Research Article

A Correlated Model for Evaluating Performance and Energy of Cloud System Given System Reliability

Hongli Zhang, Panpan Li, and Zhigang Zhou

School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

Correspondence should be addressed to Panpan Li; panleehit@yeah.net

Received 14 January 2015; Revised 22 April 2015; Accepted 28 April 2015

Academic Editor: Juan R. Torregrosa

Copyright © 2015 Hongli Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The serious issue of energy consumption for high performance computing systems has attracted much attention. Performance and energy-saving have become important measures of a computing system. In the cloud computing environment, the systems usually allocate various resources (such as CPU, Memory, Storage, etc.) on multiple virtual machines (VMs) for executing tasks. Therefore, the problem of resource allocation for running VMs should have significant influence on both system performance and energy consumption. For different processor utilizations assigned to the VM, there exists the tradeoff between energy consumption and task completion time when a given task is executed by the VMs. Moreover, the hardware failure, software failure and restoration characteristics also have obvious influences on overall performance and energy. In this paper, a correlated model is built to analyze both performance and energy in the VM execution environment given the reliability restriction, and an optimization model is presented to derive the most effective solution of processor utilization for the VM. Then, the tradeoff between energy-saving and task completion time is studied and balanced when the VMs execute given tasks. Numerical examples are illustrated to build the performance-energy correlated model and evaluate the expected values of task completion time and consumed energy.

1. Introduction

One of the important criteria for appraising the superiority of modern computing systems is whether it satisfies the increasing demand for high performance and energy-saving [1, 2]. Due to the issue of increasing energy consumption in large-scale computing systems, many efficient techniques, such as dynamic voltage and frequency scaling [1] and virtual resource management [3], have been proposed to control energy consumption. On the other hand, distributed resource sharing technology [4] which effectively improves the performance of system has been more widely employed in computing systems, especially for cloud computing systems. Meanwhile, how to guarantee the reliability of a complex system is always an important research issue.

Although these technologies and methods can somehow solve the corresponding issues, it is inadequate to handle these metrics separately. The existing approaches cannot be used in the situation of studying the correlation among energy, performance, and reliability.

Cloud computing is an emerging technology recently, which has numerous novel features, such as large-scale resource sharing, dynamic and flexible resource management, and on-demand resource provisioning [5]. Cloud computing takes advantage of Grid technology which enables integration of resources across distributed heterogeneous dynamic virtual organizations [6]. A grid service is designed to execute a certain task under the control of the resource management system (RMS) [7]. Similarly, the cloud computing system has cloud operating system (COS) to flexibly schedule computational resources (including CPU, memory, storage, and bandwidth) for task execution. Moreover, autonomic computing technologies can be applied in local computational environments, which enables dynamic application scale-out and live migration of virtual machine (VM) to achieve more efficient resource utilization and address dynamic workloads requirements [8, 9].

In a cloud computing system, a task is usually executed by a VM of which computational ability directly depends on the resources assigned by the COS. If the COS decreases the

number of CPU cores or the CPU utilization for the VM, the power consumption could be effectively reduced. However, such approach also causes lower computational speed of the VM, which results in longer task completion time and more chance of failures. On the other hand, the occurrence of failures usually causes the increase in task completion time which results in the decrease of performance waiting for redoing the task. The reexecution further consumes more electricity power. Thus, energy, performance, and reliability are closely related and affect each other, which should not be separated in modeling and scheduling.

To solve the problem that huge energy waste typically exists in large-scale distributed systems, there are many studies on energy reduction. The low average utilization rate of resources in computing systems generally creates enormous waste of computing ability and causes high energy consumption in cooling and other overheads [10]. As this situation is the most obvious factor that induces energy problems, most of the existing researches focus on energy-efficient consolidation of computing resources based on energy consumption prediction [11], required Quality of Service [12], memory-aware virtual machine scheduling [13], load balancing strategy [14], and control-theoretic techniques of multiple high-density servers [15]. However, as mentioned above, it is inadequate to solve the energy problem without considering reliability and performance. Taking consolidated processors for example, once hardware failures of a processor occur, all tasks executed by the same processor cannot perform immediately, which causes a common cause failure (CCF) and induces a decrease in reliability. This situation is typical in cloud computing systems which widely employ virtualization technology to improve the average utilization of computing resources. Thus, the precise evaluation of energy consumption should consider not only software failures but also hardware failures. Dai et al. [16] studied the correlated software failures of multiple types and analyzed the uncertainty of software reliability based on maximum-entropy principle [17]. However, as mentioned above, the analysis for the reliability of distributed computing systems should take hardware failures into account. So Dai et al. [18] studied the combination of various failures interacting with one another and presented the hierarchical model for the grid service reliability analysis and evaluation. There are also many other reliability models for software-hardware systems; Markov models are usually used to analyze and evaluate the reliability [19, 20]. For the performance, it is always a research focus. Meyer [21] proposed the notion of performability which can effectively evaluate both performance and reliability. Then, the performability evaluation for multiprocessor systems [22], fault-tolerant computer systems [23], and distributed real-time systems [24] were studied. For a grid computing system, Dai et al. [25] presented a combined model of performance and reliability, in which the precedence constraints caused by data dependence and the common cause failure were considered. Moreover, the optimal resource allocation for both performance and reliability in grid systems also was studied [26].

