Received July 24, 2014, accepted September 3, 2014, date of publication September 19, 2014, date of current version October 1, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2358992

Power Metering for Virtual Machine in Cloud Computing–Challenges and Opportunities

CHONGLIN GU, HEJIAO HUANG, (Member, IEEE), AND XIAOHUA JIA, (Fellow, IEEE)
Department of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 150001, China
Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen 518055, China

Corresponding author: H. Huang (hjhuang@aliyun.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 11071271 and Grant 11371004 and in part by the Shenzhen Strategic Emerging Industries Program under Grant ZDSY20120613125016389, Grant JCYJ2012061315120-1451, and Grant JCYJ20130329153215152.

ABSTRACT The virtual machine (VM) is the most basic unit for virtualization and resource allocation. The study of VM power metering is the key to reducing the power consumption of data centers. In this paper, we make a comprehensive investigation in issues regarding VM power metering, including server models, sampling, VM power metering methods, and the accuracy of the methods. We will review many up-to-date power metering methods in this paper, and analyze their efficiencies, as well as evaluate their performance. Open research issues, such as VM service billing, power budgeting, and energy saving scheduling, are discussed, with an objective to spark new research interests in this field.

INDEX TERMS Virtual machine, cloud computing, power metering.

I. INTRODUCTION

Cloud computing has quickly become the most important way for people to obtain computing resources, information services and entertainment services. As data centers are getting more and more powerful, energy consumption becomes the primary concern of data centers. In a recent study regarding high energy cost of data centers, the overall energy cost by data centers is estimated to be 100 billion KWH per year [1], and the energy expenditure of global companies is more than 40 billion US dollars [2]. People from both academia and industry have devoted a huge amount of effort in developing energy-saving technologies. The latest development includes hardware solutions such as power capping, DVFS, and software solutions such as power-saving scheduling. In the virtualized environment, consolidating virtual machines and switching off idle servers is a common technique for energy saving. VM is the basic unit for virtualization and resource allocation. It is important to study power consumption and power metering of VMs. There are several reasons this issue is important. First, the study power consumption of VMs would lead us to a better understanding about energy consumption in data centers, so that better scheduling algorithm or VM consolidation algorithm can be developed. Second, the study of energy consumption of VMs can lead to more accurate power metering of VMs, so that more reasonable pricing schemes can be employed for the charge of VMs. The current data center systems, such as EC2, charge users according to configuration types and rental time of VMs [3], [4]. VMs with the same configuration and rental time may have totally different amounts of energy consumption due to running different tasks. The amount of energy consumption should be considered in the charge of VMs.

Despite of the importance of power metering of VMs, it is a difficult task to measure the energy consumption accurately for each VM. First, methods for server power metering cannot be directly used for VMs. There is no device that can measure the power consumption of each VM, and the power models of servers cannot be directly applied to measure VM power. Second, the power consumption of each VM is composed of the amount of each hardware resource power consumed by the VM. The power consumption of hardware resources is changing dynamically with the behavior of applications. It is not an easy thing to measure the power consumption of hardware resources. Third, VM power can be affected by other VMs on the same host competing resources together. It is a challenging task to distinguish the portions of each VM in the hardware resource usage. The latest cloud monitoring systems such as GreenCloud [5] and HPiLO [6] can only measure the power consumption in the granularity of server and resource. However, there is no system, so far, that can measure power in the granularity of VMs.

There are some works that are reported in the literature about power metering of VMs. There are two classes of methods for VM power metering: the white box method and the black box method. The former uses information collected inside the VM for modeling, while the latter uses profiling information of each VM from the host level for modeling. Most systems adopt the black box method because collecting information outside of the VMs will not break VM integrity. In those methods, linear model and non-linear model are adopted with PMCs (Performance Monitor Counters) or resource utilization information. The PMC is preferred for power modeling since the counters can well profile the behaviors of the system. For the data training of the models, different benchmarks will be running inside each VM and the modeling information will be collected at the same time. To evaluate the accuracy of VM power metering, absolute error between the estimated total power of the server and the measured power will be used. The variance of error can be used to evaluate stability of the model. There are still many open research issues about VM power metering such as: billing for VM services, power budgeting for data centers, and power saving scheduling. In this paper, we will collect as many power metering methods as possible for cloud data centers, so that more interesting work surrounding VM power can be conducted in the future.

The rest of paper is organized as follows. Section 2 presents the power consumption model of VM power metering. In Section 3, we mainly discuss samplings for modeling. In Section 4, detailed methods for virtual machine power metering are introduced, including modeling methods and architectures. Section 5 discusses accuracy evaluation approaches and benchmarks for VM power metering. In section 6, we will discuss challenges in VM power metering. Section 7 will discuss open research issues advanced by VM power metering. Finally, Section 8 concludes this paper.

II. ENERGY CONSUMPTION MODEL FOR VM POWER METERING

In this section, we will discuss the power metering of VM. The total power consumption of a physical server consists of two components, P_{Static} and P_{VM}. P_{Static} is the fixed power of a server regardless of running VMs or not, and P_{VM} is the dynamic power that is consumed by VMs running on it. Suppose there are n VMs and each of them is denoted by VM_i, $1 \leq i \leq n$. Let P_{VM_i} denote the energy consumed by VM_i. Thus, we have:

$$P_{Total} = P_{Static} + \sum P_{VM} = P_{Static} + \sum_{i}^{n} P_{VM_i}$$

P_{VM_i} can be further decomposed into the power consumption of components such as CPU, memory and IO, denoted by $P_{VM_i}^{CPU}$,$P_{VM_i}^{Mem}$, $P_{VM_i}^{IO}$ respectively. $P_{VM_i}^{IO}$ includes general energy cost of all devices that involve IO operations such as disk and network data transfer. Thus, the power consumption of VM_i is:

$$P_{VM_i} = P_{VM_i}^{CPU} + P_{VM_i}^{Mem} + P_{VM_i}^{IO}$$

P_{VM_i} can also be decomposed into the power consumption of PMCs (Performance Monitor Counters) of the system. Suppose there are m PMCs used for modeling and each of the them is denoted by e_j, $1 \leq j \leq m$. Let $P_{VM_i}^{e_j}$ denote the energy of e_j consumed by VM_i. Thus, we have:

$$P_{VM_i} = P_{VM_i}^{e_1} + P_{VM_i}^{e_2} + \cdots + P_{VM_i}^{e_m}$$

As is clearly shown in Figure 1, we can easily understand the power consumption of each VM in the server power. The power consumption of each resource can be obtained by adding the amount of the resource power consumed by VMs. In practice, how to divide the static power into each VM is decided by the service providers, equally or to the proportion of resource utilization of each VM. In this paper, we mainly focus on VM power metering.

