

## HEURISTIC ALGORITHMS FOR ENERGY AND PERFORMANCE DYNAMIC OPTIMIZATION IN CLOUD COMPUTING

Lizheng GUO

*Department of Computer Science and Engineering  
Henan University of Urban Construction, Pingdingshan 467036, China*  
&  
*College of Information Sciences and Technology  
Donghua University, Shanghai 201620, China*  
*e-mail: kftjhpds@163.com*

Yifei ZHANG

*Department of Computer Science and Engineering  
Henan University of Urban Construction, Pingdingshan 467036, China*  
*e-mail: 10002091@qq.com*

Shuguang ZHAO

*College of Information Sciences and Technology  
Donghua University, Shanghai 201620, China*  
*e-mail: sgzhao@dhu.edu.cn*

**Abstract.** Cloud computing becomes increasingly popular for hosting all kinds of applications not only due to their ability to support dynamic provisioning of virtualized resources to handle workload fluctuations but also because of the usage based on pricing. This results in the adoption of data centers which store, process and present the data in a seamless, efficient and easy way. Furthermore, it also consumes an enormous amount of electrical energy, then leads to high using cost and carbon dioxide emission. Therefore, we need a Green computing solution that can not only minimize the using costs and reduce the environment impact but also

improve the performance. Dynamic consolidation of Virtual Machines (VMs), using live migration of the VMs and switching idle servers to sleep mode or shutdown, optimizes the energy consumption. We propose an adaptive underloading detection method of hosts, VMs migration selecting method and heuristic algorithm for dynamic consolidation of VMs based on the analysis of the historical data. Through extensive simulation based on random data and real workload data, we show that our method and algorithm observably reduce energy consumption and allow the system to meet the Service Level Agreements (SLAs).

**Keywords:** Cloud computing, green computing, virtual machine, dynamic consolidation

## 1 INTRODUCTION

Nowadays the Cloud computing is rapidly developing and it has already been used in life sciences [1], climate [2], astrophysics [3], engineering, and so forth. All the above mentioned applications in modern scientific research and common lives are resulting in a very large number of data centers generating all over the world. As corporations look for more energy efficiency, they examine their operations more closely. In order to handle the sheer magnitude of today's data, data centers have to use much more power as they become larger, denser, hotter, and significantly more costly to operate. The United States Environment Protection Agency (EPA) report to Congress on servers and data center energy efficiency estimates that USA data centers consume 1.5% of total USA electricity consumption for a cost of \$4.5 billion [4]. From the year 2000 to 2006, data center electricity consumption has doubled in the USA and is currently on a road to double again by 2011 to more than 100 billion kWh that equals to \$7.4 billion in annual electricity costs [5]. These data centers not only consume huge energy but also are a major contributor towards company's electricity bill [6]. Gartner warns that today's data centers are big energy consumers and are filled with high density power hungry IT equipments. If data center managers remain unaware of these energy issues they would most probably run the risk of doubling their energy costs between 2005 and 2011 [7]. If energy costs continue to double every five years, they will substantially increase to 1600% in the scope of the years 2005 and 2025 [8].

On the other hand, power is required to feed the cooling system operation in the data center. For each watt of power consumed by computing resources, an additional 0.5–1 W is required for the cooling system [9]. In addition, high energy consumption by the infrastructure leads to substantial carbon dioxide (CO<sub>2</sub>) emissions contributing to the greenhouse effect. So, in some sense, we try to optimize the service energy, and then reduce the energy of cooling system and carbon dioxide.

The reason for such extremely high energy consumption is not only just the large quantity of computing resources and the power inefficiency of hardware, but it

also lies in the inefficient usage of these resources. Data center energy savings can come from a number of places: on the hardware and facility side, e.g. by designing energy-efficient servers and data center infrastructures, and on the software side, e.g. through resource management. Our approach is motivated by two observations from real data sets collected from operating Internet services. First, data collected from more than 5 000 production servers over a six-month period has shown that although servers usually are not idle, the utilization rarely approaches 100 % [10]. Most of the time servers operate at 10–50 % of their full capacity, leading to extra expenses on over-provisioning. Moreover, managing and maintaining over-provisioned resources result in an increase of the Total Cost of Ownership (TCO). Another problem is the narrow dynamic power range of servers: even completely idle servers still consume about 70 % of their peak power [11]. Therefore, keeping servers underutilized is highly inefficient from the energy consumption perspective. Assuncao et al. [12] have conducted a comprehensive study on monitoring energy consumption by the Grid'5000 infrastructure. They have shown that there exist significant opportunities for energy conservation via techniques utilizing switching servers off or to low power modes.

With the capabilities of the virtualization technology, Cloud providers can create multiple VMs instances on a single physical server, thus improving the utilization of resources and increasing the Return On Investment (ROI). The reduction in energy consumption can be achieved by switching idle physical machines to low-power modes or shutdown, thus eliminating the idle power consumption. Although by using live migration the VMs can be dynamically consolidated to minimize the number of physical nodes according to their current resource requirements [13], efficient resource management in the Cloud is very important as modern service applications often experience highly variable workloads causing dynamic resource usage patterns. For example, researchers have reported the magnitude of daily workload fluctuations to be in the 40–50 % range for social networking applications, and about 70 % for e-commerce Web sites [14].

In the consolidation of VMs, the performance degradation is taken into account when an application encounters an increasing demand resulting in an unexpected rise of the resource usage. If the resource requirements of an application are not fulfilled, the application can face increased response times, time-outs or failures. Ensuring reliable Quality of Service (QoS) defined via SLAs established between Cloud providers and their customers is essential for Cloud computing environments; therefore, Cloud providers have to deal with the energy-performance trade-off, seeking the minimization of energy consumption while meeting the SLAs.

The key problem of this paper is on energy and performance efficient optimization using resource management strategies that can be applied in a virtualized data center by a Cloud provider. We investigate performance characteristics of heuristic algorithms for the problem of energy and performance through efficient dynamic VMs consolidation. In order to optimize performance and energy efficiency, we design integrated strategies for managing energy consumption and performance. These strategies include overloading and underloading detection based on historical data

from the source by the host, and VMs selection based on the performance of the VMs. Then, we propose and evaluate novel heuristics that adapt their behavior based on an analysis of historical data from the resource usage by VMs to optimize the VM deployment. In the meantime, we evaluate the proposed algorithms by random workload and real workload data which come from the resource usage by more than a thousand PlanetLab VMs provisioned for multiple users using the CloudSim toolkit. Simulation results show that the algorithms significantly not only reduce energy consumption but also maintain a high standard of the SLAs.

The rest of the paper is organized as follows: In Section 2, we discuss the related work. Section 3 describes the architectural model of the system. Section 4 presents the challenges of VM live migration and the details of the analysis. In Section 5, we present energy and performance aware dynamic consolidation of VMs. Section 6 describes the simulations and discusses the results. Finally, this paper draws a conclusion in Section 7.

## **2 RELATED WORK**

In the first work, power management has been researched in the context of virtualized data centers [15]. The authors have proposed local and global management policies to manage resources. At the local level, the system leverages the guest OS's power management strategies and the global manager gets the information on the current resource allocation from the local managers. Then, according to its policy, it decides whether the VMs placement needs to be adapted. But, the authors have not proposed a method for automatic resource management at the global level. Job scheduling policies in a cluster have been studied in [16]. The authors considered that there are servers with different performance and power characteristics, and the scheduler is not aware of the service times of jobs. Simulation results showed that each one of the three introduced policies has its own advantages and disadvantages. The shortest queue with energy efficiency priority (SQEE) policy, which is energy awareness, reduces the energy consumption of the system and yields average job response times. The shortest queue with high performance priority (SQHP) policy is optimized for performance, and thus, it outperforms both the other policies, while it yields the highest energy consumption. Lastly, the performance-based probabilistic-shortest queue (PBP'CSQ) policy performs the worst, especially at high load, and yields average energy consumption. It is the policy which provides the most effective load balance among the two server types, mainly at medium load. Their management policies are local and, thus, not suitable in a Cloud computing. Even worse, they do not dynamically adjust the number of VMs running on a given host. A comprehensive energy consumption analysis of Cloud computing has been presented in [17]. The analysis considered both public and private Clouds and included energy consumption in switching and transmission as well as data processing and data storage. Their conclusion is that the energy consumption of Cloud computing needs to be considered as an integrated supply chain logistics problem, in

which processing, storage, and transport are all considered together. Using this approach, they have shown that Cloud computing can enable more energy-efficient use of computing power, especially when the users' predominant computing tasks are of low intensity or infrequency. In [18], the authors have presented a novel adaptive scheduling strategy or adaptive energy-efficient scheduling (AEES) for periodic, independent real-time tasks on his dynamic voltage scaling (DVS) technique enabling heterogeneous clusters. AEES seamlessly integrates two algorithms-energy-efficient global scheduling algorithm (EEGS) and local voltage adjusting (LVA). EEGS is implemented in the scheduler that is able to adaptively adjust voltages according to system load to guarantee deadlines of all waiting tasks in local queues. LVA, implemented in the local adjuster, can decrease voltage levels of waiting tasks to conserve energy when a task is scheduled and dispatched to a computing node. Their method and strategy only improve the energy consumption through the DVS technology. In our simulation tests, we use the Dynamic voltage and frequency scaling (DVFS), which is more easily implemented than DVS.