Since reliability, performance, and energy cannot be treated separately, this paper proposes a correlated model for evaluating both performance and energy based on

the analysis for hardware and software reliability. The primary innovation of this correlated model focuses on the essential connection between performance, energy, and reliability which is provided by resource allocation in cloud systems. A semi-Markov process is formulated for the modeling of software/hardware reliability and the evaluation of performance and energy is based on Laplace-Stieltjes transform and Bayesian approach. A new functional relationship between expected energy consumption and processor utilization is constructed. According to the analysis of the derivative of the function, it is easy to derive an optimal resource allocation to minimize energy consumption in a task completion procedure. This optimal resource allocation also implies the balance of tradeoff between power consumption and task completion time.

The remainder of the paper is organized as follows. Section 2 describes a performability model considering both hardware failures and software failures in cloud computing systems. Section 3 presents a power consumption model to evaluate the expected energy consumption. Based on the evaluation of energy consumption, a feasible approach that derives an optimal processor utilization to reduce energy consumption has been proposed. Section 4 illustrates several numerical examples.

2. Performability Model for Task Process

In a cloud computing system, tasks are usually executed by virtual machines which provide isolation technology to ensure noninterfering share of various computing resources, such as CPU, memory, and hard disk. Considering that the energy consumption caused by processor operation is the major constituent of total energy consumption of servers [10], reasonable CPU allocation for running VMs has significant effects on the tradeoff balance between reliability, performance, and energy consumption. The following model will first analyze and evaluate performability based on processor frequencies which are assigned to VMs for completing given tasks.

2.1. Hardware and Software Reliability Model. In this paper, the presented reliability model considers hardware failures of the processor and software failures of VMs. In cloud computing environments, a single physical computing node usually runs multiple VMs to execute tasks simultaneously and hardware failures of the processor will terminate the operations of all VMs. As a consequence, the influence of hardware failures plays an important role in the reliability. The design method considering the quality tradeoff between hardware and software components is called hardware/software codesign [27]. According to the properties of cloud computing, the following assumptions are made for reliability modeling to run a VM:

- (1) Once a hardware failure of processor occurs, the system cannot operate and start to restore. A running VM is aborted when a hardware failure occurs and it will be reexecuted after the recovery of the hardware.

FIGURE 1: Markov interpretation of the VM execution process.

- (2) A software failure of a VM is an obvious failure which can be detected by the cloud operating system immediately. A running VM is instructed to suspend as soon as a software failure is detected.
- (3) Software restoration actions halt a VM which has been suspended and create a new instance of the same virtual image. The given task which is executed by the VM will be restarted anew (preemptive repeat mode) when the software restoration action is complete.
- (4) For all VMs which are created from the same VM template, the software parameters will not be changed. The execution of these VMs is independent and identically distributed (i.i.d.).
- (5) If a VM finishes a given task, it will be shut down by the cloud operation system immediately.

Let the stochastic process $\{X(t), t \geq 0\}$ represent the state of the system at the time point t , as shown in Figure 1. State i represents the start of the i th run of the VM. If the VM does not finish the given task within the i th run (i.e., a hardware or software failure occurs before the completion of the given task), $X(t)$ will finally transit to $i + 1$ and the VM will be restarted again to reexecute the given task. State H represents a hardware failure of the processor occurs and, according to assumption (A1), the hardware failure also induces the termination of the VM and the start of a restoration action. Both the hardware uptime (U_H) or time to hardware failure and the hardware downtime (D_H) or time to hardware repair are random variables. Similarly, state S represents a software failure of the VM occurs. U_S and D_S are the random times which represent the software uptime or software failure time and software downtime or the software restoration time, respectively. In general, U_H , D_H , U_S , and D_S follow the exponential distributions with means $1/\theta$, $1/\eta$, $1/\lambda$, and $1/\mu$, respectively [28].

2.2. Cumulative Distribution Function of Time between Two Successive Runs. In this paper, we named the random time interval from the beginning of the i th ($i = 1, 2, 3, \dots$) run of the VM to the beginning of the next ($i + 1$)st run of the VM (i.e., from state i to $i + 1$) as i th instance lifetime of the VM. According to Figure 1, the VM keeps operating until a failure (i.e., a hardware or software failure) occurs during a run of the VM. Denote the random operation time of the VM as U_{VM} , which is not only determined by the software failure but also affected by the hardware failure; that is, $U_{VM} = \min(U_H, U_S)$.

If assumption (A5) is not considered, the distribution of U_{VM} can be obtained as

$$\begin{aligned} F'_{VM}(t) &= \Pr\{U_{VM} < t\} = 1 - \Pr\{U_H \geq t\} \Pr\{U_S \geq t\} \\ &= 1 - e^{-(\lambda+\theta)t} \quad (0 \leq t < \infty). \end{aligned} \quad (1)$$

Suppose the given task needs to execute a number of instructions w (i.e., work requirement). In an idealistic failure-free scenario, the completion time of the given task is determined by the computational speed of the VM. Denote such idealistic task completion time as t_0 . However, in realistic hardware and software failure scenario, the task will be interrupted upon a failure. The probability that the given task is complete within a run of the VM is given by

$$p = \Pr\{U_H \geq t_0\} \Pr\{U_S \geq t_0\} = e^{-(\lambda+\theta)t_0}. \quad (2)$$

Moreover, we should note that the idealistic task completion time t_0 gives a bound on the operational time of the VM under assumptions (A3) and (A5). That means all possible values of operation time U_{VM} must satisfy $0 \leq t \leq t_0$. In fact, the bound on the operation time is especially relevant to define performance and energy consumption. Similar to the analysis mentioned by Sheahan et al. [29], the cdf of bounded operation time U_{VM} is given by

$$\begin{aligned} F_{VM}(t) &= \Pr\{U_{VM} < t \mid U_{VM} < t_0\} = \frac{1 - e^{-(\lambda+\theta)t}}{1 - e^{-(\lambda+\theta)t_0}} \\ &\quad (0 \leq t \leq t_0). \end{aligned} \quad (3)$$

