances and operational costs. Here, our goal is to optimize the operational cost and of course, obey a response time guarantee via the optimal service rate and the controlled N value, say (μ^*, N^*). An EGC algorithm is presented to solve the nonlinear constrained optimization problem effectively. Meeting a SLA constraint has the highest priority, followed by 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 $[C_o, C_1, \ldots, C_6]$.
4. A response time guarantee x.
5. System parameters $\{\Theta_i, \Theta_d, \Theta_s\}$ used by the ISN policy.
6. System parameter $\{k\}$ used by the SN policy.
7. System parameters $\{\Theta, N = 1\}$ used by the SI policy.

Output: μ^*, N^* and $F_c(\mu^*, N^*)$

Step 1. 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
Step 5. When all the test parameters have been done, then $\{(\mu_{i+a}, N_{j+a}), \ldots, (\mu_u, N_b)\} \leftarrow$ current set of the approved parameters;
  Bring cost parameters into the objective function by using Eq. (6) and test all approved joint parameters;
Step 6. If the joint values of (μ_{i+a}, N_{j+a}) can obtain the minimum cost value in all testing, then,
    Output (μ_{i+a}, N_{j+a}) and $F(\mu_{i+a}, N_{j+a})$
  Else
    Return to step 5 and begin to test a next approved parameter.
  End

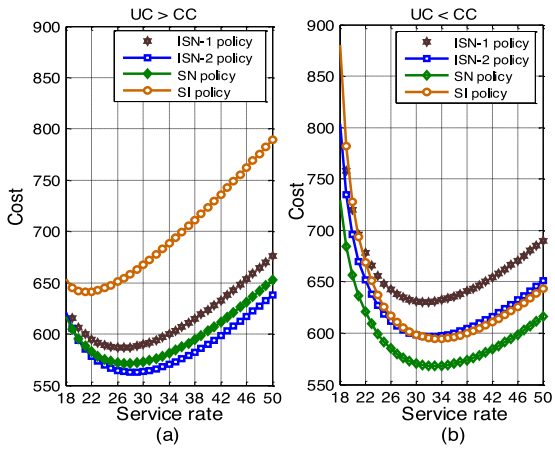

Fig. 13. Cost distributions under (a) $UC > CC$ situation. (b) $UC < CC$ situation.

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 $(W \le 0.5)$. Figs. 14, 15, and 16 demonstrate the cost distribution for systems with the ISN-1, the ISN-2 and the SN policies, respectively by assuming the $\lambda = 1{,}200/\text{min}$ and using the same cost and system parameters in Fig. 12. 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 $(\mu, N) = (32, 12)$, (32, 8) and (33, 3) for the ISN-1, the ISN-2, the SN policies, respectively. The corresponding response times to satisfy the SLA $(W \le 0.5)$ are 0.4016, 0.2228 and 0.1048 sec, respectively, as shown in Figs. 17, 18 and 19. It's known that both the service rate and the N value have non-negligible impact on the cost distribution (see Figs. 14, 15, and 16), but the effect of the N value on the response time is more obvious than the service rate.

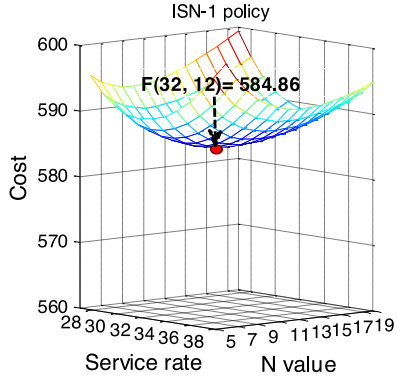

Fig. 14. Cost distribution for the system with the ISN-1 policy.

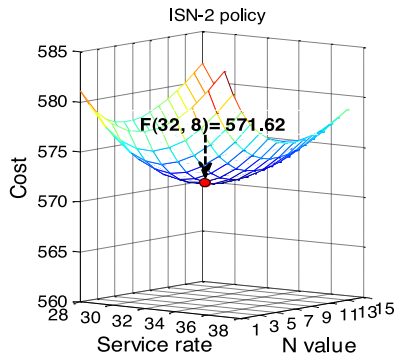

Fig. 15. Cost distribution for the system with the ISN-2 policy.

