An On-Line Virtual Machine Consolidation Strategy for Dual Improvement in Performance and Energy Conservation of Server Clusters in Cloud Data Centers

Weiwei Lin, Wentai Wu, *Student Member, IEEE*, and Ligang He, *Member, IEEE*

Abstract—As data centers are consuming massive amount of energy, improving the energy efficiency of cloud computing has emerged as a focus of research. However, it is challenging to reduce energy consumption while maintaining system performance without increasing the risk of Service Level Agreement violations. Most of the existing consolidation approaches for virtual machines (VMs) consider system performance and Quality of Service (QoS) metrics as constraints, which usually results in large scheduling overhead and impossibility to achieve effective improvement in energy efficiency without sacrificing some system performance and cloud service quality. In this article, we first define the metrics of peak power efficiency and optimal utilization for heterogeneous physical machines (PMs). Then we propose Peak Efficiency Aware Scheduling (PEAS), a novel strategy of VM placement and reallocation for achieving dual improvement in performance and energy conservation from the perspective of server clusters. PEAS allocates and reallocates VMs in an on-line manner and always attempts to maintain PMs working in their peak power efficiency via VM consolidation. Extensive experiments on Cloudsim show that PEAS outperforms several energy-aware consolidation algorithms with regard to energy consumption, system performance as well as multiple QoS metrics.

Index Terms—Cloud computing, energy efficiency, heterogeneous server clusters, virtual machines, dynamic consolidation

1 Introduction

1.1 Motivation

As the demands for scalable and easy-to-use compute resources keep growing rapidly, cloud services have been widely adopted in a wide range of researches as well as business applications. As a trend, Cloud Service Providers (CSPs) are developing flexible features especially in resource provisioning. For example, features like automatic deployment and adaptive scale-out/scale-in make application development and operations (DevOps) much easier via PaaS. However, the developer-friendly and user-friendly natures of cloud services also make it challenging for CSPs to organize resources and provide services of high availability in a cost-efficient manner. Meanwhile, excessive energy consumption by cloud data centers has already become a prominent issue these years [1]. While early in 2011, the energy consumption of data centers already accounted for 1.1–1.5 percent of the total energy consumption globally [2], data centers' total power requirement soared by 63 percent in one year [3]. Without reining in power, the ever-growing consumption of electricity will certainly make the development of cloud services unsustainable. Therefore, it is of great necessity to promote energy efficiency of cloud computing.

Cloud data centers are complex systems and their energy consumption comes from a lot of sub-systems such as power supply/distribution, cooling system and ICT infrastructures. Reddy *et al.* [47] present a holistic view of measuring energy efficiency of a whole cloud data center. It also reveals the intractable complexity we face in optimizing the system as a whole. Nevertheless, managing ICT infrastructures are believed to be the key because they typically make the most significant impact on data centers' energy efficiency [49], [50]. Besides, solutions for ICT infrastructures are more generic than those for other sub-systems like cooling module which varies in design from data center to data center. In view of that, we in this paper focus on the optimization of the most fundamental part—the server cluster.

Much work has been done on optimizing dynamic virtual machine (VM) consolidation with the aim of increasing the performance (e.g., [42], [43]), reducing energy consumption (e.g., [37], [38], [40]) or comprehensive optimization of cloud servers (e.g., [36], [39], [41]). Some studies (e.g., [8], [44], [45], [46]) formulated the consolidation problem as a single-objective optimization in which energy consumption, number of hosts or budget is the target function, subject to several constraints such as resource demand, performance requirements and job deadline. These solutions can avoid serious performance degradation and service unavailability. However, there are two prominent limitations for constraint-based approaches. First, solving the optimization problem

• *W. Lin is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China. E-mail: linww@scut.edu.cn.*
• *W. Wu and L. He are with the Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom. E-mail: {wentai.wu, ligang.he}@warwick.ac.uk.*