Then, we have the probability density function (pdf) of U_{VM} :

$$f_{VM}(x) = \frac{(\lambda + \theta) e^{-(\lambda+\theta)x}}{1 - e^{-(\lambda+\theta)t_0}} \quad (0 \leq x \leq t_0). \quad (4)$$

The Laplace-Stieltjes transform (LST) of $F(t)$ is defined as $\tilde{F}(s) \equiv \int_0^\infty e^{-st} dF(t)$, so the LST of $F_{VM}(t)$ becomes

$$\begin{aligned} \tilde{F}_{VM}(s) &= \int_0^{t_0} e^{-sx} f_{VM}(x) dx \\ &= \frac{\lambda + \theta}{\lambda + \theta + s} \left[\frac{1 - e^{-(\lambda+\theta+s)t_0}}{1 - e^{-(\lambda+\theta)t_0}} \right]. \end{aligned} \quad (5)$$

For the stochastic process $\{X(t), t \geq 0\}$, let $P_{A,B}(t)$ represent the one-step transition probability from state A to state B during the time interval t . Note that the state transition from i to H implies that the hardware failure occurs before the occurrence of the software failure. Similarly, the state transition from i to S also implies that the software failure

occurs before the occurrence of the hardware failure. Subject to bound $U_{VM} < t_0$, the expression for $P_{A,B}(t)$ is given by

$$\begin{aligned}
 P_{i,H}(t) &= \Pr\{U_H < t, U_H < U_S \mid X(0) = i, U_{VM} < t_0\} \\
 &= \int_0^t \frac{\theta e^{-\theta x} e^{-\lambda x}}{1 - e^{-(\lambda+\theta)t_0}} dx = \frac{\theta}{\lambda + \theta} \left(\frac{1 - e^{-(\lambda+\theta)t}}{1 - e^{-(\lambda+\theta)t_0}} \right) \\
 &\quad (0 \leq t \leq t_0), \\
 P_{i,S}(t) &= \Pr\{U_S < t, U_S < U_H \mid X(0) = i, U_{VM} < t_0\} \\
 &= \int_0^t \frac{\lambda e^{-\lambda x} e^{-\theta x}}{1 - e^{-(\lambda+\theta)t_0}} dx = \frac{\lambda}{\lambda + \theta} \left(\frac{1 - e^{-(\lambda+\theta)t}}{1 - e^{-(\lambda+\theta)t_0}} \right) \\
 &\quad (0 \leq t \leq t_0),
 \end{aligned} \tag{6}$$

$$\begin{aligned}
 P_{H,i+1}(t) &= \Pr\{D_H < t \mid X(0) = H\} = 1 - e^{-\eta t} \\
 &\quad (0 \leq t < \infty), \\
 P_{S,i+1}(t) &= \Pr\{D_S < t \mid X(0) = S\} = 1 - e^{-\mu t} \\
 &\quad (0 \leq t < \infty).
 \end{aligned}$$

Denote the first passage time of $X(t)$ from state i to state $i + 1$ as $T_{i,i+1}$; that is, $T_{i,i+1}$ ($i = 1, 2, 3, \dots$) represents the i th instance lifetime of the VM and each of the lifetimes is an i.i.d. random variable under assumption (A4). Then we have the cumulative distribution of time between two successive runs of the VM (i.e., an instance lifetime of the VM) as follows:

$$\begin{aligned}
 F_{i,i+1}(t) &= \Pr\{T_{i,i+1} < t\} \\
 &= P_{i,H}(t) * P_{H,i+1}(t) + P_{i,S}(t) * P_{S,i+1}(t),
 \end{aligned} \tag{7}$$

where $F_{i,i+1}(t)$ represents the distribution of $T_{i,i+1}$ and “*” denotes the Stieltjes convolution of the two functions. Applying the LST to (7), we can obtain

$$\tilde{F}_{i,i+1}(s) = \tilde{P}_{i,H}(s) \cdot \tilde{P}_{H,i+1}(s) + \tilde{P}_{i,S}(s) \cdot \tilde{P}_{S,i+1}(s). \tag{8}$$

Then, applying the LST to (6) and substituting the corresponding transformed expressions into (8) yields

$$\begin{aligned}
 \tilde{F}_{i,i+1}(s) &= \frac{1 - e^{-(s+\lambda+\theta)t_0}}{1 - e^{-(\lambda+\theta)t_0}} \left(\frac{\theta}{s + \theta + \lambda} \cdot \frac{\eta}{s + \eta} \right. \\
 &\quad \left. + \frac{\lambda}{s + \theta + \lambda} \cdot \frac{\mu}{s + \mu} \right).
 \end{aligned} \tag{9}$$

2.3. Expected Task Completion Time. Since a given task is executed by the VM repeatedly until the first operational time U_{VM} is longer than t_0 , the number of runs of the VM is a random variable which follows a geometric distribution.