A linear programming (LP) formulation and heuristics to control VMs migration have been presented [19], which prioritized VMs with steady capacity. This is possible by including constraints to define that VMs with steady usage are not migrated and virtual machines with variable capacity can be migrated to reduce the number of required physical servers. Their main aim is to minimize the migration problem in virtualized data centers. A threshold-based dynamic resource allocation scheme has been proposed for Cloud computing that dynamically allocates the virtual resources among the Cloud computing applications based on their load changes (instead of allocating resources needed to meet peak demands) and can use the threshold method to optimize the decision of resource reallocation [20]. The proposed threshold-based dynamic resource allocation scheme is implemented by CloudSim, and experimental results show that the proposed scheme can improve resource utilization and reduce the user's usage cost. Speitkamp and Bichler in [21] described linear programming formulations for the static and dynamic server consolidation problems. They also designed extension constraints for limiting the number of VMs on a physical server, guaranteeing some VMs that are assigned to different physical servers, mapping VMs to a specific set of physical servers that contain some unique attribute, and limiting the total number of migrations for dynamic consolidation. In addition, they proposed an LP-relaxation based heuristic for minimizing the cost of solving the linear programming formulations. However, the authors only concern about the cost but do not concern about the performance and energy efficiency. The authors in [22] revealed that processor time is not a satisfactory criterion for accurate estimation of power consumption by a VM and that some in-processor events affect processor power consumption more significantly than others. They suggested a model for estimating the energy consumption that calculates the amount consumed by a VM via monitoring of processor performance counters. Based on the proposed estimation scheme, they have proposed an energy-aware VM scheduler that can limit the energy consumption of the VM to their energy budget. Conventional VM schedulers only consider the processor time when it comes to scheduling decisions. Different from its

traditional counterparts, the energy-credit scheduler uses the energy consumption rates of VMs for scheduling. It schedules VMs so that their energy consumption rates remain below user-defined values. However, the authors only take into account the energy and they do not concern about the performance. Berral et al. have studied the problem of dynamic consolidation of VMs running applications with deadlines that are set in the SLAs [23]. Using machine learning techniques they optimize the combination of energy consumption and SLAs fulfillment. The proposed approach is designed for specific environments, such as High Performance Computing (HPC), where applications have deadline constraints. Therefore, such an approach is not suitable for environments with mixed workloads. In [24], the authors applied an approach based on the idea of setting fixed utilization thresholds. However, fixed utilization thresholds are not efficient for IaaS environments with mixed workloads that exhibit non-stationary resource usage patterns.

Our previous works [25, 26, 27], includes resource deployment and task scheduling model and algorithm, mainly optimize the cost, the time of processing and transferring. Unlike the above discussed studies, this paper proposes an approach which effectively deals with stringent QoS requirements, multi-core CPU architectures, heterogeneous infrastructure and heterogeneous VMs consolidation to save the energy consumption and keep performance. Moreover, we propose an energy and performance aware heuristic algorithm based on the analysis of historical data, which dynamically optimizes the allocation of VMs at runtime according to current resource utilization. At the same time, the idle node is shut down or switched to sleep mode to minimize energy consumption.

### 3 ARCHITECTURAL MODEL

Our main purpose is to reduce the energy consumption and to maintain the quality of service. Our research object is infrastructure as a service (IaaS), which contains a large number of data centers consisting of  $M$  heterogeneous physical nodes. The performance of each node is characterized by the CPU capacity, amount of RAM and network bandwidth. The CPU capacity is defined by the Millions Instructions Per Second (MIPS). We do not consider the disk size dimension because we assume that network-attached storage (NAS) is used as the main storage across the cluster. The potential benefits of network-attached storage, compared to file servers, include easier administration and simple configuration. So, the servers in the data center have not local disk and the storage is provided as NAS to enable live migration of VMs. In addition, a lot of users request the resources provided by the system hosted in a Cloud computing environment. These resources are characterized by requirements for processing power defined in MIPS, amount of RAM and network bandwidth. As the requirements of each user are different, the workload and the using time of each VM and physical node are also different.

The architecture of the system is described in Figure 1. Multiple independent users submit requests for provisioning of  $N$  heterogeneous VMs, which have various

MIPS, RAM, and bandwidth. The user negotiates SLAs with the Cloud provider. In the running time, if the provider violates the SLAs, it should pay a penalty for the users. In addition to providing SLAs, the system should provide efficient energy management. This is realized through a software layer of the system which is tiered comprising local and global managers. The local manager is in each of the nodes. The aim of the local manager is to monitor the node CPU utilization and RAMs available capacity, resize the VMs according to each requesting resource, and decide when and which VMs should be migrated from the node. The global manager is in the master node and gets information from the local manager to master the overall view of resource utilization. At the same time, global manager is in charge of optimizing the VM placement and decides to set up or shut down a physical node according to its CPU utilization.

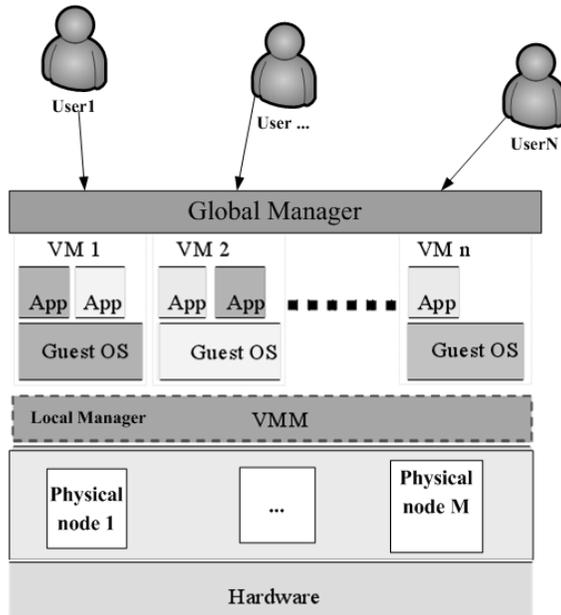


Figure 1. Architecture system

At present, physical servers are equipped with multi-core CPUs. If two VMs are running on the same server, the CPU utilization of the server is estimated as the sum of the CPU utilization of the two VMs. This is the case with memory resources. For example, let (10%, 20%) be a pair of the CPU and memory requests of a VM, and (7%, 15%) be that of another VM. Then, the utilizations of a server accommodating the two VMs are estimated at (17%, 35%). To prevent CPU and memory usage of a server reaching 100%, we have to impose an upper bound on resource utilization of a single server with some threshold value. The main idea behind this is that 100% utilization can cause the server queue to explode (that

is, the queue length increases abruptly and in an uncontrolled way), then you have a performance degradation. The only limitation is that the CPU capacity required for a VM must be less or equal to the capacity of a single core. The reason is that if the CPU capacity required for a VM that is higher than the capacity of a single core, a VM must be deployed on more than one core in parallel. However, we do not assume that VMs can be arbitrarily parallelized, as there is no a priori knowledge of the application running on a VM and automatic parallelization is a complex research problem.