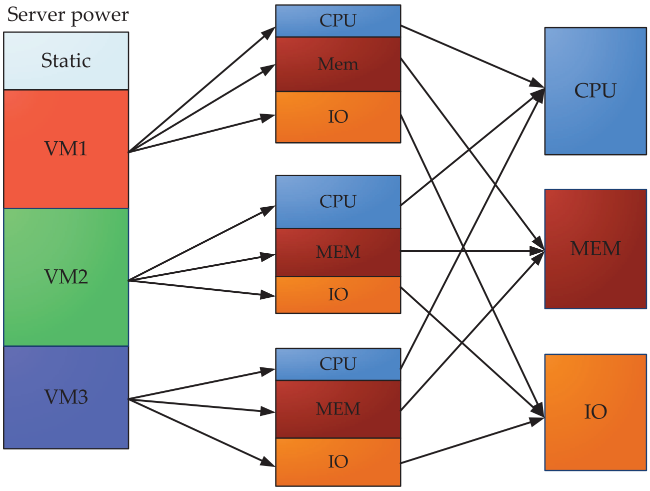


FIGURE 1. The Power Consumption of VMs in the Server.

III. MODELING INFORMATION COLLECTIONS

Virtual machine power metering usually has three steps: information collection, modeling and estimation. Since VM power is closely related to the hardware resources, PMCs and the power consumption of the server, we will discuss power measuring for servers, approaches for modeling information collection and sampling rate in the following:

A. POWER MEASURING FOR SERVERs

The measuring of physical server power is the basis for VM power metering. There are usually two types of methods to measure physical server power. One is using external attached power meter, the other is using internal meter such as power sensors or a special motherboard. The external power meter such as PDU (Power Distributed Unit) has the merit of flexibility. The PDU can be easily attached or detached from the machine without affecting normal operations of the system. One typical PDU is Watts UP series with precision of 0.1watt [7]. Watts UP PRO logs power information inside the flash of PDU and the loggings can be downloaded into the computer when needed. Watts UP ES provides remote access to the power information through network cable in real time.

Another external PDU used in the literatures is Schleifenbauer power meter with precision of 0.01watt [8]. The power information can be obtained through APIs in Python and Perl. Using an externally attached power meter can save a lot of investment in updating current infrastructure for data centers. But it is almost infeasible to apply external PDU on a large scale. On the one hand, it is a complex and costing task to design the plug and wires of PDUs for servers of different types [9]; on the other hand, extra expenses will be incurred to replan data centers for power management. Internal power meter such as power sensors or special motherboard inside the server has the merit of easy management. Power information can be accessed through distributed programming interfaces or command lines, or even GUI. Dell Power Series server is just the type of server with power sensors inside. It can provide comprehensive power information about the CPU; memory; network; motherboard, and fans through Dell Open Manage suite [10]. But it also brings performance degradation of the system if the sampling is too frequent. Despite this, servers with internal power sensors are preferred, due to the convenience in power management for data centers, and the sampling overhead can be controlled by proper rate setting.

B. MODELING INFORMATION COLLECTION

Since there is no such device that can directly measure VM power, information such as resources or events of the system can be converted into power consumption by modeling methods. There is a need to track that information in the granularity of each VM. In this part, we mainly discuss approaches of collecting modeling information for VM power metering.

1) COLLECTING HARDWARE RESOURCE POWER

VM power is usually composed of the portions of hardware resource power that is consumed by the VM. To measure the power consumption of each hardware resource, some works in [11]–[13] use wires to connect self-developed power sensors or registers directly to the motherboards or other components. But those methods are infeasible for large scale data centers due to the complexity and unavailability of the knowledge about architectures of modern servers [14]. As is mentioned above, Dell Power Series servers provide power information in the granularity of the hardware resource. For servers without power sensors inside, the modeling method could be used to estimate the power consumption of each hardware resource in the server.

2) COLLECTING RESOURCES INFORMATION

a: CPU INFORMATION COLLECTION

The power consumption of CPU is affected by too many factors including cache usage; frequency; specific instructions, and subunits of CPU [15]. To estimate the power consumption of CPU of the server and VMs on it, the modeling method can be used by correlating CPU related information to power. The information can be categorized into three types: CPU utilization, PMCs and time slices of processors. Methods for CPU information collection are as follows:

1) Kansal in [15] proposes a method to calculate CPU utilization using active time divided by the total time of the processor for a certain period. To account CPU utilization by each VM, the usages of virtual processors is calculated first by tracking CPU performance counters in Hyper-V, and then transforming the usage of the virtual processor into the utilization of the physical processor. It should be noted that Xentrace can be used to track CPU utilization for each VM on Xen hypervisor.
2) Stoess in [16] collects PMCs of CPU by instrumenting a program in the hypervisor. The program collects PMCs in hypervisor and writes the information into a sharing buffer, which is then mapped into the address space of a monitor program at user-level. Thus, the information of PMCs can be obtained through the monitor program. To account the PMCs for each VM, it can be calculated through deviation of PMCs between the two consequent scheduling switches of the processor.
3) IBM in [17] uses time slices of processors to account the portion of CPU usage by each VM. When vCPU scheduling happens, the time slices is allocated to a VM. Thus, the time slices can be captured from a data structure that can map physical CPU ID to vCPU ID that is corresponding to a VM. The amount of CPU resource used by each VM can be inferred from time slices of the processor used by the VM.

b: MEMORY INFORMATION COLLECTION

The reading and writing throughput of memory has significant effect on memory power. Y.Bao in [18] developed an external instrument that can accurately capture the throughput of memory. Kansal in [15] and Krishnan in [19] proposed a light-weighted method using LLC(Last Level Cache) misses to indirectly estimate the power consumption of memory. Krishnan believes LLC misses can reflect the utilization of memory at different levels. Many processors expose the information as performance counters so that LLC misses can be tracked using tools like Oprofile. LLC misses of each VM can also be obtained in the same way. Kim [20] estimates the power consumption of memory by modeling the number of memory accesses of the server and each VM on it.

c: IO INFORMATION COLLECTION

The power consumption of IO (Input and Output) is usually consumed by devices such as disk and network. Most work in the literature take only disk as the power consuming component for IO, and the network power is so low that it can be ignored. Methods for disk information collection are as follows:

1) Kansal in [15] estimates the power consumption of disk by throughput of disk that can be obtained from hypervisor. For each VM, it can be obtained through performance counters in the Hyper-V virtual storage device and the Hyper-V virtual IDE controller. The

power contribution from varying spinning speeds of the disks is usually not considered, because it is rarely used in data centers.

2) Stoess in [16] proposes that the time to finish an IO request is closely related to disk power. It can be calculated through the size of a request divided by the transfer rate. Note though that the transfer rate is dynamically calculated using the byte size of 50 requests divided by the transfer interval.
3) IBM in [17] proposes a method to get disk and network throughput in the granularity of VM on the Xen hypervisor. The IO requests of disk are mostly initiated by VM, so the disk throughput information for each VM can be easily obtained from the requests captured at the hypervisor level. Similarly, the writing throughput of network can be obtained in this way. But the reading throughput of network cannot be captured from hypervisor, since the requests are initiated by senders. It can be obtained by monitoring the shared memory ring, which records the information of reading requests with corresponding VM ID. Thus, we can obtain throughput of both disk and network for each VM on Xen hypervisor.