Manuscript received 13 June 2019; revised 7 Nov. 2019; accepted 10 Dec. 2019. Date of publication 20 Dec. 2019; date of current version 8 Apr. 2022. (Corresponding author: Wentai Wu.)
Digital Object Identifier no. 10.1109/TSC.2019.2961082

with multiple constraints is usually time-consuming. Virtual machine consolidation is commonly formulated as an upgraded bin-packing problem with heterogeneous bins (servers) and more constraints. So it is of high complexity to find the optimal solution (allocation plan) considering a large set of virtual machines (i.e., an NP-hard problem). A widely-adopted approach in previous studies (e.g., [7], [8], [23]) is to consolidate VMs in batches. However, it is difficult to determine the batch size because there is a trade-off between scheduling efficiency and solutions' optimality. Second, constraint-based approaches can hardly achieve comprehensive optimization. In other words, existing studies neglect the potential possibility that a cloud system's energy efficiency, performance and service quality can be simultaneously improved. Juarez *et al.* [4] incorporated performance metric (s) in their target function, which turns the problem into a multi-objective optimization. Though the problem can be converted to single-objective by weighing each objective, it is apparently hard to determine the empirical parameters of weights.

Online consolidation algorithms with energy awareness show advantages in efficiency and adaptability. Some heuristic approaches (e.g., [9], [29], [35]) also showed their effectiveness in reducing computing or holistic energy consumption of cloud data centers. However, system performance is not taken into account in their heuristic principles, which may unexpectedly increase job run time as well as the risk of Service Level Agreement (SLA) violations.

1.2 Our Contributions

In this paper, we propose to take advantage of servers' power efficiency to achieve dual improvement in clusters' performance and energy efficiency. Usually it is possible to attain a fine allocation plan that effectively reserves energy with acceptable job run time and Quality of Service (QoS) guaranteed. We achieve this, in our approach, by utilizing PMs' peak power efficiency. Peak power efficiency is attained when the physical machine (PM) is working in an "optimal" state where the ratio of its performance to its power consumption is maximized. Power efficiency well measures the performance (output) of a PM provided with a fixed amount of power (input). Peak power efficiency thus indicates the potential "productivity" of a machine. We build our consolidation strategy based on three basic principles. First, PMs are prioritized based on peak power efficiency because "productive" PMs are likely to be both high-performance and energy-efficient. Second, the scheduler should try to keep every active PM in its optimal state. We do not maximize PMs' resource usage but try to maintain them in their optimal utilization instead. Third, workload (VMs) should be consolidated without violating the second principle when resources in the system remain adequate. Besides the basic principles, the overhead of scheduling (i.e., making consolidation decisions) in our design should not increase drastically as the problem scales up. Batch-based schedulers (e.g., [8], [9]) can reorder the unscheduled VMs to achieve better result but in some way prolongs job response time and increases consolidation overhead. Thus we adopt an online approach in which VMs are scheduled in an FIFO manner. In this way the awaiting time for VM creation and migration is shortened. The major contributions of this paper are as follows:

1. We define the metrics of peak power efficiency and optimal utilization for physical machines. By using Compute Resource Units (CRUs) to quantify performance, we further derive the optimal number of CRUs that a PM needs to offer before attaining its peak power efficiency.
2. A novel VM consolidation strategy, Peak Efficiency Aware Scheduling (PEAS), is proposed. PEAS is comprised of two algorithms: Peak Efficiency Aware Placement (PEAP) and Peak Efficiency Aware Cost-efficient Reallocation (PEACR). PEAP finds the best host for each VM based on the three principles mentioned. PEACR is responsible for redistributing the VMs from overloaded and under-utilized hosts.
3. Extensive experiments were conducted on CloudSim [10] for evaluating the performance of PEAS and several baseline algorithms such as PABFD [11] and TVRSM [12]. The result shows the novelty of our strategy in reducing energy consumption, shortening average job run time and improving QoS with only small scheduling overheads.

The remainder of this paper is organized as follows. Section 2 introduces the related studies on optimizing the scheduling of virtual machines. In Section 3 we present a system model and provide mathematics problem definition for VM scheduling. Then we propose and illustrate our PEAS strategy for VM placement and reallocation in Section 4. Section 5 demonstrates the results of our simulation conducted on Cloudsim 3.0.3, where PEAS is compared with several energy-aware algorithms. We finally conclude our research in Section 6.