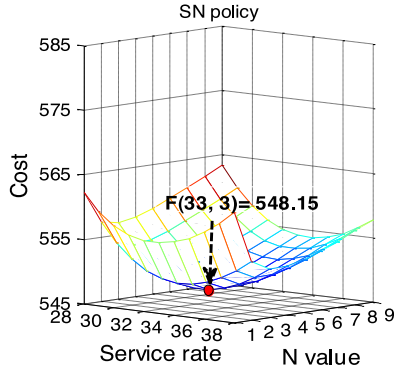

Fig. 16. Cost distribution for the system with the SN policy.

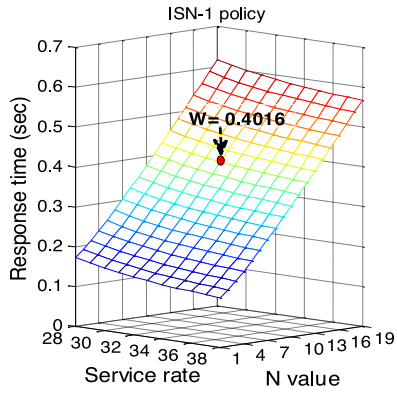

Fig. 17. Response time distribution for the system with the ISN-1 policy.

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 [28], [29], [30]. Different arrival rates ranging from 360 to 1,200 request/min are demonstrated to investigate a wide range of 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 to obey the same performance guarantee of SLA($W \leq 0.5$) 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. 20. The sleep probabilities and the idle probability will be reduced as the arrival rate increases since a server has more probability to work and stay in a busy mode.

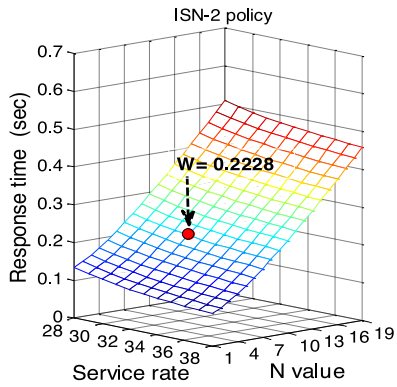

Fig. 18. Response time distribution for the system with the ISN-2 policy.

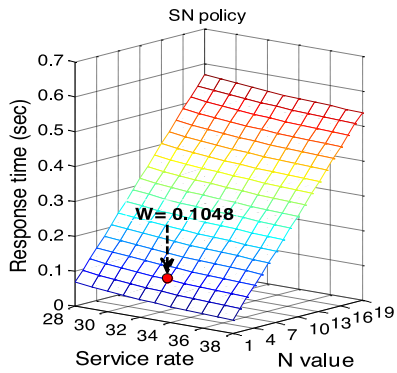

Fig. 19. Response time distribution for the system with the SN policy.

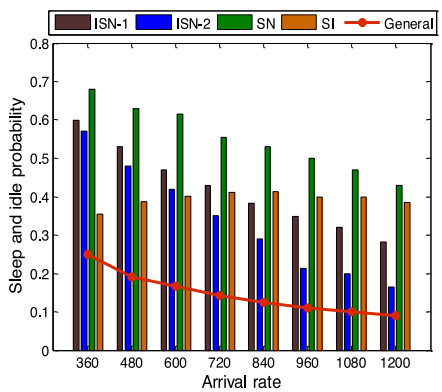

Fig. 20. Sleep and idle probability comparisons.

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 efficiency in busy modes. Therefore, service rates are controlled at higher values with power-saving policies 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 shows the response time comparisions. The system with the SI policy doesn't reduce the sleep probability as the arrival rate increases; hence, it results in higher response times than other power-saving policies with a

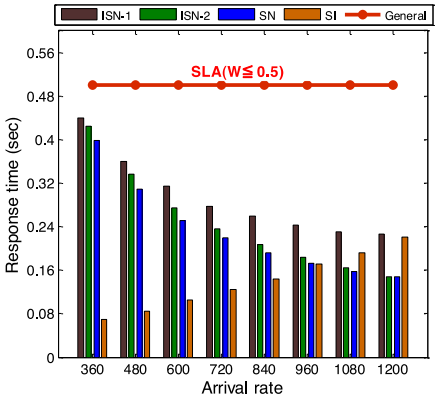

Fig. 21. Response time comparisons.