3) TOOLS FOR INFORMATION COLLECTION

To obtain modeling information, tools provided by Linux OSes or developed by other organizations could be used. System events and register values can be collected through PMC tools like Oprofile [21], Pfmon [22] and Perf-suite [23]. All of those tools can profile the system in the granularity of process. Thus, VM can be profiled since it is running as a process on the host. Chen Q in [24] uses Ganglia [25], a cluster monitoring tool for large data centers, to collect modeling information. Monitoring tools provided by Linux such as iostat and sar in toolkit sysstat can be used for modeling information collection. Some other event information can be obtained by reading Linux files like /proc and /sys. In addition, specialized tools are developed to collect modeling information on Xen [26] KVM [27] and VMware. For example, Xentop [28], XenMon [29] and XenoProf [30] are designed to collect information of host and that of each VM in Xen platform, an enhanced perf [31], [32] is for KVM, and ReTrace [33] for VMware. With those tools, collecting modeling information for VM power metering is no longer a challenge for virtualized platform.

C. SAMPLING RATE

Sampling is important for modeling information collection. Too frequent sampling will affect the normal running of the server, and the accuracy will be lowered otherwise. The goal of sampling rate setting is to keep high accuracy with low overhead. Most researchers empirically set sampling rate to 1 or 2 seconds. McCullough in [34] holds that 1 second is the right choice and the accuracy cannot be enhanced obviously by adjusting the interval. While in [9], 2 seconds sampling is proved to be suitable through experiments on power overheads of different sampling intervals. In addition, Chen Q in [24] proposes that sampling interval can be adjusted to be 5 seconds for stable applications, and 1 second for applications with dynamic workload and power. In summary, it is reasonable to set and adjust sampling rate flexibly according to real applications.

IV. METHODS FOR VM POWER METERING

Based on the sampling information, we will discuss detailed methods for VM power metering including basic procedures, white box and black box methods, and finally mathematic methods in modeling. Virtual machine power metering usually includes three steps as follows:

1) **Information collection:** collecting necessary information for modeling such as the power consumption of server, resources utilization and PMCs that can be obtained by approaches mentioned above for the server and each VM.
2) **Modeling:** build a proper mathematic model using the most power-related resources or PMCs as variables, and then train the model using samples collected to generate a set of parameters.
3) **Estimation:** calculate the power consumption of each VM by taking the information of each VM into the model with latest parameters.

VM power metering can be classified into two categories: white-box method and black-box method. So the following will discuss the two methods in detail:

A. WHITE BOX METHOD

White box method is also called pitching-in or proxy method in VM power metering. A running proxy program is inserted into each VM to collect resources utilization or PMC events of the VM for power modeling. The architecture of white box method is in Figure 2.

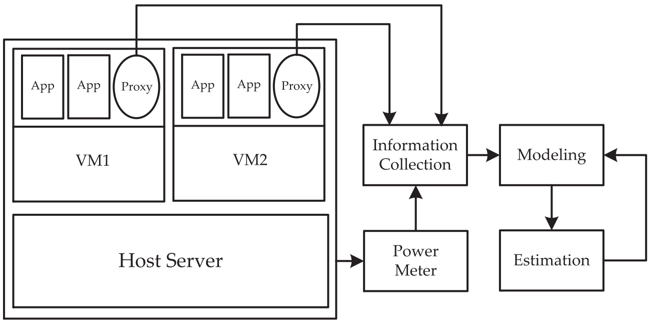


FIGURE 2. Architecture of White Box Method.

In the architecture, there are several VMs running on the host with each VM executing several applications inside. The proxy program in each VM is responsible for collecting resource utilization of the virtual machine. Then the collection module gathers information from the proxy of each VM and power meter. The modeling module will train the collected data and generate a set of model parameters. Finally, we use the model to estimate the power consumption of each VM. The estimation module will feedback to the modeling module when errors exceed a certain threshold, then the modeling module re-trains the latest samples to update parameters of the model.

Based on the architecture, Li in [35] proposes a white box method to measure the power consumption of each VM. Suppose there are n VMs on the server, and each of them is denoted by VM_i, $1 \leq i \leq n$. The author proposes server power model as:

$$P_{Server} = P_{static} + \alpha \sum_{i}^{n} U_{VM_i}^{CPU} + \beta \sum_{i}^{n} U_{VM_i}^{Mem} + \gamma \sum_{i}^{n} U_{VM_i}^{IO} + ne,$$

where P_{Server} is the measured power consumption of the server, P_{static} is the fixed power regardless of running VMs or not. The utilization of CPU, memory and disk IO of the VM_i are represented as $U_{VM_i}^{CPU}$, $U_{VM_i}^{Mem}$, $U_{VM_i}^{IO}$ respectively. e is defined as adjusting bias for each VM. α, β, and γ are the parameters to train.

Li in [35] first runs benchmarks inside each VM. Meanwhile, modeling information is collected by a running proxy inside each VM and the server power is collected through an external attached PDU. The sampling process is executed for at least 100 times. Then, Least Squares are adopted to train the model, generating a set of parameters. Thus, the model for the power consumption of server is built. In the work, the same parameters are used for estimating the power consumption of each VM, so each VM power is represented as:

$$P_{VM_i} = \alpha U_{VM_i}^{CPU} + \beta U_{VM_i}^{Mem} + \gamma U_{VM_i}^{IO} + e.$$

To make the parameters suitable to various applications, Li proposes to train extra sets of parameters corresponding to different utilization segments of certain resources. To explain the segmentation idea, define S_{CPU} as the total CPU utilization of VMs. We divide S_{CPU} into three segments: $S1$, $S2$, and $S3$, corresponding to low, middle, and high utilization of CPU respectively. If there are three VMs running on a server, S_{CPU} is between 0 to 300%. Suppose S_{CPU} is divided into three segments as $0 \leq S1 \leq 120\%$, $120 \leq S2 \leq 240\%$, $240 \leq S3 \leq 300\%$ to represent CPU utilization of system in high, middle and low state respectively. In training, a separate set of parameters will be generated for each segment. In estimation, the parameters corresponding to segment $S2$ will be used when S_{CPU} is 105%. Besides, other resources such as memory can be added, so there will be more sets of parameters corresponding to the segments of multi-resources. The advantage of multi-sets of parameters is that it will reduce the impact of fluctuation when system resource utilization is at peak or bottom.