#### 4 CHALLENGES OF LIVE MIGRATION

Live migration describes the process of copying a VM from one physical machine to another physical machine, while the VM is still powered on [13]. It provides special benefits to server virtualization and becomes a significant tool for a variety of scenarios. Many researches [28, 29, 30, 31] have verified that consolidating VMs through live migration of an optimal number of servers and selectively switching off underutilized servers can reduce data center's heat loss and power consumption, but it causes performance loss of processes running inside a VM as well as energy overhead. For example, previous studies demonstrate that the transmission rate of an Apache Web Server slows down by 12% to 20% [13] and energy consumption may increase by up to 10 Watt during live migration [32]. So, in order to optimize the performance and energy consumption, some criteria are used to test optimizing algorithms.

##### 1. Total Migration Time

Many researchers agree that, besides some constant overhead for resource reservation on the target host, the total migration time  $T_{mig}$  highly depends on the total amount of memory  $V_{mig}$  that has to be transmitted from source to target hypervisor and the average network link speed (network bandwidth)  $B_w$  between both hosts. It varies linearly with  $V_{mig}$  and inversely proportional to  $B_w$  and can be calculated as in (1).

$$T_{mig} = \frac{V_{mig}}{B_w} \quad (1)$$

##### 2. Performance Degradation

Previous studies have demonstrated that the transmission rate of a Web Server slows down by 12% to 20% and [34] has shown that the average performance degradation including the downtime of web-application can be estimated at approximately 10% of the CPU utilization. Thus, for our experiments we define degradation experienced by a  $VM_j$  as shown in (2).

$$P_{dmigj} = \alpha \cdot \int_{t_0}^{t_0+T_{migj}} S_j(t) dt \quad (2)$$

where  $P_{dmigj}$  is the total performance degradation by  $VM_j$  and  $t_0$  is the time when the migration begins;  $T_{migj}$  is the time taken to complete the migration.  $S_j(t)$  is the CPU utilization of the  $VM_j$  and  $\alpha$  is the coefficient that can be obtained from training.

### 3. Energy Consumption of Live Migration

VM live migration leading to the power increasing of the source and destination servers has been verified by reference [13]. First, the power influence of migration on the original server goes down with the increase of CPU usage of the migrated VM, but for the destination server, the influence is stable, which is around 10-Watt power cost. Additionally, the cost of migration processing is not impacted by the CPU usage of a VM. So, the energy consumption of live migration can be expressed as (3):

$$E_{mig} = \int_{t_0}^{t_0+T_{mig}} (1 + \delta) \cdot P_{s(j)}(t) dt + \int_{t_0}^{t_0+T_{mig}} (1 + \lambda) \cdot P_{d(j)}(t) dt \quad (3)$$

where  $E_{mig}$  is the energy consumption of VM live migration;  $P_{s(j)}$  is the power of the source server involved in the VM migration and  $P_{d(j)}$  is the destination;  $\delta$  and  $\lambda$  are the increasing amount power of the server. According to the above definitions, the total energy consumption is as follows (4):

$$E = E_{mig} + \int_{t_{st}}^{t_0} P_{sb(j)}(t) dt + \int_{t_0+T_{mig}}^{t_{sh}} P_{sa(j)}(t) dt \quad (4)$$

$$+ \int_{t_{st}}^{t_0} P_{db(j)}(t) dt + \int_{t_0+T_{mig}}^{t_{sh}} P_{da(j)}(t) dt \quad (5)$$

where  $t_{st}$  and  $t_{sh}$  are the start time and the shutdown time of a server;  $P_{sb}$  and  $P_{sa}$  are the power of the source server migrating before and after;  $P_{db}$  and  $P_{da}$  are the power of destination server migrating before and after.

### 4. Performance Metric of Live Migration

Today, customers are charged based upon resource usage or reservation. So, the Cloud provider should meet the QoS requirements. QoS requirements are commonly formalized in the form of SLAs, which can be determined in terms of such characteristics as minimum throughput or maximum response time delivered by the deployed system. However, the performance that an application will obtain from a given amount of resource can vary. A significant source of variation from performance interference effects virtualized applications that are deployed onto multicore servers. It is very important to ensure that the performance experienced by applications is independent of whether it is consolidated with other workloads. So, in this paper, we propose two metrics that can be used to evaluate the SLAs delivered to any VM deployed in IaaS. The first is

Performance Violation Percentage (PVP) (6).

$$PVP = \frac{1}{M} \sum_{i=1}^M \frac{T_{vi}}{T_{ai}} \cdot 100\% \quad (6)$$

where  $M$  is the number of physical nodes in an IaaS;  $T_{vi}$  is the time during the physical node  $i$  experiencing the utilization of 100% to lead to an SLAs violation;  $T_{ai}$  is the time of the physical node  $i$  being in the active state. The second is Performance Degradation Percentage (PDP) (7).

$$PDP = \frac{1}{N} \sum_{i=1}^N \frac{P_{dmigj}}{P_{rj}} \cdot 100\% \quad (7)$$

where  $N$  is the number of VMs;  $P_{dmigj}$  is the performance degradation of the VM  $j$  caused by migrations;  $P_{rj}$  is the total CPU capacity requested by the VM  $j$  during its lifetime. Both the PVP and PDP metrics are independent and of equal importance in the characterization of the SLAs violations by the infrastructure. Therefore, we propose a combined metric that encompasses performance degradation both due to host overloading and due to VM migrations. We denote the combined metric SLAs Violation (SLAV), which is calculated as shown in (8).

$$SLAV = PVP \cdot PDP. \quad (8)$$

## 5 ENERGY AND PERFORMANCE AWARE DYNAMIC CONSOLIDATION METHOD OF VIRTUAL MACHINE

Through the above analysis, we know that energy and performance are two dependent aspects in all kinds of application. In this section, we propose several heuristic algorithms for dynamic consolidation of VMs based on energy and performance awareness. In fact, the server utilization is usually less than 50% in Cloud computing which has most of the servers and all kinds of application servers, so, in order to optimize the energy consumption, we should dynamically migrate VMs and shut down the redundant servers.

Energy and performance awareness of dynamic consolidation of VMs can be broken down into three questions:

1. determining when one or more VMs should be migrated from the server which is overloaded or underloaded; in overloaded case, migrating virtual machine is to keep performance, and in underloaded case, migrating virtual machine is to reduce energy consumption;
2. determining which VM should be migrated from an overloaded or underloaded server; and
3. determining where should be migrated into for a overloaded or underloaded server. In the following, we discuss the above three problems.

## 5.1 Overloaded or Underloaded Detection Method

A heuristic method for deciding the time to migrate VMs from a host based on utilization threshold has been proposed in [24]. The main ideas are to set upper and lower utilization threshold for hosts and keep the total utilization of the CPU by all the VMs between these thresholds. If the CPU utilization of a host exceeds the upper threshold, one or some VMs must be migrated from the host to reduce the utilization. If the CPU utilization of a host is below the lower threshold, all the VMs have to be migrated from this host and the host has to be switched to the sleep mode or shut down in order to reduce energy consumption. Although the fixed value of utilization threshold is simple, it is unsuitable for an environment with dynamic workload, in which different classes of applications and requests can share a host. In order to fit the threshold to a varying environment, designing an auto-adjustment of the utilization thresholds based on the historical data is necessary.

### 1. Overloading Detection Method

Qiao et al. have proposed a polynomial regression modeling to predict the central processing rate (CPU) of MapReduce jobs in a Cloud computing environment [33]. Furthermore, four different methods for adaptively adjusting the utilization threshold have been previously proposed in literature [34]. They are the Median Absolute Deviation, Interquartile Range, Local Regression and Robust Local Regression, respectively. In addition, their experiments showed that the Local Regression method is the best one. In this paper, we also use the Local Regression method to predict the upper threshold utilization of the host's CPU.