2 Related Work

As an NP-hard problem, scheduling in a heterogeneous system has been explored extensively. Li *et al.* [13] proposed a simple heuristic in VM placement to minimize total job completion time of VM requests. The scheme allocates VMs to the most high-performance host in a greedy manner and attempts to force another VM to migrate out in case resources on that host is not sufficient. This aggressive policy may cause a large number of migrations. Yokoyama *et al.* [14] proposed a VM consolidation algorithm for private clouds based on task affinity, which reflects the interference between virtual machines. The authors experimentally explored the affinity relation between applications and then proposed an affinity-aware consolidation algorithm to consolidate them. Lin *et al.* [15] proposed a VM placement algorithm based on analyzing virtual machines' similarity in peak workload characteristics and improved resource utilization by consolidating VMs with low correlation.

Rather than performance, improving the energy efficiency of cloud computing arose as the new focus of research. Uddin *et al.* [16] and Wu *et al.* [17] summarize techniques for reducing energy consumption in cloud computing and big data systems, respectively. Rong *et al.* [18] figured out the significance of energy-efficient scheduling for energy conservation in cloud data centers. The work introduces multi-level resource optimization and emphasizes that fine-grained consolidation (of VMs) has a significant impact on data centers' energy efficiency. Borgetto *et al.* [19] present an energy-saving framework in their work combining dynamic scaling, VM reallocation and VM reconfiguration. For VM reallocation, they proposed four migration schemes accordingly based on First Fit, Round Robin, Monte Carlo and Vector Packing. However, the framework may bring about performance degradation as only energy consumption and SLA are considered. Bölöni and Turgut [20] proposed a Value of Information (VoI) based scheduler to reduce the cost of cloud computing. Their key idea is a benefit-cost model by which the scheduler can calculate the financial benefit from a job given a specified amount of compute resources. Taking workload variation into account, Chen *et al.* [8] proposed FDSP (feasibility driven stochastic Placement) algorithm which consists of two tiers. The first tier is variation-unaware VM placement that generates an energy-efficient allocation plan resolved using Integer Linear Programming. The allocation plans' feasibility, in the second tier, is checked given an incremental amount of load. The two processes run recurrently for a batch of VMs until a feasible plan is accepted. Duan *et al.* [7] applied Ant Colony Optimization (ACO) along with a load prediction module to the consolidation of VMs. In their design, ants' pheromone on each host is associated with the estimated power consumption of the whole plan. [22] used Artificial Bee Colony (ABC) to search solutions for problem of consolidation in a cloud with multiple different service providers. Cho *et al.* [23] adopted a combined approach of ant colony optimization with particle swarm (ACOPS) to achieve reduction in job completion time and load balancing. The algorithm also rejects requests that are unable to satisfy for speeding up the procedure. A common drawback search-based approaches (e.g., [5], [6], [7]) is that they usually lead to large scheduling overheads due to the immense search space.

Online heuristic algorithms are commonly adopted in VM consolidation. Beloglazov *et al.* [29] proposed a Modified Best Fit Decreasing (MBFD) VM placement algorithm that greedily allocates each VM to the host that will yield the minimum increment of energy consumption. A similar energy-aware VM placement strategy is used in [11] while the study also proposes a Minimum Migration Time (MMT) policy to optimize virtual machine reallocation. With a different focus, Lago *et al.* [21] suggested that the prominent increase in bandwidth had made migration much cheaper. They therefore proposed a Bandwidth-Aware Lago Allocator for energy-efficient VM placement, which greedily selects PMs with high energy efficiency and low power. The allocator cooperates with a bandwidth provisioning module to allow more flexible VM migrations.

Search-based consolidation algorithms have common drawbacks. They can only work in batch mode and usually suffer large scheduling overheads due to the immense search space especially in large-scale heterogeneous circumstances like hybrid cloud and federated cloud [24], [25]. Adopting online approaches, therefore, is a better option in large-scale data centers. Zhu *et al.* [12] proposed a heuristic VM placement algorithm named HVRAA that experimentally outperformed ABC in energy efficiency by reducing both time consumption and hosts in use. The authors also designed algorithms, namely MP-VRSA and VROA, for VM migration from overloaded hosts and migration from under-utilized hosts, respectively. The entire strategy was proved effective in reducing overall energy consumption.