In summary, the white box method is simple in implementation, but it can be used only in the private cloud where proxy programs are allowed to be inserted into VMs. For the public cloud such as Amazon EC2, the white box method is almost infeasible due to the security and integrity worries from users. Besides, the resources usage information collected inside each VM cannot objectively reflect the usage of hardware resources by the VM. So, the black box method that collects events for each VM at host level is needed, as is to be discussed in the following.

B. BLACK BOX METHOD

The black box method collects information of each VM at host level. The architecture of black box method is similar to that of white box method, as is shown in Figure 3. The difference lies in that VM profiling information such as PMCs are collected outside VMs at hypervisor level. A typical example of this architecture is Xen virtualization platform and we can use Xenoprofile as tool to collect events of each VM on it.

The black box method for VM power metering is based on the power model of the physical server. The power consumption of each VM is calculated by taking resources used or PMCs of each VM into the server power model. The key is to distinguish and account for the resources or PMCs for each VM. The accuracy of the power model for physical server will directly affect the result of VM power metering. PMCs are the counters that record the accumulation values of the registers or events of the system. PMCs are always used to profile the power consumption of system, applications and VMs. There are two categories of VM power metering methods using PMCs: component-based models with PMCs representing each component and pure PMC based models.

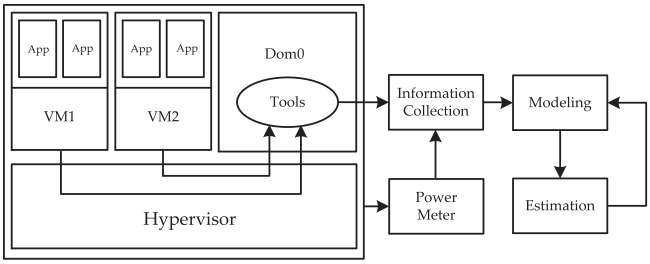


FIGURE 3. Architecture of Black Box Method.

1) COMPONENT BASED VM POWER ESTIMATION

The power consumption of a physical server consists of static power and dynamic power as is mentioned in section 3. The dynamic power is composed of the power consumption by CPU, memory, IO and so on, denoted as P_{CPU}, P_{Mem}, P_{IO}, etc. respectively. Thus, the total power can be expressed:

$$P_{Total} = P_{Static} + P_{CPU} + P_{Mem} + P_{IO} + \cdots .$$

Kansal in [15] estimates the power consumption of server by correlating information of components like CPU, memory and IO to power. Thus the server power is:

$$P_{Total} = P_{Static} + \alpha u_{cpu} + \beta N_{LLCM}(T) + \gamma b_{io},$$

where u_{cpu} is the utilization of CPU, $N_{LLCM}(T)$ is the last level cache missing pages, and b_{io} is the transfer time of IO. The power consumption of IO only includes disk power, since the network power is so small that can be neglected. Disk power is estimated by the total throughput of reading and writing bytes, since there is no difference in the power consumption between the reading and writing of disk. Based on server power model, the power consumption of *VM i* can be expressed:

$$P_{VM_i} = \alpha u_{VM_i}^{CPU} + \beta N_{VM_i}^{LLCM} + \gamma b_{VM_i}^{IO},$$

where $u_{VM_i}^{CPU}$ is the CPU utilization used by VM i, $N_{VM_i}^{LLCM}$ is the LLC of VM i, and $b_{VM_i}^{IO}$ is the disk transfer bytes of VM i.

Bohra in [36] uses the PCA method to analyze the component information. He finds the correlation between the components: $\{CPU, Cache\}$ and $\{Disk, DRAM\}$, so the power model is:

$$\begin{aligned} P_{Total} &= \alpha(a_1 + a_2 P_{CPU} + a_3 P_{Cache}) \\ &\quad + \beta(a_4 + a_5 P_{DRAM} + a_6 P_{Disk}) \end{aligned}$$

This formula can be transformed into a common linear one as follows:

$$P_{Total} = \alpha P_{CPU} + \beta P_{Cache} + \gamma P_{DRAM} + \omega P_{Disk} + \delta,$$

where δ is the adjusting bias for the model. Based on the model of server power, VM power can be expressed:

$$P_{VM_i} = \alpha P_{VM_i}^{CPU} + \beta P_{VM_i}^{Cache} + \gamma P_{VM_i}^{DRAM} + \omega P_{VM_i}^{Disk}$$

Bohra builds the model using PMCs such as CPU_CLK_UNHALTED, DRAM_ACCESSES, INSTRUCTION_CACHE_FETCHES and DATA_CACHE_FETCHES to represent the component states of CPU, memory and caches respectively, and disk information is collected separately using other tools.

Krishnan in [19] argues that using only CPU usage cannot fully profile system power, and the power consumption of memory should be taken into consideration. He uses two components for modeling: CPU and memory. He believes disk and network power is small and static, therefore, CPU and memory for modeling is enough. In the modeling, the instructions retired are considered for CPU. For memory, the difference of the power consumption between different memory hierarchies of memory is considered. Since the last level cache (LLC) hits consumes more energy than the former cache, LLC is used for memory. The author proved the overhead and the feasibility of the model using the information from the two components:

$$P_{Total} = \alpha P_{CPU} + \beta P_{Cache} = \alpha N_{ins\_ret} + \beta N_{LLC},$$

where N_{ins_ret} and N_{LLC} denotes the number of instructions and the number of LLC missing, respectively.

Cherkasova in [37] proves much energy will be consumed by IO virtualization due to its need for CPU processing. In consideration of IO power, Chen in [24] proposes a modified model using CPU and Hard disk:

$$P_{Total} = P_{Static} + \alpha P_{CPU} + \beta P_{HDD},$$

where P_{HDD} is the power consumption of hardware.

In modeling, the number of cores may have effect on the power consumption of the server. Some works in the literature propose that the power consumption of the server has linear relationship with the number of active processors of the system. So Kim in [20] gives a power model as follows:

$$P_{Total} = \sum_{i=1}^{n} (\alpha N_i + \beta M_i - \gamma S_t),$$

where N_i is the number of the retired instructions, M_i is the number of memory accesses, and S_t is the running core during time t. The coefficients can be obtained by multi-regression method. Bertran in [38] and [39] gives another linear model considering the number of active cores. He uses several PMCs to represent CPU and memory events. The model is a conditional function considering the conditions when there is one processor core running or two cores running.