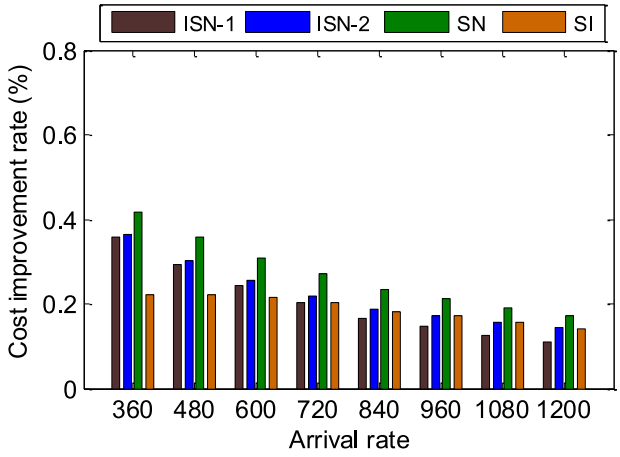

Fig. 23. Operational cost improvement rates.

higher arrival rate. For the general policy, the response times are all kept at the 0.5 sec.

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. 22. Finally, we measure the cost improvement rates, which calculate the relative value of improvements to the original value instead of an absolute value, the results are shown in Fig. 23. It's noted that the proposed power-saving policies can effectively reduce cost, especially when an arrival rate is low. 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.

6 Conclusion

The growing crisis in power shortages has brought a concern in existing and future cloud system designs. To mitigate unnecessary idle power consumption, three power-saving policies with different decision processes and mode-switching controls are considered. 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. The issue of choosing a suitable policy among diverse power managements 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 significantly improve the response time in a low arrival rate situation. On the other hand, applying others policies can obtain more cost benefits when the startup cost is high. As compared to a general policy, cost savings and response time improvement can be verified.

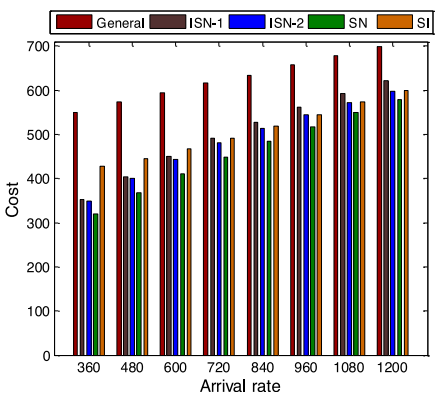

Fig. 22. Operational cost comparisons.

References

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

[2] R. Ranjan, L. Zhao, X. Wu, A. Liu, A. Quiroz, and M. Parashar, "Peer-to-peer cloud provisioning: Service discovery and load-balancing," in *Cloud Computing*. London, U.K.: Springer, 2010, pp. 195–217.

[3] R. N. Calheiros, R. Ranjan, and R. Buyya, "Virtual machine provisioning based on analytical performance and QoS in cloud computing environments," in *Proc. Int. Conf. Parallel Process.*, 2011, pp. 295–304.

[4] Server virtualization has stalled, despite the hype [Online]. Available: http://www.infoworld.com/print/146901, 2010.

[5] Y. C. Lee and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," *J. Supercomput.*, 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," *Adv. Comput.*, vol. 82, pp. 47–111, 2011.

[7] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.

[8] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: A perspective study," *New Generation Comput.*, 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 Comput.: Practice Exp.*, vol. 24, no. 9, pp. 935–937, 2012.

[10] M. Yadin and P. Naor, "Queueing systems with a removable service station," *Operations Res.*, vol . 14, pp. 393–405, 1963.

[11] W. Huang, X. Li, and Z. Qian, "An energy efficient virtual machine placement algorithm with balanced resource utilization," in *Proc. 7th Int. Conf. Innovative Mobile Internet Serv. Ubiquitous Comput.*, 2013, pp. 313–319.

[12] R. Nathuji, K. Schwan, A. Somani, and Y. Joshi, "VPM tokens: Virtual machine-aware power budgeting in datacenters," *Cluster Comput.*, 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," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci.*, 2012, pp. 42–49.