Local Regression based on the Loess method has been proposed by Cleveland [35]. The main idea of the method of LR is fitting simple models to localized subsets of data to build up a curve that approximates the original data. The observations  $(x_i, y_i)$  are assigned neighborhood weights using the tricube weight function shown in (9).

$$T(u) = \begin{cases} (1 - |u|^3)^3, & \text{if } |u| < 1, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

Using the described method derived from Loess, for each new observation they have found a new trend line  $g^\wedge(x) = a^\wedge + b^\wedge x$ . This trend line is used to estimate the next observation  $g^\wedge(x_{k+1})$ . The algorithm decides that the host is considered overloaded and some VMs should be migrated from it if the inequalities (10) are satisfied.

$$s \cdot g^\wedge(x_{k+1}) > 1, \quad x_{k+1} - x_k < t_m \quad (10)$$

where  $s \in R^+$  is the safety parameter; and  $t_m$  is the maximum time required for a migration of any of the VMs allocated to the host. The safety parameter  $s$

can be obtained from training. The training result was the best when the  $s$  took 1.2 in the training. So in this paper, with the term “LR overloading detection method” we mean Local Regression with  $s = 1.2$ . The value of  $a$  and  $b$  are found by minimizing the function shown in (11).

$$\sum_i^k w_i(x)(y_i - a - bx_i)^2 \quad (11)$$

where  $x_k$  is the last observation, and  $x_1$  is the  $k^{\text{th}}$  observation from the right boundary. Let  $x_i$  satisfy  $x_1 \leq x_i \leq x_k$ , and, therefore, the tricube weight function can be simplified as for  $0 \leq u \leq 1$  and the weight function is as follows (12):

$$w_i(x) = T\left(\frac{\Delta_i(x_k)}{\Delta_1(x_k)}\right) = \left(1 - \left(\frac{x_k - x_i}{x_k - x_1}\right)^3\right)^3. \quad (12)$$

## 2. Host Underloading Detection Method

Arithmetic mean (AM) and minimal utilization of host (MU)

The arithmetic mean (or simply “mean”) of a sample  $(x_1, x_2, \dots, x_n)$  is the sum of the sampled values divided by the number of items in the sample:

$$\bar{x} = \frac{x_1 + x_2 + \dots + x_n}{n}. \quad (13)$$

The method decides that the host is considered underloading and one VM, some VMs or all the VMs should be migrated from it if the inequalities (14) are satisfied.

$$h \in H_i \mid \forall a \in H_i, H_u(h) \leq H_u(a) \text{ and } H_u(h) < s \cdot AM \quad (14)$$

where  $H_u(h)$  is the amount of CPU currently utilized by the host  $h$ ;  $H_u(a)$  is the amount of CPU currently utilized by the host  $a$ ; and  $s \in R^+$  is the safety parameter. According to the AM method, if the CPU utilization of the host is the least in all the hosts and less than the  $s \cdot AM$ , this host is considered as underloading. If a host is detected as underloading, some VMs or all the VMs must be migrated from the underloading host to other host in the light of the following VM placement method. If we change inequality (14) to inequality (15), the AM detection method becomes MU.

$$h \in H_i \mid \forall a \in H_i, H_u(h) \leq H_u(a). \quad (15)$$

In terms of the MU method, if the CPU utilization of the host is minimal in all the hosts, this host is considered underloading. Just as well the AM method, the VMs have to be migrated from the underloading host.

First, all the overloaded hosts are detected using the overloading detection algorithm and the VMs which are selected by the VM selecting algorithm are migrated

from the overloading hosts to the destination hosts. Then, computing the arithmetic mean of all the host CPU utilization, the system finds the host with the minimum utilization compared to the other hosts with CPU utilization below the AM, and tries to place the VM from underloading host on other hosts keeping them not overloaded. If this can be accomplished, the VM will be migrated to the determined target hosts, and the source host is switched to the sleep mode once all the migrations have been completed. If all the VMs from the underloading host cannot be placed on other hosts, the underloading host is kept active. This process is iteratively repeated for all hosts.

## 5.2 VM Selection Method

Once the host overloading detection algorithm has run, an overloaded host is selected and has to choose a VM to migrate to the other hosts. The key problem is a VM selecting policy which not only saves energy consumption, but also preserves higher performance. This section presents three policies for VM selecting. When a selecting VM has been migrated, the host must be checked again by overloaded algorithm. Once it is still considered as being overloaded, the VM selection policy is applied again to select another VM to migrate from the host until the host is considered as being not overloaded.

### 1. Minimum Migration Time Policy

The Minimum Migration Time (MMT) policy migrates a VM  $v$  that requires the minimum time to complete a migration relatively to the other VMs. The migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host  $j$ . Let  $V_j$  be a set of VMs currently allocated to the host  $j$ . The MMT policy finds a VM  $v$  that satisfies the conditions formalized in (16)

$$v \in V_j \mid \forall a \in V_j, \frac{RAM_u(v)}{NET_j} \leq \frac{RAM_u(a)}{NET_j} \quad (16)$$

where  $RAM_u(a)$  is the amount of RAM currently utilized by the VM  $a$ ; and  $NET_j$  is the spare network bandwidth available for the host  $j$ .

### 2. Maximum CPU Utilization and the Minimum CPU Utilization

The Maximum CPU Utilization (MAU) policy migrates a VM  $v$  that meets the maximum CPU utilization. Let  $V_j$  be a set of VMs currently allocated to the host  $j$ . The MAU policy finds a VM  $v$  which satisfies the conditions formalized in (17)

$$v \in V_j \mid \forall b \in V_j, CPU_u(v) \geq CPU_u(b) \quad (17)$$

where  $CPU_u(b)$  is the amount of CPU utilization currently utilized by the VM  $b$ . Likewise, the Minimum CPU Utilization (MCU) can be expressed as follows:

$$v \in V_j \mid \forall b \in V_j, CPU_u(v) \leq CPU_u(b). \quad (18)$$

### 5.3 VM Placement Method

The VM placement problem that deploys all the VMs from the overloaded hosts and underloaded hosts to other hosts makes the active host minimal and keeps better performance. This problem can be seen as a two-dimensional bin packing problem with variable bin width and height, where bins represent the physical nodes; items are the VMs that have to be allocated; bin width is the available RAM ( $wr$ ) amount of the nodes; and bin height is the available CPU ( $hc$ ) capacities of the nodes. A set  $V = \{v_1, v_2, v_i, \dots, v_n\}$ , which  $v_i = \{wr_i, hc_i\}$  is an element, is waiting for allocation VMs, and destination host machine set is  $H = \{h_1, h_2, \dots, h_m\}$ ,  $h_j = \{Wr_j, Hc_j\}$ . Find a sub set  $H'$  of  $H$ ,  $H' = \{h'_1, h'_2, \dots, h'_m\}$ ,  $h'_j = \{Wr'_j, Hc'_j\}$

$$H' = \{h'_1, h'_2, \dots, h'_{|H'|}\} \subset H, h'_j = \{Wr'_j, Hc'_j\} \quad (19)$$

and set up a mapping  $f: V \rightarrow H'$

$$\text{Min}|H|, \text{Min}|E|, \quad \text{s.t.} \quad \sum_{v_i \rightarrow h'_j} hc_i < Hc'_j, \quad \sum_{v_i \rightarrow h'_j} wr_i < Hr'_j. \quad (20)$$

As the bin packing problem is NP-hard, we apply a greedy strategy. We use the modification of the algorithm denoted Power Aware Best Fit (PABF). Firstly, get all the VMs and hosts, then allocate each VM to a host that provides the least increase of the power consumption caused by the allocation. This allows to leverage the node heterogeneity by choosing the most power-efficient ones first. The details of the PABF are listed in Algorithm 1. The complexity of the algorithm is  $O(nm)$ , where  $n$  is the number of hosts and  $m$  is the number of VMs that have to be allocated. Similarly, the other methods like Best Fit Decreasing (BFD), First Fit Decreasing (FFD) and First Fit (FF) can also be implemented.

### 5.4 The General Optimizing Algorithm of VM Deployment

The general optimizing algorithm of VM deployment is shown in Algorithm 2. Firstly, each host is checked by the overloading method, and determines whether a host is overloaded. If the host is overloaded, the VM selection method chooses a VM from the overloaded host and adds it to waiting for migration list. This operation is repeated until the load of the overloaded host drops below the threshold. Once the hosts are checked by the overloaded method and build the migration list of VMs, the PABF optimization algorithm is applied to deploy the VMs of waiting for migration. Secondly, applying the underloading method finds the underloaded host and using the VM deployment method deploys the VMs from these hosts. In the worst case, the complexity of the algorithm is  $O(n + nm + mn^2)$ , where  $n$  is the number of hosts and  $m$  is the number of VMs that have to be allocated.