One thing in common is that all of the works mentioned above adopt linear models. The differences lie in the selections of components for modeling and PMCs to represent each component in the model. The linear model is the most widely used method in the estimation of power consumption. Liu in [40] built a power model for VM migration. He proved that the power consumption of VM migration is in linear relationship with transmitting volumes, not with data transmitting rate. The commonly used mathematic methods in the linear model are least squares and multiple linear regression. However, linear models suit only the condition that the features are independent of each other [34], and the accuracy is declined if the components or events are correlated with each other. Many non-linear models are proposed as discussed in the following:

Versick in [41] and [42] proposes a polynomial model using the information of the CPU, hard disk, and network interface card to express the dynamic power of server. The estimation error can be reduced to 3.1% when the order is set to be six. The power consumption of CPU, hard disk and network card are denoted as P_{CPU}, $P_{Network}$, and P_{NIC}, respectively. The model is:

$$\begin{aligned} P_{Total} &= P_{CPU} + P_{HDD} + P_{NIC} + P_{Static} \\ &= a_{1,C}x_C + a_{2,C}x_C^2 + \cdots + a_{m,C}x_C^m + b_C \\ &\quad + a_{1,H}x_H + a_{2,H}x_H^2 + \cdots + a_{m,H}x_H^m + b_H \\ &\quad + a_{1,N}x_N + a_{2,N}x_N^2 + \cdots + a_{m,N}x_N^m + b_N \\ &\quad + P_{Static}, \end{aligned}$$

where $a_{i,j}$ is the parameters of model, x is the variable that depicts the information of the components. Different from the former methods, Versick trains the parameters for each component one after another. The first step is to measure the static server power. Then, use the CPU intensive benchmark to train the parameters of polynomial for the CPU only. After this, use the disk IO benchmark to train parameters of polynomial for the disk. Note though that the power for IO modeling is the deviation of the total power subtracting static power and estimated CPU power. Thus, the same method is used to train parameters for NIC.

Xiao in [43] and [44] builds the dynamic power of server model using these four components: CPU; memory; disk, and network IO. Thus, server model is:

$$P_{Total} = P_{Static} + P_{CPU} + P_{Memory} + P_{Disk} + P_{IO}$$

Xiao gives the model for the power consumption of each component expressed by PMCs including uOps (micro-operations); Halt; LLC (Last Level Cache); TLB (Translation Look-aside Buffer); FSB(Front-Side Bus); Interrupt, and DMA. All the information of the models is collected for a certain time. The models for each component are:

$$\begin{aligned} P_{CPU} &= \mu Ops - Halt^2 \\ P_{Memory} &= LLC + TLB + FSB \\ P_{Disk} &= Interrupt + DMA^3 \\ P_{IO} &= Interrupt + DMA, \end{aligned}$$

Based on this model, the power consumption of each VM can be estimated by adding the subparts of each component power by the VM.

Wen in [45] proposes a LUT (Look Up Table) method to store the power consumption corresponding to different CPU and memory states, expressed as LUT[CPU][LLC]. Each VM has a single LUT. The row of LUT is CPU utilization and the column is LLC for the VM. The estimation will be more accurate when more data is collected. For samples with the same CPU and LLC information, the power in the table is set to be the median of the values. For LUT without any power value, it can be inferred from the nearby value. LUT method is flexible and easy to implement. It avoids the inaccuracy of modeling methods when the workload of different types changes dynamically. But it needs large memory to store LUT for each VM, especially for tables with more dimensions if other resource information is added.

For the selection of modeling components, the power consumption of net card with high throughout and fans with varying speed can also be considered. In [46], Ma takes the power consumption of fans into consideration for VM power metering. He divides the power consumption of the fans into each VM according to the proportion of VM power in the whole dynamic server power. In [47], Quesnel improves the exsisted power dividing method for the idle part for each server. He proposes that the idle power should be completely proportionated to the VM when there is only one VM running on the host with one resource usage 100%. In other conditions, the idle power is divided into each VM according to the proportions of the resources usage.

Most of the literature adopts the component based method using the linear model to measure VM power. However, the selection for the events or PMCs for modeling is based on experience without any theoretical analysis. Pure PMC based models are proposed to complement the insufficiencies of component based models.

2) PURE PMC BASED VM POWER ESTIMATION

Pure PMC based models use only PMCs for modeling, avoiding the inaccuracy incurred by empirical PMCs selection for the components. The PCA is excellent in reducing the dimension of raw data and selecting the most influenced independent components [48]. Yang in [9] uses the PCA method to filter out the most power-related PMCs. Yang in [49] also uses ε-SVR [50] model to train the samples and estimate the power consumption for each VM. Since the model is so complicated, LIBSVM [51] package is used to complete the calculation of the model. The drawback of PMCs is that they always have high correlation with the under architecture, and the number of PMCs is so large that it may cost more time to train an accurate set of parameters with architectural knowledge [52]. In spite of this, the PMC based method is preferred in the study. We can process PMC traces offline to enhance accuracy. The mathematical methods are very important in the modeling of VM power metering. The commonly used linear models in power modeling are multiple linear regression, Mantis, and Lasso Regression [53], [54]. For non-linear models, Polynomial with exponential and Lasso [55], gaussian mixture models [56], and SVR (Supported Vector Regression) are the widely used. Which type of model is better, the simple linear model or the complex non-linear? McCullough, in paper [34], holds that the accuracy of power may not be improved by increasing model complexity.

In discussion, the linear model is simple but easy in implementation with low overhead. It is wise to make a tradeoff between accuracy and complexity in modeling methods selection, in consideration of real application requirements. For future research in the methods for VM power metering, one direction is to use the non-linear multiple regression method to improve the accuracy of current methods. Another direction is to process the sampling data, using machine learning methods, to choose the most suitable events for modeling. Besides, machine learning algorithms can be a good choice when the sampling data does not fit in any prevalently used linear or polynomial model.

V. EVALUATION OF VM POWER METERING METHODS

For the white box and black box method in VM power metering, the accuracy evaluation approach is the same, using error between estimated server power and the measured server power. The error can be used as a threshold to decide when to update model parameters. In this section, we will discuss how to evaluate the accuracy of VM power modeling and summarize the commonly used benchmarks.

A. ACCURACY OF METERING METHODS

A ground truth is proposed in [15] that the error between estimated power and measured power of server is in an acceptable range if the modeling method is accurate enough.

TABLE 1. Benchmarks and Descriptions.

Category	Benchmark	Function Description
Processor	SPEC CPU2006 [58]	A tool suite to test CPU performance through a wide range of CPU-intensive workload from real user applications.
	Dhrystones [59]	A benchmark from Unix Benchmark suite for testing the performance of processor.
	BYTEmark [60]	Providing integer or floating points tests for CPU, memory and cache.
Memory	SPEC OMP2001 [61]	Benchmarks for measuring shared-memory parallel processing and metrics for energy consumption.
	Cachebench [62]	A program testing memory hierarchy performance
Disk	Bonnie++ [63]	A tool for testing disk and file system IO performance
	IOzone [64]	A benchmark for testing the reading and writing performance for file system.
	IOmeter [65]	Measuring the IO subcomponents performance for single server and clusters
Network	Netperf [66] and iPerf	Benchmark for testing network performance
Paralell	NAS-NPB [67]	A benchmark developed by NASA for parallel computing evaluation
System Performance	Linpack [68]	Benchmarks to evaluate the system performance using scientific computation.
	StressAppTest [69]	A high load benchmark to test hardware devices performance

Since the power consumption of VM cannot be measured by any device, the accuracy evaluation of VM power is using the error between measured power of sever and the estimated server power, denoted as $P_{measured}$ and $P_{estimated}$ respectively. To evaluate the accuracy of the models, Li in [35] uses relative error as metric, denoted as Error:

$$Error = \frac{|P_{measured} - P_{estimated}|}{P_{measure}}$$

To avoid random fluctuation, it is advised that the same experiments should be tested at least 10 or more times, and then average error value is taken as credible [34], [57]. So the evaluation formula is:

$$Error = \frac{1}{n}\sum_{i=1}^{n}|\frac{P_{measured} - P_{estimated}}{P_{measure}}|$$

where n is the experiment times of the same benchmark.