[14] K. Ye, D. Huang, X. Jiang, H. Chen, and S. Wu, "Virtual machine based energy-efficient data center architecture for cloud computing: A performance perspective," in *Proc. IEEE/ACM Int. Conf. Green Comput. Commun. Int. Conf. Cyber, Phys. Soc. Comput.*, 2010, pp.171–178.

[15] G. P. Duggan and P. M. Young, "A resource allocation model for energy management systems," in *Proc. IEEE Int. Syst. Conf.*, 2012, pp. 1–3.
[16] M. Mazzucco, D. Dyachuky, and R. Detersy, "Maximizing Cloud Providers Revenues via Energy Aware Allocation Policies," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, 2010, pp. 131–138.
[17] Q. Zhang, M. Zhani, R. Boutaba, and J. Hellerstein, "Dynamic heterogeneity-aware resource provisioning in the cloud," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 14–28, Jan.–Mar. 2014.
[18] M. Guazzone, C. Anglano and M. Canonico, "Energy-efficient resource management for cloud computing infrastructures," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci.*, 2011, pp. 424–431.
[19] A. Amokrane, M. Zhani, R. Langar, R. Boutaba, and G. Pujolle, "Greenhead: Virtual data center embedding across distributed infrastructures," *IEEE Trans. Cloud Comput.*, vol. 1, no. 1, pp. 36–49, Jan.–Jun. 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 Trans. Cloud Comput.*, vol. 1, no. 1, pp. 22–35, Jan.–Jun. 2013.
[21] Y. Deng, W. J. Braun, and Y. Q. Zhao, "M/M/1 queueing system with delayed controlled vacation," *OR Trans.*, vol. 3, pp. 17–30, 1999.
[22] K. H. Wang and H. M. Huang, "Optimal control of an $M/E_k/1$ queueing system with a removable service station," *J. Oper. Res. Soc.*," vol. 46, pp. 1014–1022, 1995.
[23] Y. Levy and U. Yechiali, "Utilization of idle time in an M/G/1 queueing system," *Manage. Sci.*, vol. 22, no. 2, pp. 202–211, 1975.
[24] T. Naishuo, Z. Daqing, and C. Chengxuan, "M/G/1 queue with controllable vacations and optimization of vacation policy," *Acta Math. Appl. Sinica*, vol. 7, no. 4, pp. 363–373, 1991.
[25] D. A. Wu and H. Takagi, "M/G/1 queue with multiple working vacations," *Perform. Eval.*, vol. 63, no. 7, pp. 654–681, 2006.
[26] M. Zhang, and Z. Hou, "M/G/1 queue with single working vacation," *J. Appl. Math. Comput.*, vol. 39, no. 1–2, pp. 221–234, 2012.
[27] D. Meisner, B. T. Gold, and T. F. Wenisch, "PowerNap: Eliminating server idle power," in *Proc. 14th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2009, pp. 205–216.
[28] H. Mi, H. Wang, G. Yin, Y. Zhou, D. Shi, and L. Yuan, "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in *Proc. IEEE Int. Conf. Serv. Comput.*, 2010, pp. 514–521.
[29] H. AbdelSalam, K. Maly, R. Mukkamala, M. Zubair, and D. Kaminsky, "Towards energy efficient change management in a cloud computing environment," in *Proc. 3rd Int. Conf. Auton. Infrastructure, Manage. Security: Scalability Netw. Serv.*, 2009, pp. 161–166.
[30] J. Song, T. Li, Z. Wang, and Z. Zhu, "Study on energy-consumption regularities of cloud computing systems by a novel evaluation model," *Computing*, vol. 95, no. 4, pp. 269–287, 2013.

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

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

Ching-Hsien (Robert) Hsu is a professor in the Department of Computer Science and Information Engineering at Chung Hua University, Taiwan; and a distinguished chair professor in the School of Computer and Communication Engineering at Tianjin University of Technology, China. His research includes high-performance computing, cloud computing, parallel and distributed systems. He has published 200 papers in refereed journals, conference proceedings, and book chapters in these areas. He has been involved in more than 100 conferences and workshops as various chairs and more than 200 conferences/workshops as a program committee member. He is a senior member of the IEEE.