Suppose the completion procedure of the given task exactly passes $N+1$ runs of the VM; that is, it contains N unsuccessful runs (the given task is not complete in these runs) and one successful run (the given task is complete in the run) of the VM. Let $p(n)$ be the probability that the task completion procedure occupies n unsuccessful runs and one successful run. From (2), we can obtain

$$\begin{aligned}
 p(n) &= \Pr\{N = n\} = (1 - p)^n p \\
 &= e^{-(\lambda+\theta)t_0} [1 - e^{-(\lambda+\theta)t_0}]^n.
 \end{aligned} \tag{10}$$

Here, the completion time of the given task is defined as the time interval from the starting time when the task is first executed by the VM to the end time when the task is finally finished by the VM. In general, we can set the time origin $t = 0$ as the starting time. As to the completion time of the given task which can be denoted as T_{task} , under the condition of (10), it consists of the sum of n instance lifetimes of VM and a final operational time that equals t_0 . Let $F_{\text{task}}(t \mid n) = \Pr\{T_{\text{task}} < t \mid n\}$ represent the conditional distribution of T_{task} . It can in principle be found by taking the convolution of $F_{i,i+1}(t)$ with itself n times and then with $\delta(t - t_0)$. Since we already get $\tilde{F}_{i,i+1}(s)$ from (8), the LST of the conditional distribution $F_{\text{task}}(t \mid n)$ can be obtained by

$$\tilde{F}_{\text{task}}(t \mid n) = [\tilde{F}_{i,i+1}(s)]^n e^{-st_0}, \tag{11}$$

where e^{-st_0} is the LST of $\delta(t - t_0)$ and $[\tilde{F}_{i,i+1}(s)]^n$ is the LST of $F_{1,2}(t) * F_{2,3}(t) * \dots * F_{n-1,n}(t)$. As mentioned above, each of the instance lifetimes ($T_{i,i+1}, i = 1, 2, 3, \dots$) is an i.i.d. random variable. Therefore, the equation $\tilde{F}_{1,2}(s) = \tilde{F}_{2,3}(s) = \dots = \tilde{F}_{n-1,n}(s) = \tilde{F}_{i,i+1}(s)$ is satisfied.

Now, for the unconditional distribution $F_{\text{task}}(t) = \Pr\{T_{\text{task}} < t\}$, using (10) and the Bayesian theorem on conditional probability, the condition in (11) can be removed and the LST of the task completion time becomes

$$\begin{aligned}
 \tilde{F}_{\text{task}}(s) &= \sum_{n=0}^{\infty} \tilde{F}_{\text{task}}(t \mid n) \Pr\{N = n\} \\
 &= \frac{(s + \lambda + \theta) e^{-(s+\lambda+\theta)t_0}}{(s + \lambda + \theta) - (\theta\eta/(s + \eta) + \lambda\mu/(s + \mu)) (1 - e^{-(s+\lambda+\theta)t_0})}.
 \end{aligned} \tag{12}$$

Due to the fact that LST has property that becomes a Moment Generator, we can derive the expected time of T_{task} as

$$\begin{aligned}
 E[T_{\text{task}}] &= - \left. \frac{d\tilde{F}_{\text{task}}(s)}{ds} \right|_{s=0} \\
 &= \frac{(\mu\eta + \theta\mu + \lambda\eta)}{\mu\eta(\lambda + \theta)} (e^{(\lambda+\theta)t_0} - 1).
 \end{aligned} \tag{13}$$

3. Evaluation of Energy Consumption

3.1. Power Consumption Modeling. To estimate the energy consumed in the entire task completion procedure, the power

consumption model is of critical importance besides the random task completion time. There exist several studies introducing the power consumption models for the processor. Choi et al. [11] discussed the statistic models for the power usage distribution and the analytical method for the nonlinear power consumption curves, respectively. Wang et al. [15] developed the piecewise linear function for the power consumption of the processor. Furthermore, Lee [30] considered that the imperfectly linear model for the power consumption can be linearized to decrease the complexity and computation overhead. In this paper, we apply the power consumption model introduced by Zhu et al. [31], which can be summarized as

$$H = c_e \cdot f^m + b, \quad (14)$$

where H and f are the power consumption and the processing frequency of a processor, respectively. $c_e \cdot f^m$ in (14) is the frequency-dependent active power, in which c_e is the effective switching capacitance and m is the dynamic power exponent. b in (14) is the sum of the sleep power maintaining basic circuits and the frequency-independent active power. These parameters are system dependent constants which can be estimated by the statistical analysis. For easy discussion, the frequencies of processor can be normalized by processor utilization. Suppose the maximum frequency of processor is $f_{\max}$, that is, $f = x f_{\max}$, in which x ($0 < x \leq 1$) is the utilization of processor. Let $a = c_e \cdot f_{\max}^m$; (14) can be transformed to

$$H(x) = ax^m + b \quad (0 < x \leq 1). \quad (15)$$

We should notice that b is the basic power consumption to keep the computational node working and it always has a relatively large value. In fact, this also implies significant overhead of turning on/off a computational node (with corresponding $x \neq 0$, $H(x) \geq b$; $x = 0$, $H(x) = 0$).

To analyze the situation of the power consumption in an instance lifetime, we divide an instance lifetime into two phases: operational phase and restoration phase. The operational phase is the time interval in which the VM keeps operating until a failure occurs. The processing frequency of the processor remains unchanged during the operational phase which means the power consumption is $H(x) = ax^m + b$ with a fixed utilization x . In contrast, the processing frequency of the processor is different in the restoration phase in which a restoration action starts immediately after a failure occurs. In the restoration phase, the power supply for keeping the running of basic circuits, clock, and processor is still sustained, but the VM cannot operate until the restoration action is complete. Based on such case, the following assumptions are made for the power consumption in restoration and operation phases:

- (1) In a restoration phase, a relative small utilization of the processor for a restoration action is negligible here. Thus the power consumption in a restoration phase is $H(0_+) = \lim_{x \rightarrow 0_+} H(x) = b$.
- (2) Apply the underlying resource virtualization technology; the COS keeps the utilization of the processor x

FIGURE 2: Power consumption for an instance lifetime of the VM.

and the power consumption $H(x)$ for any VM unchanged in an operational phase.