The author in [57] also proposes the stability as an external evaluation metric for the goodness of power model, and it can be expressed in the following:

$$\begin{aligned} E_{Stability} &= Variance(Error) \\ &= Variance(\frac{1}{n}\sum_{i=1}^{n}|\frac{P_{measure} - P_{predict}}{P_{measure}}|) \end{aligned}$$

To verify the model, K-cross validation method is advised [24], [36]. For example, the whole sampling set can be randomized, and then select the first half as training data and the second part as testing data. Then repeat the process for several times, so the model can be objectively validated by mean error and variance.

The evaluation method has been well used in the estimation of physical server power. For VM power metering using the black box method, this evaluating method is still sound enough. The calculation of the power consumption for each VM is based on the power model of physical server. Thus, the resource or event information for calculating VM power is just the subset of the information collected in the physical server. The VM power metering problem is transformed to build an accurate server model and how to properly divide the power into the VMs running on it. Therefore, the accuracy of VM power is mainly decided by the accuracy of the power model for the physical server. For the white box method, this system of evaluation may not be accurate enough. For one thing, the calculation of VM power using the white box method is not based on the model of server power; for another, the minus error and positive error from different VMs may offset each other. The evaluation for the white box method needs further study in the future.

B. BENCHMARKS

To build proper models and to evaluate their accuracy, we must run multiple VMs on the same server, each running a certain benchmark inside. Usually, benchmarks can be designed according to real application scenarios. To evaluate the model, various benchmarks are used to verify the effectiveness of the methods. In the following, the most commonly used benchmarks are summarized in Table 1.

VI. CHALLENGES FOR VM POWER METERING

Many techniques have been developed in measuring the power consumption of server such as PDU (power distribution unit), an external power meter for older servers and inner power sensors for the motherboard of newer servers. However, there is no such devices that can directly measure the power consumption of VMs. Therefore, software methods using the mathematical models have been proposed: the resource based models and the PMC based models. There are some challenges in the implementation of the two types of models:

1) Distinguishing the activities of each VM in resource utilization or PMC changes, so that we can quantify the contribution of each VM to the power of each resource.
2) Determining what resources or PMCs should be considered for the measuring of VM power.
3) Deciding the type of mathematic models to be used for PMC based model.

The first challenge has been addressed through methods mentioned in Section 3. Unfortunately, the inaccuracies and instabilities of current methods are mainly caused by the second and the third challenge, which remain unsolved. Future research will be conducted to overcome those two challenges, so as to get more accurate power consumption for the VMs. To the best of our knowledge, machine learning methods like regression tree could be used to replace the linear models, because it will automatically divide the values of the resource in different dimensions into segments, and then train each generated segment leaf using linear model. Therefore, VM power metering using machine learning models is the future research direction. It could be used to automatically select key features(here refer to the resource utilization or changing values of the events). Features with similar characters could be clustered, and a separate model could be trained using data from the cluster.

VII. OPEN RESEARCH ISSUES

VM power metering is an important and emerging topic. There are many research issues, yet, to be investigated. Some typical issues include VM service billing, power budgeting, and energy saving scheduling.

A. VM SERVICE BILLING

VM power is the basic unit for virtualized data centers, so future data centers will improve the monitoring system with the visibility of VM power. On the one hand, it will be helpful for us to understand the power consumption of data centers in a finer granularity; on the other hand, reasonable billing for VM services can be made. The traditional billing is based on the configuration and running time of VMs, but the resource usage can be different for VMs with same configuration and running time. Future data centers will make full use of VM power metering technology to improve billing schemes in VM services, especially for services like Amazon EC2.

B. POWER BUDGETING

Power budgeting is playing an important role in modern data centers. To support more servers running in the data center without breaking the upper bound power, power capping technology is introduced. The problem is that too many CPU-intensive VMs, consolidated to the same server, may intrigue DVFS of the server so that all the VMs will suffer the degradation of the performance of server. This breaks the isolation of each VM indirectly, and prolongs the running time of tasks, reducing energy efficiency. Therefore, VM consolidation cannot always save energy without budgeting VM power. For modern data centers with power capping servers, there is need to budget power in different granularities. In VM granularity, the users can decide how much energy their VMs will use. They can budget the cost of applications running inside their VMs. In server granularity, VM consolidation will be reasonably designed so that the resource usage and the power efficiency can be enhanced without breaking the SLA or QoS of the servers. In data center granularity, more servers can be running at the same time without exceeding the peak power of data center.

C. POWER SAVING SCHEDULING

Future green data centers cannot continue without power-saving mechanisms. Many scheduling polices are studied in VM migration and consolidation with idle servers powered off. Those scheduling methods only consider the constraints of resources in deployment. In fact, the energy of each VM should also be considered so as to design power-saving scheduling for data centers. VM power metering provides an opportunity to optimize the already power-aware algorithms to save more energy cost in virtualized cloud data centers. VM power metering is of great significance for the future green cloud data centers, providing us opportunities to study new techniques for cloud data centers.

VIII. CONCLUSION

In this paper, we conducted a comprehensive investigation regarding issues of VM power metering. We mainly focused on estimating VM power at the software level. Details of the implementation for VM power metering has been discussed including: tools for information collection, modeling methods, and estimation. We have found that black box method using PMC information for modeling is more preferred, because it can well profile the system power without violating the integrity of each VM. Through analyzing the efficiencies and deficiencies of linear models and non-linear models, we believe that machine learning methods could be used to enhance the accuracy of current method in VM power metering. For the evaluation metrics like error and stability, we have found they are suitable for evaluating black box methods only. For white box method, future work should be done on the evaluation for the white box method, since current metrics are not accurate enough for the white box method. Based on the technique of VM power metering, many more interesting topics like power budgeting, fair billing and power-aware scheduling will be studied for the future green cloud data centers.

ACKNOWLEDGMENT

The authors would like to thank Joyce Webb Gail for her constructive suggestions on the revision of our final paper. The authors would also like to thank all the editors and anonymous reviewers for their comments and feedback that improved our paper.

REFERENCES

[1] P. Kurp, "Green computing," *Commun. ACM*, vol. 51, no. 10, pp. 11–13, 2008.