---

**Algorithm 1: Power aware best fit (PABF)**


---

```

1. input: host_list, vm_list, output: mapping of vm to host
4. for vm in vm_list do
5.   minPower = plus_infinity
6.   vmMappingHost = null
7.   for host in host_list do
8.     if a host has enough resources for VM then
9.       power = computing (host, VM)
10.      if power < minPower then
11.        vmMappingHost = host
12.        minPower = power
13.      end
14.    end
15.  end
16.  if vmMappingHost ≠ NULL then
17.    Map.add(vm, vmMappingHost)
18.  end
19.  return vmMappingHost
20. end

```

---

## 6 PERFORMANCE EVALUATION

### 6.1 Simulation Set Up

In order to evaluate the performance of the proposed algorithms in a Cloud computing environment, we must research all kinds of workload models, resources provisioning policies, resource deploying policies and resource scheduling algorithms and so on. On the one hand, there are large scales of resources. On the other hand, the user is charged according to the using resource type. So, evaluating the performance of Cloud provisioning policies, application workload models, and resource performance models in a repeatable manner, varying system and different user requirements is difficult to achieve. To overcome this challenge and ensure the repeatability of experiments, simulations have been used to evaluate the performance of Cloud computing.

The CLOUDS Lab at the University of Melbourne has developed the CloudSim Toolkit software which is released as open source and has been used as a simulation platform. For instance, HP Labs (Palo Alto) researchers are using CloudSim for evaluation of resource allocation algorithms for HP's Cloud data centers; Duke University (USA) researchers are using it for energy-efficient management of Data Centers; National Research Center for Intelligent Computer Systems (Beijing, China) researchers are using it for SLAs oriented management and optimization of Cloud computing environments; and Kookmin University (Seoul, Korea) researchers are using the toolkit for their investigation on workflow scheduling in Clouds. So, in

---

 Algorithm 2: The general optimizing algorithm for VM deployment
 

---

```

1. input: host_list, output: mapping of vm to host
2. for host in host_list do
3.   while host_overloaded(host) do
4.     vms_to_migrate.add(Selecting a vm from the overloaded host)
5.   end
6. end
7. PABF(host, vms_to_migrate_list)
8. for host in host_list do
9.   if host_underloaded(host) then
10.    vms_to_migrate.add (host.getVmList ())
11.   end
12.   PABF(host, vms_to_migrate_list)
13.   if host_is_NULL(host) then
14.     shutDown(host) or switch to sleep mode
15.   end
16. end
17. return vmMappingHos

```

---

this paper, we also use the CloudSim which is the 3.02 version to simulate our experiment<sup>1</sup>.

In order to simulate the real experiment, we have selected three servers from the main page of the Standard Performance Evaluation Corporation<sup>2</sup> as physical servers, and the configurations of them are listed in Table 1. Power consumption characteristics of the selected servers are presented in Table 2. In addition, the characteristics of the VM types in our test correspond to Amazon EC2 instance types. In our test, we use the M1 instance types of family, which provides a balance of computing, memory, and network resources, and it is a good choice for many applications. The configurations of the VMs are listed in Table 3.

Simulation test data is divided into two categories: random and real workload. In the random test, we used the ProLiant ML110 G5 and the IBM System x3650 M4 servers as the host. We have simulated a data center, which comprised 100 heterogeneous physical hosts, and the users submitted 100 heterogeneous VM requests at the same time in the random test data. The CPU utilization of the VMs distributed was generated based on the uniform distribution. Initially the VMs were allocated according to the resource requirements defined by the VM types. Besides, to make the experiments reproducible, it is important to rely on test data to regenerate the workload consistently, which would allow the experiments to be repeated as many times as necessary. At the same time, it is more important for test data to use workload traces collected from a real system rather than artificially generated, as this would help to reproduce a realistic scenario. The real workload comes from the

---

<sup>1</sup> <http://cloudbus.org/cloudsim/>

<sup>2</sup> [http://www.spec.org/power\\_ssj2008/results/](http://www.spec.org/power_ssj2008/results/)

open platform PlanetLab<sup>3</sup>. The workload has been applied to 1052 VMs spanning over 800 servers located at more than five hundred locations around the world. The interval of utilization measurements is 5 minutes. But at the same time, in the real workload, we used the ProLiant ML110 G5 and the ProLiant ML110 G4 servers as the host. Initially, each VM was randomly assigned a workload trace from the real workload. In the simulations all the tasks were submitted to the data center at the same time, as this would benefit to dynamically adjust the consolidation of the VMs and stress the experiment objective of the consolidation algorithms.

Hardware Vendor	Server Type	CPU Description	MHz	Core	Total Memory (GB)	Servers CPUs Are Mapped Onto MIPS
Hewlett-Packard	ProLiant ML110 G4	Intel Xeon Processor 3040	1 860	2	4	1 860
Hewlett-Packard	ProLiant ML110 G5	Intel Xeon Processor 3075	2 660	2	4	2 600
IBM	IBM System x3650 M4	Intel Xeon E5-2660	2 200	16	24	2 200

Table 1. Configuration of the selected servers

Server	0 %	10 %	20 %	30 %	40 %	50 %	60 %	70 %	80 %	90 %	100 %
ProLiant ML110 G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
ProLiant ML110 G5	93.7	97	101	105	110	116	121	125	129	133	135
IBM System x3650 M4	57.2	84	93.2	103	114	129	148	171	193	226	262

Table 2. Power consumption of the selected servers at different load levels in watts

Instance Type	vCPU	Memory (GiB)	VM vCPUs Are Mapped onto MIPS
m1. small	1	1.7	500
m1. medium	1	3.75	1 000
m1. large	2	7.5	1 500
m1. xlarge	4	15	2 000

Table 3. Configuration of the VMs

<sup>3</sup> <http://www.planet-lab.org/>

## 6.2 Comparison of Methods and Metrics

Here, we have simulated some combinations of the host overloading detection method (LR), two detection methods of hosts underloading (MU, AM), three methods of VM selection (MMT, MCU, MAU) and four methods of VM deployment (PABFD, PABF, PAFFD, PAFF). In order to test the performance of every combination, the same test data and metrics are used to evaluate the above methods. One of the metrics is the total energy consumption which is caused by the physical servers in a data center. The metrics used to evaluate the SLAs are SLAV, PDP, and PVP described in Section 3.2. The other metrics are the number of VMs migration (VMN) and the number of hosts shutdown (HN) by the management system during the running time. In the above metrics, the main metrics are energy consumption and SLAV, but, there is a strong negative correlation between the two metrics as SLAs violations are usually decreased by the cost of the energy increase. Our main purpose is to optimize the energy and SLA violations, so we propose a metric that is the combination of the two metrics, which can be defined Energy and SLA violation (ESLVA) in (21).

$$ESLVA = E \cdot SLVA. \quad (21)$$

Policy	Energy (KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
No power aware	77.75	0	0	0	0	56	0
DVFS	14.14	0	0	0	0	56	0
LR_MU_MMT_PABFD	4.99	340	0.11402 %	0.30 %	37.84 %	120	0.00569
LR_MU_MMT_PABF	4.89	279	0.08031 %	0.26 %	31.34 %	111	0.003927
LR_MU_MMT_PAFF	4.35	363	0.20369 %	0.42 %	48.49 %	108	0.008861
LR_MU_MMT_PAFFD	4.21	316	0.16101 %	0.35 %	46.60 %	108	0.006779
LR_MU_MCU_PABFD	4.72	285	0.13908 %	0.34 %	40.78 %	81	0.006565
LR_MU_MAU_PABFD	4.84	257	0.09430 %	0.28 %	32.23 %	102	0.004564

Table 4. Simulation results of the PABFD, PABF, PAFF, PAFFD, MCU and MAU algorithm combination using random data

Policy	Energy(KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
LR_MU_MMT_PABFD	4.99	340	0.11402 %	0.30 %	37.84 %	120	0.00569
LR_0.8AM_MMT_PABF	4.85	229	0.06571 %	0.23 %	28.76 %	96	0.003187
LR_0.8AM_MAU_PABF	4.86	232	0.06719 %	0.23 %	28.76 %	96	0.003265
LR_0.8AM_MMT_PABFD	4.87	225	0.06055 %	0.23 %	26.90 %	96	0.002949
LR_0.8AM_MAU_PABFD	4.84	224	0.07078 %	0.24 %	29.35 %	96	0.003426

Table 5. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination using random data

### 6.3 Simulation Results and Analysis

The authors have demonstrated that the LR\_MU\_MMT\_PABFD combination is the best method of optimizing the performance and energy in their simulations [34]. So in our test, we mainly compare our test to the LR\_MU\_MMT\_PABFD policy. We firstly test the random data and use the overloading methods (LR), underloading methods (MU), the VM's deploying methods, the VM's selecting methods. In the simulation, the hosts and the VMs are all 100 and the simulation time is 4 hours; the scheduling interval is 300 seconds for the live migration of VMs. The simulation results are shown in Table 4.