The power consumption for an instance lifetime of the VM is shown in Figure 2.

3.2. Expected Energy Consumption for a Task Completion Procedure. Let Y be a random variable representing random energy consumption. Obviously, energy consumption is the product of a power consumption and a time; that is, $Y = H(x)T$, and the distribution of Y can be derived by

$$\begin{aligned} F_Y(y) &= \Pr\{Y < y\} = \Pr\{H(x)T < y\} \\ &= \Pr\left\{T < \frac{y}{H(x)}\right\} = F_T\left(\frac{y}{H(x)}\right), \end{aligned} \quad (16)$$

where $F_T(t)$ is the distribution of a random time T . The power consumption $H(x)$ is a constant with a fixed utilization x . Applying properties of LST, we can get the LST of $F_Y(y)$ as

$$\tilde{F}_Y(s) = \tilde{F}_T[H(x)s], \quad H(x) > 0. \quad (17)$$

As mentioned above, a given task has an amount of work requirement w . In this paper, the work requirement w is measured in the number of commands or instructions to be executed. Suppose the maximum computational speed of the processor is $c_{\max}$ and the utilization of the processor that the cloud operation system assigns to the VM is x ($0 < x \leq 1$). According to the previous similar study [25], the idealistic task completion time t_0 in a failure-free scenario should be

$$t_0 = \frac{w}{xc_{\max}}. \quad (18)$$

Thus, under the bound that an operation time U_{VM} must be less than t_0 , we can get the energy distribution between two successive runs of the VM. For the stochastic process $\{X(t), t \geq 0\}$, let $Q_{A,B}(\tau)$ represent the transition probability

from state A to state B within the energy consumption τ . Substituting (17) into (6) yields

$$\begin{aligned}\tilde{Q}_{i,H}(s) &= \tilde{P}_{i,H}[H(x)s], \\ \tilde{Q}_{H,i+1}(s) &= \tilde{P}_{H,i+1}[H(0_+)s], \\ \tilde{Q}_{i,S}(s) &= \tilde{P}_{i,S}[H(x)s], \\ \tilde{Q}_{S,i+1}(s) &= \tilde{P}_{S,i+1}[H(0_+)s].\end{aligned}\quad (19)$$

Let $Y_{i,i+1}$ denote the random energy consumed in the i th instance lifetime of the VM. We can derive the distribution of energy consumed in i th instance lifetime of the VM as

$$\begin{aligned}Q_{i,i+1}(\tau) &= Q_{i,H}(\tau) * Q_{H,i+1}(\tau) + Q_{i,S}(\tau) \\ &\quad * Q_{S,i+1}(\tau),\end{aligned}\quad (20)$$

$$\begin{aligned}\tilde{G}_{\text{task}}(s) &= \sum_{n=0}^{\infty} \tilde{G}_{\text{task}}(y|n) \Pr\{N=n\} \\ &= \frac{(sH(x) + \lambda + \theta) e^{-(sH(x) + \lambda + \theta)t_0}}{(sH(x) + \lambda + \theta) - (\theta\eta/(sH(0_+) + \eta) + \lambda\mu/(sH(0_+) + \mu))(1 - e^{-(sH(x) + \lambda + \theta)t_0})}.\end{aligned}\quad (23)$$

Then we can derive the expected energy consumption for a task completion procedure as

$$\begin{aligned}E[Y_{\text{task}}] &= - \left. \frac{d\tilde{G}_{\text{task}}(s)}{ds} \right|_{s=0} \\ &= \frac{\mu\eta H(x) + \mu\theta H(0_+) + \lambda\eta H(0_+)}{\mu\eta(\lambda + \theta)} (e^{(\lambda + \theta)t_0} - 1).\end{aligned}\quad (24)$$

3.3. Analysis of Energy Consumption

3.3.1. Expected Optimal Processor Utilization to Reduce Energy. Based on the derivation mentioned above, we can get the expected task completion time and the expected energy consumption from (13) and (24), respectively. Moreover, for a fixed given task work requirement w , these two important indices are functions of the processor utilization x ; that is, by substituting (18) into (13), we can get the functional relationship between the expected task completion time and the processor utilization $E(T_{\text{task}}) = g_T(x)$. Similarly, for the expected energy consumption, by substituting (15) and (18)

where $Q_{i,i+1}(\tau)$ is the distribution of $Y_{i,i+1}$; that is, $Q_{i,i+1}(\tau) = \Pr\{Y_{i,i+1} < \tau\}$. Then, from (19), the LST of $Q_{i,i+1}(\tau)$ becomes

$$\begin{aligned}\tilde{Q}_{i,i+1}(s) &= \tilde{P}_{i,H}[H(x)s] \cdot \tilde{P}_{H,i+1}[H(0_+)s] \\ &\quad + \tilde{P}_{i,S}[H(x)s] \cdot \tilde{P}_{S,i+1}[H(0_+)s] \\ &= \frac{1 - e^{-(sH(x) + \lambda + \theta)t_0}}{1 - e^{-(\lambda + \theta)t_0}} \left(\frac{1}{sH(x) + \theta + \lambda} \right) \\ &\quad \cdot \left(\frac{\theta\eta}{sH(0_+) + \eta} + \frac{\lambda\mu}{sH(0_+) + \mu} \right).\end{aligned}\quad (21)$$

Under condition (10), the cumulative energy conditional distribution for the task completion can be denoted as $G_{\text{task}}(y|n)$, which can be derived from (11), (17), and (21) as

$$\tilde{G}_{\text{task}}(y|n) = [\tilde{Q}_{i,i+1}(s)]^n e^{-sH(x)t_0}.\quad (22)$$