[2] P. Ranganathan, "Recipe for efficiency: Principles of power-aware computing," *Commun. ACM*, vol. 53, no. 4, pp. 60–67, 2010.

[3] E. Elmroth, F. G. Marquez, D. Henriksson, and D. P. Ferrera, "Accounting and billing for federated cloud infrastructures," in *Proc. 8th Int. Conf. Grid Cooperat. Comput. (GCC)*, 2009, pp. 268–275.

[4] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[5] L. Liu *et al.*, "GreenCloud: A new architecture for green data center," in *Proc. 6th Int. Conf. Ind. Session Auton. Comput. Commun. Ind. Session*, 2009, pp. 29–38.

[6] *HPiLO*. [Online]. Available: http://h17007.www1.hp.com/us/en/enterprise/servers/management/ilo/#.U4Wk4uM_jKc, accessed May 29, 2014.

[7] *WattsUp Meter*. [Online]. Available: https://www.wattsupmeters.com/secure/index.php, accessed May 29, 2014.

[8] *Public APIs of Schleifenbauer PDU*. [Online]. Available: http://sdc.sourceforge.net/index.htm, accessed May 29, 2014.

[9] H. Yang, Q. Zhao, Z. Luan, and D. Qian, "iMeter: An integrated VM power model based on performance profiling," *Future Generat. Comput. Syst.*, vol. 36, pp. 267–286, Jul. 2014.

[10] J. Jenne, V. Nijhawan, R. Hormuth. (2009). *Dell Energy Smart Architecture (Desa) for 11g Rack and Tower Servers*. [Online]. Available: http://www.dell.com

[11] W. Dargie and A. Schill, "Analysis of the power and hardware resource consumption of servers under different load balancing policies," in *Proc. IEEE 5th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2012, pp. 772–778.

[12] E. M. Elnozahy, M. Kistler, and R. Rajamony, "Energy-efficient server clusters," in *Proc. 2nd Int. Conf. Power-Aware Comput. Syst.*, 2003, pp. 179–197.

[13] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-system power analysis and modeling for server environments," in *Proc. IEEE Int. Symp. Comput. Archit.*, 2006, pp. 1–4.

[14] C. Mobius, W. Dargie, and A. Schill, "Power consumption estimation models for processors, virtual machines, and servers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 6, Jun. 2014.

[15] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *Proc. 1st ACM Symp. Cloud Comput.*, 2010, pp. 39–50.

[16] J. Stoess, C. Lang, and F. Bellosa, "Energy management for hypervisor-based virtual machines," in *Proc. USENIX Annu. Tech. Conf.*, 2007, pp. 1–14.

[17] Y. Chen, L. Li, L. Liu, H. Wang, and Y. Zhou, "Method and apparatus for estimating virtual machine energy consumption," U.S. Patent 13 596 612, Aug. 28, 2012.

[18] Y. Bao *et al.*, "HMTT: A platform independent full-system memory trace monitoring system," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 1, pp. 229–240, 2008.

[19] B. Krishnan, H. Amur, A. Gavrilovska, and K. Schwan, "VM power metering: Feasibility and challenges," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 3, pp. 56–60, 2010.

[20] N. Kim, J. Cho, and E. Seo, "Energy-based accounting and scheduling of virtual machines in a cloud system," in *Proc. IEEE/ACM Int. Conf. Green Comput. Commun. (GreenCom)*, Aug. 2011, pp. 176–181.

[21] J. Levon and P. Elie, *OProfile: A System Profiler for Linux*, 2004. [Online]. Available: http://perfmon2.sourceforge.net, accessed May 29, 2014.

[22] *Perfmon2*. [Online]. Available: http://perfmon2.sourceforge.net, accessed May 29, 2014.

[23] R. Kufrin, "PerfSuite: An accessible, open source performance analysis environment for Linux," in *Proc. 6th Int. Conf. Linux Clusters: HPC Revol.*, vol. 151. 2005, p. 5.

[24] Q. Chen, P. Grosso, K. Van Der Veldt, C. de Laat, R. Hofman, and H. Bal, "Profiling energy consumption of VMs for green cloud computing," in *Proc. IEEE 9th Int. Conf. Depend., Auton. Secure Comput. (DASC)*, Dec. 2011, pp. 768–775.

[25] *Ganglia Homepage*. [Online]. Available: http://ganglia.sourceforge.net/, accessed May 29, 2014.

[26] P. Barham *et al.*, "Xen and the art of virtualization," *ACM SIGOPS Operat. Syst. Rev.*, vol. 37, no. 5, pp. 164–177, 2003.

[27] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "KVM: The Linux virtual machine monitor," in *Proc. Linux Symp.*, vol. 1. 2007, pp. 225–230.

[28] *Xentop*. [Online]. Available: http://www.xen.org, accessed May 29, 2014.

[29] D. Gupta, L. Cherkasova, R. Gardner, and A. Vahdat, "Enforcing performance isolation across virtual machines in Xen," in *Proc. Int. Conf. Middleware*, 2006, pp. 342–362.

[30] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel, "Diagnosing performance overheads in the Xen virtual machine environment," in *Proc. 1st ACM/USENIX Int. Conf. Virtual Execution Environ.*, 2005, pp. 13–23.

[31] Y. Zhang. (2010). *Enhance Perf to Collect KVM Guest os Statistics from Host Side*. [Online]. Available: http://lwn.net/Articles/378778/

[32] P. Patil, P. Kulkarni, and U. Bellur, "VirtPerf: A performance profiling tool for virtualized environments," in *Proc. IEEE Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2011, pp. 57–64.

[33] M. Sheldon and G. Weissman, "ReTrace: Collecting execution trace with virtual machine deterministic replay," in *Proc. 3rd Annu. Workshop Modeling, Benchmarking Simulation (MoBS)*, 2007, pp. 1–8.

[34] J. C. McCullough, Y. Agarwal, J. Chandrashekar, S. Kuppuswamy, A. C. Snoeren, and R. K. Gupta, "Evaluating the effectiveness of model-based power characterization," in *Proc. USENIX Annu. Tech. Conf.*, 2011, p. 12.

[35] Y. Li, Y. Wang, B. Yin, and L. Guan, "An online power metering model for cloud environment," in *Proc. 11th IEEE Int. Symp. Netw. Comput. Appl. (NCA)*, Aug. 2012, pp. 175–180.

[36] A. E. H. Bohra and V. Chaudhary, "VMeter: Power modelling for virtualized clouds," in *Proc. IEEE Int. Symp. Parallel Distrib. Process., Workshops PhD Forum (IPDPSW)*, Apr. 2010, pp. 1–8.

[37] L. Cherkasova and R. Gardner, "Measuring CPU overhead for I/O processing in the Xen virtual machine monitor," in *Proc. USENIX Annu. Tech. Conf.*, vol. 50. 2005, p. 24.