From Table 4, we draw the conclusion that

1. the energy consumption of Dynamic Voltage Frequency Scaling (DVFS) algorithm obviously outperforms No Power Aware (NPA);
2. dynamic VM consolidation algorithms effectively reduce the energy consumption comparing with DVFS and NPA;
3. the metric of LR\_MU\_MMT\_PABF policy and LR\_MU\_MAU\_PABFD policy are better than LR\_MU\_MMT\_PABFD policy.

The conclusions of Table 4 present that the MAU of the VM's selecting algorithm and the PABF of the VM's deploying algorithm can get better results. Furthermore, large numbers of simulation results show that the AM are optimal when the safety parameter of the AM algorithm is 0.8, in all the underloading detection methods. Therefore, in the following, we use the above conclusions and combine with the underloading detection algorithm to optimize the energy consumption and performance. The results are presented in Table 5.

From Table 5, we can see that the 0.8AM and LR policy combined with the MMT, BF, MAU and BFD is better than the LR\_MU\_MMT\_PABFD policy. The reason is that the MU policy selects the host with the lowest resource utilization, and tries to migrate the VMs from this host to other hosts keeping them not overloaded. However, when using the 0.8AM policy, the system selects a host to consider the host utilization not only minimal but also less than 0.8AM. Thus, the hosts do not migrate VMs when all the hosts' utilization is more than 0.8AM. In contrast to this, the host continues to select a host to migrate its VMs when all the hosts utilization is higher when using the MU policy. The host has a relatively high utilization. One of the reasons is that the VM resource utilization running on the host may be high, and the other reason is that the number of VMs on the host may be relatively higher. At this point, if we continue to migrate VMs from the high utilization of host, serious problems will occur. On the one hand, you need to migrate a lot of VMs and the time of migration will be relatively long. At the same time, the system will lead to an increase in energy consumption and service degradation during the migration. On the other hand, the host utilization is relatively high, if the load has seen a dramatic change, the host is more easy to overload, and lead to performance degradation. Besides,

the amount of the VMs migrated and hosts shutdown reduce comparing with the MU.

Policy	Energy (KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
No power aware	2 410.80	0	0	0	0	457	0
DVFS	803.91	0	0	0	0	457	0
LR_MU_MMT_PABFD	176.16	27 041	0.00003690	0.07 %	5.21 %	5472	0.00650030
LR_0.8AM_MMT_PABFD	174.74	20 788	0.00001860	0.04 %	4.29 %	4811	0.00325016
LR_0.8AM_MAU_PABFD	149.28	11 481	0.00004840	0.10 %	4.90 %	3298	0.00722515
LR_0.8AM_MMT_PABF	177	21 042	0.00001870	0.04 %	4.22 %	4658	0.00330990
LR_0.8AM_MAU_PABF	149.47	11 272	0.00004800	0.10 %	4.82 %	3358	0.00717456

Table 6. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $300 \times 288$  seconds using real workload

Policy	Energy (KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
LR_MU_MMT_PABFD	118.83	27 366	0.00008510	0.11 %	7.81 %	5 550	0.01011243
LR_0.8AM_MMT_PABFD	117.00	20 724	0.00004100	0.06 %	6.31 %	4 830	0.004797
LR_0.8AM_MAU_PABFD	100.30	11 355	0.00010690	0.15 %	7.21 %	3 366	0.01072207
LR_0.8AM_MMT_PABF	118.50	21 021	0.00004410	0.07 %	6.36 %	4 759	0.00522585
LR_0.8AM_MAU_PABF	100.31	11 311	0.00105300	0.15 %	7.15 %	3 369	0.100562643

Table 7. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $200 \times 288$  seconds using real workload

Policy	Energy (KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
LR_MU_MMT_PABFD	63.21	27 451	0.00029900	0.21 %	14.45 %	5 550	0.01889979
LR_0.8AM_MMT_PABFD	58.99	20 297	0.00016020	0.13 %	12.46 %	4 722	0.009450198
LR_0.8AM_MAU_PABFD	51.27	11 403	0.00040680	0.29 %	13.98 %	3 350	0.020856636
LR_0.8AM_MMT_PABF	60.01	20 702	0.00016710	0.14 %	12.31 %	4 635	0.010027671
LR_0.8AM_MAU_PABF	51.17	11 209	0.00039220	0.29 %	13.72 %	3 318	0.020068874

Table 8. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $100 \times 288$  seconds using real workload

The above conclusion and analysis disclose that the 0.8AM policy and LR policy, combined with the MMT, BF, MAU and BFD, can achieve good results. So in the following, in order to make the simulation more realistic, we use the real workload to test the above conclusions. In the simulations, the scheduling interval is 300 seconds for the live migration of VMs because the interval of utilization measurements is 300 seconds in 24 hours, i.e., the original workload traces are made of 288 sample points. So, firstly, we test the above methods with the interval scheduling of 300 seconds in 24 hours and the test results are listed in Table 6. From Table 6, we can see that the real workload further validates the above conclusion. In addition, when the overloading detection policy (LR), underloading detection policy (0.8AM), the VM

deploying policy (PABF or PABFD) are the same, the VM selection policy MAU is better than MMT in the energy consumption, the number of VM migration and host shutdowns; but the performance is lower than MMT. The reason is that the MMT policy can quickly migrate a VM from an overloading host and make the overloading host to normal state, so the performance outperforms the MAU policy. However, the MAU policy can make an overloading host not to overload in a long time if the workload is relatively stable, at the same time, the host can preserve a relatively high utilization. For this reason, MAU policy reduces the energy consumption, the number of host shutdowns and VM migrations.

In order to test the robustness of the proposed methods, we scale the interval of the scheduling time that is 200, 100 and 50 seconds. All the simulation results are presented in Table 7, Table 8 and Table 9 respectively and all these results confirm the proposed methods once more. The mean value of the sample means along with 95% confidence interval (CI) measured for each method during the experiments is listed in Table 10 and Table 11. The time before a host is switched to the sleep mode (HSS) for the method combination is approximately 15 to 20 minutes when the interval of scheduling time is 300 seconds, or approximately 5 minutes when the interval of scheduling time is 50 seconds. This value is very important for real-world systems, as modern servers have low-latency transitions to the sleep mode consuming low power. According to the data provided by Meisner et al. [36], power consumption of a typical blade server is 450 W in the fully utilized state, 270 W in the idle state, and 10.4 W in the sleep mode, while the transition delay is 300 ms. The mean value of the sample means of the VM selecting (VMS) time is about 2 ms for the different methods combination respectively; the mean value of the sample means of the host selecting (HS) time is about 20 ms for the different methods combination respectively; the mean value of the sample means of the VM relocating (VMR) time from 54 ms to 240 ms with the different methods combination.