Similar as for the expected task completion time, from (10), applying the Bayesian theorem and removing the condition on n give

into (24), the function $E(Y_{\text{task}}) = g_Y(x)$ is satisfied. The derivative of $g_T(x)$ can be obtained as

$$\frac{dg_T(x)}{dx} = - \frac{(\mu\eta + \theta\mu + \lambda\eta)}{\mu\eta} \left(\frac{w}{x^2 c_{\max}} \right) e^{(\lambda + \theta)(w/x c_{\max})}, \quad (25)$$

which satisfies $dg_T(x)/dx < 0$ for all $x > 0$. This phenomenon can be clearly explained. The higher the utilization of the processor is, the shorter the task completion time is and the lower the risk that the task will be interrupted by a failure is. However, we should notice that the energy consumption is as important as the performance. For the expected energy consumption, we do not give the explicit expression of $dg_Y(x)/dx$ here because the function of power consumption $H(x)$ will have different parameters which are determined by different physical computing nodes. If there exists an expected optimal processor utilization $0 < x_{\text{opt}} < 1$ to reduce the energy consumption, we can derive it by solving the following equation:

$$x_{\text{opt}}: \frac{dg_Y(x)}{dx} = 0.\quad (26)$$

3.3.2. Variance of Random Energy Consumption. Since we have got the LST of the cumulative energy distribution

$\tilde{G}_{\text{task}}(s)$ from (23), the variance of random energy consumption can be obtained by

$$E[Y_{\text{task}}^2] = \frac{d^2 \tilde{G}_{\text{task}}(s)}{ds^2} \Big|_{s=0}, \quad (27)$$

$$\text{Var}[Y_{\text{task}}] = E[Y_{\text{task}}^2] - (E[Y_{\text{task}}])^2.$$

4. Numerical Examples

4.1. Description and Assumptions. We present several numerical examples on performance and energy analysis based on the above measures. First, as mentioned above, different physical nodes have different parameters for power models. For example, Colfax CX2266-N2, Sun Netra X4250, IBM DataPlex Server dx360 M3, and Intel Xeon X3470 have obviously different power curves [32]. Here, we apply physical node (HP ProLiant ML110G5 server) of which power consumption characteristics have been numerical analyzed by Beloglazov and Buyya [33]. Based on the real numerical results for power consumption [33], we can estimate the power consumption modes for HP G5 server as

$$H(x) = 94 + 41x^m \text{ (watts)}, \quad (28)$$

in which the peak power is $135w$ and the basic power is $94w$. These two parameters usually can be exactly measured by some analytical tools. For the parameter m , it may have different values in different application scenarios. Even if frequencies of the processor dominate the power consumption of a physical node, the change of power consumption also depends on a concrete configuration of the physical node and utilization of other components, such as memory, disk, and GPU. However, for a special kind of task or a special application scenario, it is feasible to estimate a special value of this parameter from numerical statistical analysis.

The CPU of G5 server is Intel Xeon 3075 (2 cores $\times$ 2660 MHz). The frequency of the server's CPU can be mapped onto MIPS ratings: 2660 MIPS each core of HP G5 server. The processor utilization of 100% can be achieved when all cores of the processor work at max frequencies in parallel. In fact, a task can be allowed to be split in order to enhance the utilization of the processor [34]. In this paper, the task executed by the VM is supposed to be multicore programming; that is, the maximum computational speed of HP G5 server is $c_{\text{max}} = 2 \times 2660$ MIPS.

4.2. Expected Task Completion Time. Set $\theta = 0.05$, $\eta = 6.0$, $\lambda = 0.2$, and $\mu = 12.0$ as numerical values for the parameters of random times U_H , D_H , U_S , and D_S (hours). Suppose the work requirement of the given task is $w = 3 \times 10^6$ million instructions. From (18), we can derive the perfect task completion time as

$$\begin{aligned} t_0 &= \frac{w}{xc_{\text{max}}} = \frac{3 \times 10^6}{2 \times 2660 \times 60 \times 60} \cdot \frac{1}{x} \\ &= 0.1566 \cdot \frac{1}{x} \text{ (hours)}. \end{aligned} \quad (29)$$

FIGURE 3: Relationship between task completion time and processor utilization.

Then, from (13), the expected task completion time can be obtained as

$$\begin{aligned} E[T_{\text{task}}] &= g_T(x) \\ &= \frac{72 + 0.05 \times 12 + 0.2 \times 6}{0.25 \times 72} (e^{0.25 \times 0.1566 \cdot 1/x} - 1) \\ &= 4.1 \times (e^{0.0392 \cdot 1/x} - 1). \end{aligned} \quad (30)$$

Figure 3 displays the relationship between task completion time and processor utilization. The curves in Figure 3 show that the task completion time (idealistic or expected) varies inversely with the processor utilization, which coincides with the analytical conclusion derived from (25). Moreover, the difference between idealistic completion time and expected completion time increases gradually with the decrease in the processor utilization. This phenomenon shows that the processor frequencies also have influence on reliability. Since a lower processor frequently induces an increase of task completion time, a longer idealistic task completion time implies a higher risk of failure, which finally results in a reduction in reliability.