[38] R. Bertran *et al.*, "Accurate energy accounting for shared virtualized environments using PMC-based power modeling techniques," in *Proc. 11th IEEE/ACM Int. Conf. Grid Comput. (GRID)*, Oct. 2010, pp. 1–8.

[39] R. Bertran *et al.*, "Energy accounting for shared virtualized environments under DVFS using PMC-based power models," *Future Generat. Comput. Syst.*, vol. 28, no. 2, pp. 457–468, 2012.

[40] H. Liu, H. Jin, C.-Z. Xu, and X. Liao, "Performance and energy modeling for live migration of virtual machines," *Cluster Comput.*, vol. 16, no. 2, pp. 249–264, 2013.

[41] D. Versick, I. Waßmann, and D. Tavangarian, "Power consumption estimation of CPU and peripheral components in virtual machines," *ACM SIGAPP Appl. Comput. Rev.*, vol. 13, no. 3, pp. 17–25, 2013.

[42] I. Waßmann, D. Versick, and D. Tavangarian, "Energy consumption estimation of virtual machines," in *Proc. 28th Annu. ACM Symp. Appl. Comput.*, 2013, pp. 1151–1156.

[43] P. Xiao, Z. Hu, D. Liu, G. Yan, and X. Qu, "Virtual machine power measuring technique with bounded error in cloud environments," *J. Netw. Comput. Appl.*, vol. 36, no. 2, pp. 818–828, 2013.

[44] X. Peng and Z. Sai, "A low-cost power measuring technique for virtual machine in cloud environments," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 3, p. 69, 2013.

[45] Z. Jiang, C. Lu, Y. Cai, Z. Jiang, and C. Ma, "VPower: Metering power consumption of VM," in *Proc. 4th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, May 2013, pp. 483–486.

[46] C. Ma *et al.*, "Virtual machine power metering and its applications," in *Proc. IEEE Global High Tech Congr. Electron. (GHTCE)*, Nov. 2013, pp. 153–156.

[47] F. Quesnel, H. K. Mehta, and J.-M. Menaud, "Estimating the power consumption of an idle virtual machine," in *Proc. IEEE Int. Conf. IEEE Cyber, Phys. Social Comput. Green Comput. Commun. (GreenCom), IEEE Internet Things (iThings/CPSCom)*, Aug. 2013, pp. 268–275.

[48] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscip. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433–459, 2010.

[49] H. Yang, Q. Zhao, Z. Luan, and D. Qian, "UVMPM: A unitary approach for VM power metering based on performance profiling," in *Proc. 41st Int. Conf. Parallel Process. Workshops (ICPPW)*, Sep. 2012, pp. 614–615.

[50] V. Vapnik, S. E. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 1997, pp. 281–287.

[51] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.

[52] W. L. Bircher and L. K. John, "Complete system power estimation using processor performance events," *IEEE Trans. Comput.*, vol. 61, no. 4, pp. 563–577, Apr. 2012.

[53] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-system power analysis and modeling for server environments," in *Proc. IEEE Int. Symp. Comput. Archit.*, Jun. 2006, pp. 70–77.

[54] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A comparison of high-level full-system power models," in *Proc. HotPower*, vol. 8. 2008, p. 3.

[55] W. L. Bircher and L. K. John, "Complete system power estimation: A trickle-down approach based on performance events," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, Apr. 2007, pp. 158–168.

[56] G. Dhiman, K. Mihic, and T. Rosing, "A system for online power prediction in virtualized environments using Gaussian mixture models," in *Proc. 47th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2010, pp. 807–812.

[57] W. Chengjian, L. Xiang, Y. Yang, F. Ni, and Y. Mu, "System power model and virtual machine power metering for cloud computing pricing," in *Proc. 3rd Int. Conf. Intell. Syst. Design Eng. Appl. (ISDEA)*, Jan. 2013, pp. 1379–1382.

[58] J. L. Henning, "SPEC CPU2006 benchmark descriptions," *ACM SIGARCH Comput. Archit. News*, vol. 34, no. 4, pp. 1–17, 2006.

[59] *Unixbench Project Homepage*. [Online]. Available: http://code.google.com/p/byte-unixbench/, accessed May 29, 2014.

[60] *Bytemark*. [Online]. Available: http://www.tux.org/~mayer/linux/bmark.html, accessed May 29, 2014.

[61] V. Aslot and R. Eigenmann, "Performance characteristics of the SPEC OMP2001 benchmarks," *ACM SIGARCH Comput. Archit. News*, vol. 29, no. 5, pp. 31–40, 2001.

[62] *Cachebench: Benchmark for Memory Sub-System*. [Online]. Available: http://icl.cs.utk.edu/projects/llcbench/cachebench.html, accessed May 29, 2014.

[63] *Bonnie++: Filesystem Performance Benchmark*. [Online]. Available: http://www.coker.com.au/bonnie++/, accessed May 29, 2014.

[64] D. C. W. Norcott. *Iozone Filesystem Benchmark*. [Online]. Available: http://www.iozone.org, accessed May 29, 2014.

[65] *Iometer*. [Online]. Available: http://www.iometer.org, accessed May 29, 2014.

[66] R. Jones *et al.*, "Netperf: A network performance benchmark," Information Networks Division, Hewlett-Packard Company, Palo Alto, CA, USA, Tech. Rep., 1996. [Online]. Available: http://www.netperf.org/netperf/, accessed May 29, 2014.

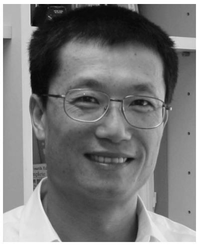
[67] *Nas Parallel Benchmark*. [Online]. Available: http://www.nas.nasa.gov, accessed May 29, 2014.

[68] J. Dongarra and P. Luszczek, "The LINPACK benchmark: Past, present and future," *Concurrency Comput., Pract. Exper.*, vol. 15, pp. 803–820, 2003.

[69] *Stressapptest*. [Online]. Available: http://code.google.com/p/stressapptest/, accessed May 29, 2014.

CHONGLIN GU received the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2011. In 2012, he joined the School of Computer Science and Technology at the Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China, where he is currently pursuing the Ph.D. degree. His research interests are in the fields of cloud computing and distributed computing, big data, in particular, algorithms design, and system implementation.

HEJIAO HUANG received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2004. She is currently a Professor with the Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China, and was an Invited Professor with the Institut National de Recherche en Informatique et Automatique, Rennes, France. Her research interests include cloud computing, trustworthy computing, formal methods for system design, and wireless networks.



XIAOHUA JIA (F'13) received the B.Sc. and M.Eng. degrees from the University of Science and Technology of China, Hefei, China, in 1984 and 1987, respectively, and the D.Sc. degree in information science from the University of Tokyo, Tokyo, Japan, in 1991. He is currently a Professor with the Department of Computer Science, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks, and mobile wireless networks. He is a fellow of the IEEE Computer Society.