Policy	Energy (KWh)	VMN	SLAV	PDP	PVP	HN	ESLVA
LR.MU.MMT.PABFD	31.39	27 551	0.00126760	0.43 %	29.29 %	5 548	0.039789964
LR.0.8AM.MMT.PABFD	30.61	20 601	0.00060410	0.25 %	24.01 %	4 736	0.018491501
LR.0.8AM.MAU.PABFD	26.69	11 484	0.00153850	0.57 %	26.86 %	3 333	0.041062565
LR.0.8AM.MMT.PABF	31.21	21 037	0.00063190	0.27 %	23.68 %	4 754	0.019721599
LR.0.8AM.MAU.PABF	26.74	11 261	0.00154580	0.58 %	26.47 %	3 367	0.041334692

Table 9. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $50 \times 288$  seconds using real workload

## 7 CONCLUSION AND FUTURE WORK

The increasing cost of energy consumption and the worldwide desire to reduce CO<sub>2</sub> emissions have felt concern about the energy efficiency of information and communication technology. Moreover, to maximize the return on investment, Cloud

Policy	HSS (s)	VMS (ms)	HS (ms)	VMR (ms)
LR_MU_MMT_PABFD	1026.91	3.85	19.13	212.12
	(817.19, 1236.63)	(3.44, 4.26)	(18.37, 19.89)	(205.77, 218.47)
LR_0.8AM_MMT_PABFD	931.22	2.99	20.29	196.75
	(659.13, 1203.31)	(2.81, 3.17)	(19.51, 21.07)	(190.94, 202.66)
LR_0.8AM_MAU_PABFD	1228.01	1.86	20.14	58.72
	(933.51, 1522.51)	(1.75, 1.97)	(19.32, 20.96)	(57.07, 60.37)
LR_0.8AM_MMT_PABF	921.61	2.68	19.46	179.92
	(672.15, 1171.07)	(2.52, 2.84)	(18.70, 20.22)	(174.66, 185.18)
LR_0.8AM_MAU_PABF	1199.49	1.80	17.72	54.72
	(897.24, 1501.74)	(1.69, 1.91)	(16.91, 18.53)	(53.14, 56.30)

Table 10. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $300 \times 288$  seconds using real workload

Policy	HSS (s)	VMS (ms)	HS (ms)	VMR (ms)
LR_MU_MMT_PABFD	181.17	3.69	20.80	239.45
	(148.76, 213.58)	(3.47, 3.91)	(20.03, 21.57)	(232.34, 246.56)
LR_0.8AM_MMT_PABFD	177.50	2.79	20.37	183.71
	(125.72, 229.28)	(2.62, 2.96)	(19.59, 21.15)	(177.91, 189.51)
LR_0.8AM_MAU_PABFD	211.34	1.91	18.23	57.14
	(173.79, 268.89)	(1.79, 2.03)	(17.38, 19.08)	(55.56, 58.72)
LR_0.8AM_MMT_PABF	164.97	2.78	19.59	191.27
	(123.98, 205.96)	(2.61, 2.95)	(18.83, 20.35)	(186.79, 195.75)
LR_0.8AM_MAU_PABF	214.75	1.52	18.50	56.13
	(168.12, 261.38)	(1.43, 1.61)	(17.72, 19.28)	(54.53, 57.73)

Table 11. Simulation results of the PABFD, PABF, MMT and MAU algorithm combination for the simulation time  $50 \times 288$  seconds using real workload

providers have to apply energy-efficient technology to reduce energy consumption, such as DVFS and dynamic consolidation of VMs. However, decreasing the energy consumption and improving performance is a self-contradiction. When we design the algorithm and management policy, we take into account not only energy, but also performance. In this paper we have described the architecture, the performance parameter and a model for the VM migrating problems. We have concluded that it is necessary to develop an adaptive underloading detection policy of the host which adaptively selects a host to improve the performance and energy consumption. Moreover, we have proposed several adaptive heuristic algorithms that are based on analyzing the historical data to optimize the VMs dynamic consolidation and to optimize the performance and energy consumption.

We have tested the proposed algorithms through the extensive simulation of random and real data. The test results of the simulation have verified that the proposed underloading detection policies, VM selection policies and VM deployment algorithms are efficient with regard to the ESLVA metric because they substantially

reduce the level of SLAs violations, energy consumption, the number of VM migrations and host shutdown. In future work, we plan to test the proposed methods in a real-work, such as Openstack. Furthermore, we plan to research on heuristic algorithm that minimizes the cost of the user using the Cloud resources while it can maintain user QoS requesting and minimize the energy consumption.

## Acknowledgement

This research was supported by the National Natural Science Foundation of China under Grant Nos. 41771433 and 61271114, Key Technologies Research and Development Program of Henan Province, China under Grant No. 172102210174, Key Project of Higher Education of Henan Province, China No. 18A520020 and the Key Programs of Science and Technology Research of Henan Education Committee, China under Grant No. 12A520006. Furthermore, we thank the reviewers very much for their positive suggestions on enhancing the quality of the paper.

## REFERENCES

- [1] SAEZ-RODRIGUEZ, J.—GOLDSIPE, A.—MUHLICH, J.—ALEXOPOULOS, L. G.—MILLARD, B.—LAUFFENBURGER, D. A.—SORGER, P. K.: Flexible Informatics for Linking Experimental Data to Mathematical Models via DataRail. *Bioinformatics*, Vol. 24, 2008, No. 6, pp. 840–847, doi: 10.1093/bioinformatics/btn018.
- [2] HENDRIX, W.—TETTEH, I. K.—AGRAWAL, A.—SEMAZZI, F.—LIAO, W. K.—CHOUDHARY, A.: Community Dynamics and Analysis of Decadal Trends in Climate Data. 2011 IEEE 11<sup>th</sup> International Conference on Data Mining Workshops (ICDMW), December 2011, pp. 9–14, doi: 10.1109/ICDMW.2011.51.
- [3] LATHAM, R.—DALEY, C.—LIAO, W. K.—GAO, K.—ROSS, R.—DUBEY, A.—CHOUDHARY, A.: A Case Study for Scientific I/O: Improving the FLASH Astrophysics Code. *Computational Science and Discovery*, Vol. 5, 2012, No. 1, Art. No. 015001, doi: 10.1088/1749-4699/5/1/015001.
- [4] BROWN, R.—MASANET, E. R.—NORDMAN, B.—TSCHUDI, W. F.—SHEHABI, A.—STANLEY, J.—KOOMEY, J. G.—SARTOR, D. A.—CHAN, P. T.: Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431. Berkeley, CA, 2008.
- [5] DAIM, T.—JUSTICE, J.—KRAMPITS, M.—LETTS, M.—SUBRAMANIAN, G.—THIRUMALAI, M.: Data Center Metrics: An Energy Efficiency Model for Information Technology Managers. *Management of Environmental Quality: An International Journal*, Vol. 20, 2009, No. 6, pp. 712–731.
- [6] MCNAMARA W.—SEIMETZ G.—VALES, K. A.: Best Practices for Creating the Green Data Center. 2008, <http://www.it-solutions.siemens.com/b2b/it/en/us/Documents/publications/best-practices-green-it-white-paperPDFe.pdf>. Accessed in April 2013.