4.3. Expected Energy Consumption. By substituting (28) and (29) into (24), we can obtain the expected energy consumption for the completion of the given task as

$$E[Y_{\text{task}}] = g_Y(x) = (164x^m + 385.4)(e^{0.0392 \cdot 1/x} - 1). \quad (31)$$

Because of the tradeoff between the power consumption and the task completion time, there may exist an optimal frequency of the processor which can effectively reduce the energy consumed in the completion of a task. As mentioned above, an estimation of m is also affected by a special application scenario. Figure 4(a) illustrates that different values of m have obvious effect on an optimal processor

FIGURE 4: Optimal frequency to reduce energy.

utilization x_{opt} . In general, the power consumption and the expected task completion time usually are the monotonically increasing and decreasing functions of the processor utilization, respectively. Thus the optimal processor indicates the balance of tradeoff between power consumption and task completion time. It is meaningful to find the optimal processor utilization because it can achieve the minimum energy consumption for the completion of task. From (26), optimal processor utilizations for $m = 5$, $m = 7$, and $m = 9$ are derived as $x_{\text{opt}} = 0.9059$, $x_{\text{opt}} = 0.8779$, and $x_{\text{opt}} = 0.8651$, respectively. We should notice that the optimal processor utilization is also determined by the other parameters of the physical node and it will be more important when the difference between the peak power and the basic power is especially large. For example, the power consumption of IBM iDataPlex Server dx360 M3 is $H(x) = 250x^m + 100$ [32]. The CPU of IBM dx360 M3 usually adopts Xeon E5606 of which maximum frequency is 2.13 GHz. For the same work requirement and the same reliability parameters, the expected energy consumption can be obtained as

$$g_Y(x) = (1000x^m + 410)(e^{0.0978 \cdot 1/x} - 1) \quad (32)$$

and then the optimal frequency for $m = 2$, $m = 3$, $m = 4$, and $m = 5$ are $x_{\text{opt}} = 0.6887$, $x_{\text{opt}} = 0.6138$, $x_{\text{opt}} = 0.6241$, and $x_{\text{opt}} = 0.6462$, respectively. This means that the tradeoff between power consumption and task completion time for IBM dx360 M3 server is more obvious than for HP G5 server, as shown in Figure 4(b).

Based on analysis above, it is reasonable to make the VM work at a state when the processor utilization is not less than the optimal processor utilization. If the present utilization assigned to the VM is $x < x_{\text{opt}}$, enhancing processor utilization to x_{opt} decreases not only task completion time

but also energy consumption, which means x_{opt} is more reasonable than all $x < x_{\text{opt}}$ for both performance and energy. However, for the situation in which the present utilization has been $x \geq x_{\text{opt}}$, improving performance by increased processor utilization also induces an extra energy cost. In fact, for all $x \geq x_{\text{opt}}$, they are the noninferior solutions in energy-performance multiobjective optimization.

5. Conclusions and Future Work

The problem of energy consumption has been a serious topic of research for the last decade. Cloud computing technology is a newly developing method for flexible resource assignment. Typically, a given task is usually executed by a VM. If the COS has a reasonable resource assignment strategy for the VM, the energy consumed in the task completion procedure will be effectively reduced. This research proposed a modeling framework for the analysis of reliability, performance, and energy. With this model, both hardware and software failures are considered. It is capable of evaluating the expected performance and energy consumption. In addition, this research considered the tradeoff between power consumption and task completion time. The proposed model also provides a feasible approach to find an optimal processor utilization which implies a balance of this tradeoff. Based on the analysis of optimal processor utilization, the system can achieve minimum energy consumption when it completes a given task with a VM.

In cloud computing environments, a physical node can run multiple VMs in parallel. Then multiple tasks are executed in the same physical node simultaneously. This is another effective approach to save energy consumption. However, parallel execution of multiple VMs in a single

physical node is a more complicated situation. For example, even if the cloud isolation technology ensures the noninterfering between VMs, the hardware failure also has serious influence on the reliability, performance, and energy. Once a hardware failure occurs, all of the VMs will be terminated; the following restoration action and repeated executions of VMs will decrease the performance and increase the energy consumption. Reliability, performance, and energy-associated modeling of running multiple VMs parallelly remains as future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are grateful to the anonymous reviewers and the editor for their valuable comments and suggestions, which have led to a better presentation of this paper. This work is partially supported by the National Basic Research Program of China (973 Program) under Grant no. 2011CB302605, the National High Technology Research and Development Program of China (863 Program) under Grant no. 2011AA010705, and the National Science Foundation of China (NSF) under Grants nos. 61100188 and 61173144.