- [7] ROWSELL-JONES, A.—MINGAY, S.: Going Green: The CIO's Role in Enterprisewide Environmental Sustainability. <https://www.gartner.com/doc/663208>. Accessed in June 2013.
- [8] KUMAR R.: Media Relations. Gartner, <http://www.gartner.com/it/page.jsp?id1/4781012>. Accessed in June 2013.
- [9] RANGANATHAN, P.—LEECH, P.—IRWIN, D.—CHASE, J.: Ensemble-Level Power Management for Dense Blade Servers. *ACM SIGARCH Computer Architecture News*, Vol. 34, 2006, No. 2, pp. 66–77, doi: 10.1109/ISCA.2006.20.
- [10] BARROSO, L. A.—HÖLZLE, U.: The Case for Energy-Proportional Computing. *IEEE Computer*, Vol. 40, 2007, No. 12, pp. 33–37, doi: 10.1109/MC.2007.443.
- [11] FAN, X.—WEBER, W.-D.—BARROSO, L. A.: Power Provisioning for a Warehouse-Sized Computer. *ACM SIGARCH Computer Architecture News*, Vol. 35, 2007, No. 2, pp. 13–23.
- [12] DE ASSUNCAO, M. D.—GELAS, J.-P.—LEFEVRE, L.—ORGERIE, A.-C.: *The Green Grid'5000: Instrumenting and Using a Grid with Energy Sensors. Remote Instrumentation for eScience and Related Aspects*, Springer New York, 2012, pp. 25–42.
- [13] CLARK, C.—FRASER, K.—HAND, S.—HANSEN, J. G.—JUL, E.—LIMPACH, C.—PRATT, I.—WARFIELD, A.: Live Migration of Virtual Machines. *Proceedings of the 2<sup>nd</sup> Conference on Symposium on Networked Systems Design and Implementation (NSDI'05)*, Vol. 2, 2005, pp. 273–286.
- [14] TAK, B. C.—URGAONKAR, B.—SIVASUBRAMANIAM, A.: To Move or Not to Move: The Economics of Cloud Computing. *Proceedings of the 3<sup>rd</sup> USENIX Conference on Hot Topics in Cloud Computing (HotCloud'11)*, 2011, p. 5.
- [15] NATHUJI, R.—SCHWAN, K.: VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems. *ACM SIGOPS Operating Systems Review – SOSP '07*, Vol. 41, 2007, No. 6, pp. 265–278, doi: 10.1145/1294261.1294287.
- [16] ZIKOS, S.—KARATZA, H. D.: Performance and Energy Aware Cluster-Level Scheduling of Compute-Intensive Jobs with Unknown Service Times. *Simulation Modelling Practice and Theory*, Vol. 19, 2011, No. 1, pp. 239–250.
- [17] BALIGA, J.—AYRE, R. W. A.—HINTON, K.—TUCKER, R. S.: Green Cloud Computing: Balancing Energy in Processing, Storage, and Transport. *Proceedings of the IEEE*, Vol. 99, 2011, No. 1, pp. 149–167, doi: 10.1109/JPROC.2010.2060451.
- [18] ZHU, X.—HE, C.—LI, K.—QIN, X.: Adaptive Energy-Efficient Scheduling for Real-Time Tasks on DVS-Enabled Heterogeneous Clusters. *Journal of Parallel and Distributed Computing*, Vol. 72, 2012, No. 6, pp. 751–763.
- [19] FERRETO, T. C.—NETTO, M. A. S.—CALHEIROS, R. N.—DE ROSE, C. A. F.: Server Consolidation with Migration Control for Virtualized Data Centers. *Future Generation Computer Systems*, Vol. 27, 2011, No. 8, pp. 1027–1034, doi: 10.1016/j.future.2011.04.016.
- [20] LIN, W.—WANG, J. Z.—LIANG, C.—QI, D.: A Threshold-Based Dynamic Resource Allocation Scheme for Cloud Computing. *Procedia Engineering*, Vol. 23, 2011, pp. 695–703, doi: 10.1016/j.proeng.2011.11.2568.

- [21] SPEITKAMP, B.—BICHLER, M.: A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers. *IEEE Transactions on Services Computing*, Vol. 3, 2010, No. 4, pp. 266–278, doi: 10.1109/TSC.2010.25.
- [22] KIM, N.—CHO, J.—SEO, E.: Energy-Credit Scheduler: An Energy-Aware Virtual Machine Scheduler for Cloud Systems. *Future Generation Computer Systems*, Vol. 32, 2014, pp. 128–137, doi: 10.1016/j.future.2012.05.019.
- [23] BERRAL, J. L.—GOIRI, Í.—NOU, R.—JULIÀ, F.—GUITART, J.—GAVALDÀ, R.—TORRES, J.: Towards Energy-Aware Scheduling in Data Centers Using Machine Learning. *Proceedings of the 1<sup>st</sup> International Conference on Energy-Efficient Computing and Networking (e-Energy '10)*, April 2010, ACM, pp. 215–224, doi: 10.1145/1791314.1791349.
- [24] BELOGLAZOV, A.—ABAWAJY, J.—BUYAYA, R.: Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing. *Future Generation Computer Systems*, Vol. 28, 2012, pp. 755–768, doi: 10.1016/j.future.2011.04.017.
- [25] GUO, L.—ZHAO, S.—SHEN, S.—JIANG, C.: Task Scheduling Optimization in Cloud Computing Based on Heuristic Algorithm. *Journal of Networks*, Vol. 7, 2012, No. 3, pp. 547–553.
- [26] GUO, L.—ZHAO, S.—SHEN, S.—JIANG, C.: A Particle Swarm Optimization for Data Placement Strategy in Cloud Computing. *Information Engineering and Applications*. Springer, London, *Lecture Notes in Electrical Engineering*, Vol. 154, 2012, pp. 946–953.
- [27] GUO, L.—HE, Z.—ZHAO, S.—ZHANG, N.—WANG, J.—JIANG, C.: Multi-Objective Optimization for Data Placement Strategy in Cloud Computing. *Information Computing and Applications (ICICA 2012)*. Springer Berlin Heidelberg, *Communications in Computer and Information Science*, Vol. 308, 2012, pp. 119–126, doi: 10.1007/978-3-642-34041-3\_18.
- [28] ANDREOLINI, M.—CASOLARI, S.—COLAJANNI, M.—MESSORI, M.: Dynamic Load Management of Virtual Machines in Cloud Architectures. *Cloud Computing (Cloud-Comp 2009)*. Springer Berlin Heidelberg, *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, Vol. 34, 2010, pp. 201–214, doi: 10.1007/978-3-642-12636-9\_14.
- [29] LI, B.—LI, J.—HUAI, J.—WO, T.—LI, Q.—ZHONG, L.: EnaCloud: An Energy-Saving Application Live Placement Approach for Cloud Computing Environments. *IEEE International Conference on Cloud Computing (CLOUD '09)*, 2009, pp. 17–24, doi: 10.1109/CLOUD.2009.72.
- [30] MI, H.—WANG, H.—YIN, G.—ZHOU, Y.—SHI, D.—YUAN, L.: Online Self-Reconfiguration with Performance Guarantee for Energy-Efficient Large-Scale Cloud Computing Data Centers. *2010 IEEE International Conference on Services Computing (SCC)*, 2010, pp. 514–521, doi: 10.1109/SCC.2010.69.
- [31] IMADA, T.—SATO, M.—KIMURA, H.: Power and QoS Performance Characteristics of Virtualized Servers. *2009 10<sup>th</sup> IEEE/ACM International Conference on Grid Computing*, 2009, pp. 232–240.

- [32] HUANG, Q.—GAO, F.—WANG, R.—QI, Z.: Power Consumption of Virtual Machine Live Migration in Clouds. 2011 Third International Conference on Communications and Mobile Computing (CMC), 2011, pp. 122–125, doi: 10.1109/CMC.2011.62.
- [33] QIAO, Y.-Y.—LIU, F.—LING, Y.—YIN, J.-S.: Resource Modeling and Performance Prediction of MapReduce in Cloud Computing Environment. Journal of Beijing University of Posts and Telecommunications, 2014, S1.
- [34] BELOGLAZOV, A.—BUYA, R.: Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers. Concurrency and Computation: Practice and Experience, Vol. 24, 2012, No. 13, pp. 1397–1420.
- [35] CLEVELAND, W. S.: Robust Locally Weighted Regression and Smoothing Scatterplots. Journal of the American Statistical Association, Vol. 74, 1979, No. 368, pp. 829–836, doi: 10.1080/01621459.1979.10481038.
- [36] MEISNER, D.—GOLD, B. T.—WENISCH, T. F.: PowerNap: Eliminating Server Idle Power. ACM SIGARCH Computer Architecture News – ASPLOS 2009, Vol. 37, 2009, No. 1, pp. 205–216, doi: 10.1145/1508244.1508269.

**Lizheng GUO** has completed his M.Sc. degree in the College of Computer Science and Technology of Chongqing University of Posts and Telecommunications, Chongqing, China in 2005. He is currently pursuing his Ph.D. in pattern recognition and intelligent systems at the College of Information Science and Technology, Donghua University, Shanghai, China. His research interests include distributed systems, intelligent computing, cloud computing.

**Yifei ZHANG** is Lecturer at the Department of Computer Science, Henan University of Urban Construction (since 2009). His main interests focus on computer networks and operating systems.

**Shuguang ZHAO** received his B.Sc. degree in information processing from Xidian University, Xi'an, China, 1986, the M.Sc. degree in signal and information processing and Ph.D. degree in circuit and systems from Xidian University, Xi'an, China, in 1992 and 2003, respectively. He is currently Professor at Donghua University. His research interests include evolvable hardware and intelligent signal processing.