References

- [1] H. Aydin, R. Melhem, D. Mossé, and P. Mejia-Alvarez, "Power-aware scheduling for periodic real-time tasks," *IEEE Transactions on Computers*, vol. 53, no. 5, pp. 584–600, 2004.
- [2] W. Hanning, X. Weixiang, J. Yang, L. Wei, and J. Chaolong, "Efficient processing of continuous skyline query over smarter traffic data stream for cloud computing," *Discrete Dynamics in Nature and Society*, vol. 2013, Article ID 209672, 10 pages, 2013.
- [3] V. N. Hien, F. D. Tran, and J. M. Menaud, "Autonomic virtual resource management for service hosting platforms," in *Proceedings of the ICSE Workshop on Software Engineering Challenges of Cloud Computing (CLOUD '09)*, pp. 1–8, Vancouver, Canada, May 2009.
- [4] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "Grid services for distributed system integration," *Computer*, vol. 35, no. 6, pp. 37–46, 2002.
- [5] 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 Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.
- [6] D. Villegas, I. Roderio, I. Fong et al., "The role of grid computing technologies in cloud computing," in *Handbook of Cloud Computing*, pp. 183–218, Springer, 2010.
- [7] K. Krauter, R. Buyya, and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," *Software: Practice and Experience*, vol. 32, no. 2, pp. 135–164, 2002.
- [8] M. Parashar and S. Hariri, "Autonomic computing: an overview," in *Proceedings of the International Workshop on Unconventional Programming Paradigms*, pp. 257–269, September 2004.
- [9] H. Kim and M. Parashar, "CometCloud: an autonomic cloud engine," *Cloud Computing: Principles and Paradigms*, pp. 275–297, 2011.
- [10] J. Baliga, R. W. A. Ayre, K. Hinton, and R. S. Tucker, "Green cloud computing: balancing energy in processing, storage, and transport," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.
- [11] J. Choi, S. Govindan, J. Jeong, B. Urgaonkar, and A. Sivasubramanian, "Power consumption prediction and power-aware packing in consolidated environments," *IEEE Transactions on Computers*, vol. 59, no. 12, pp. 1640–1654, 2010.
- [12] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10)*, pp. 826–831, May 2010.
- [13] J.-W. Jang, M. Jeon, H.-S. Kim, H. Jo, J.-S. Kim, and S. Maeng, "Energy reduction in consolidated servers through memory-aware virtual machine scheduling," *IEEE Transactions on Computers*, vol. 60, no. 4, pp. 552–564, 2011.
- [14] J. Doyle, R. Shorten, and D. O'Mahony, "Stratus: load balancing the cloud for carbon emissions control," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 116–128, 2013.
- [15] X. Wang, M. Chen, and X. Fu, "MIMO power control for high-density servers in an enclosure," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 10, pp. 1412–1426, 2010.
- [16] Y.-S. Dai, M. Xie, and K.-L. Poh, "Modeling and analysis of correlated software failures of multiple types," *IEEE Transactions on Reliability*, vol. 54, no. 1, pp. 100–106, 2005.
- [17] Y.-S. Dai, M. Xie, Q. Long, and S.-H. Ng, "Uncertainty analysis in software reliability modeling by bayesian approach with maximum-entropy principle," *IEEE Transactions on Software Engineering*, vol. 33, no. 11, pp. 781–795, 2007.
- [18] Y.-S. Dai, Y. Pan, and X. Zou, "A hierarchical modeling and analysis for grid service reliability," *IEEE Transactions on Computers*, vol. 56, no. 5, pp. 681–691, 2007.
- [19] A. L. Goel and J. Soenjoto, "Models for hardware-software system operational performance evaluation," *IEEE Transactions on Reliability*, vol. 30, no. 3, pp. 232–239, 1981.
- [20] S. R. Welke, B. W. Johnson, and J. H. Aylor, "Reliability modeling of hardware/software systems," *IEEE Transactions on Reliability*, vol. 44, no. 3, pp. 413–418, 1995.
- [21] J. F. Meyer, "On evaluating the performability of degradable computing systems," *IEEE Transactions on Computers*, vol. 29, no. 8, pp. 720–731, 1980.
- [22] R. M. Smith, K. S. Trivedi, and A. V. Ramesh, "Performability analysis: measures, an algorithm, and a case study," *IEEE Transactions on Computers*, vol. 37, no. 4, pp. 406–417, 1988.
- [23] A. T. Tai, J. F. Meyer, and A. Aviziems, "Performability enhancement of fault-tolerant software," *IEEE Transactions on Reliability*, vol. 42, no. 2, pp. 227–237, 1993.
- [24] S. M. R. Islam and H. H. Ammar, "Performability analysis of distributed real-time systems," *IEEE Transactions on Computers*, vol. 40, no. 11, pp. 1239–1251, 1991.
- [25] Y.-S. Dai, G. Levitin, and K. S. Trivedi, "Performance and reliability of tree-structured grid services considering data dependence and failure correlation," *IEEE Transactions on Computers*, vol. 56, no. 7, pp. 925–936, 2007.
- [26] Y.-S. Dai, M. Xie, and K.-L. Poh, "Availability modeling and cost optimization for the grid resource management system," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, vol. 38, no. 1, pp. 170–179, 2008.

- [27] G. de Micheli, "Hardware/software co-design," *Proceedings of the IEEE*, vol. 85, no. 3, pp. 349–365, 1997.
- [28] K. Tokuno and S. Yamada, "Codesign-oriented performability modeling for hardware-software systems," *IEEE Transactions on Reliability*, vol. 60, no. 1, pp. 171–179, 2011.
- [29] R. Sheahan, L. Lipsky, and P. Fiorini, "The effect of different failure recovery procedures on the distribution of task completion times," in *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*, April 2005.
- [30] W. Y. Lee, "Energy-efficient scheduling of periodic real-time tasks on lightly loaded multicore processors," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 3, pp. 530–537, 2012.
- [31] D. K. Zhu, R. Melhem, D. Mossé, and E. Elnozahy, "Analysis of an energy efficient optimistic tmr scheme," in *Proceedings of the 10th International Conference on Parallel and Distributed Systems (ICPADS '04)*, pp. 559–568, July 2004.
- [32] G. Varsamopoulos, Z. Abbasi, and S. K. S. Gupta, "Trends and effects of energy proportionality on server provisioning in data centers," in *Proceedings of the 17th IEEE International Conference on High Performance Computing (HiPC '10)*, pp. 1–11, IEEE, Dona Paula, India, December 2010.
- [33] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency Computation Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.
- [34] K. Lakshmanan, R. Rajkumar, and J. P. Lehoczky, "Partitioned fixed-priority preemptive scheduling for multi-core processors," in *Proceedings of the 21st Euromicro Conference on Real-Time Systems (ECRTS '09)*, pp. 239–248, July 2009.