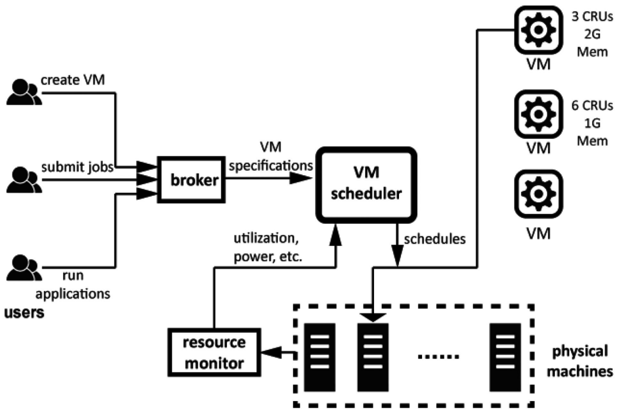


Fig. 1. A virtual machine (re)allocation framework for clouds providing flexible services.

Xiao *et al.* [25] used Markov Decision Process to model the states of a scheduler. The study used Q-learning to constantly update the maximum accumulated reward for each state and optimized the exploration by assigning dynamic learning rates to different states and transitions. Though machine learning techniques may bring about new approaches to optimizing VM consolidation, it is still very challenging to apply them to complex cloud environments. As an alternative, Berral *et al.* [26] exploited machine learning to predict workload, power consumption and SLA metrics.

3 System Model

Cloud computing still has a great potential of growth as predicted by Gartner [27]. We can see a trend that cloud services are showing more flexibility to developers and providing easy-to-use functions to business clients. On this point, we propose to abstract CPU resource as Compute Resource Units (CRUs) and present a scheduling framework (Fig. 1) adaptive to the heterogeneity of infrastructures.

A CRU is defined as a fixed unit of compute capacity (i.e., 1000 MIPS in our research) similar to the Elastic Compute Unit in Amazon Web Service (AWS). Using CRUs to define compute resource has two benefits. First, it measures the performance of VM instances without exposing the underlying physical resources. Second, physical resources (i.e., bare-metal hosts) can be efficiently organized based on the number of CRUs they can offer. Fig. 1 demonstrates the proposed framework for VM consolidation (i.e., dynamic allocation and reallocation of VMs). Users can specify resource demands (CRUs, memory size, etc.) when sending VM creation requests, submitting batch jobs or deploying applications via the service broker, which is an abstract gateway module and can be associated to a concrete implementation like the API server in Openstack-Nova. The resource monitor module collects server states and sends the information to the scheduler on a regular basis. As the core of our framework, the scheduler finds the most suitable host for the coming VM after analyzing performance, resource utilization and power efficiency of all physical machines in the cluster (Fig. 1). The proposed strategy should serve in the central scheduling module of a cloud system, for example, the Nova-scheduler in the Nova sub-system of Openstack.

TABLE 1
List of Symbols

Name	Description
$p(t)$	The power consumption of host at time t
$a(t)$	The activity of host at time t
u_s	The CPU utilization of host s
$h_{i,j}(t)$	The placement of VM j on host i at time t
$d_{j,r}(t)$	VM_j's demand for resource r at time t
$res_{r,i}$	Total amount of resource r on host i
H	The set of hosts
V	The set of virtual machines
N	Number of hosts
M	Number of VMs
V	The current VM to be scheduled
V_{out}	The set of VMs migrated out
MAX_C	The max number of CRUs that any host can offer
SH	The list of hosts ordered by *peak_peff*
L_c	The Optimal Offer List containing hosts of which $CRUs\_offer = c$
$ps_s(u)$	The processing speed of host s under utilization u
$peff_s(u)$	The power efficiency of host s under utilization u
u_opt_s	The optimal utilization of host s
$peak\_peff_s$	The peak power efficiency of host s
$CRUs\_offer_s$	The optimal number of CRUs to be offered by host s
CRU_max_s	The total number of CRUs that host s has in full capacity
bw_s	The available bandwidth of host s
$CRUs\_VM_v$	The number of CRUs required by the VM v
m_size_v	The size of memory used by VM v
UP_THR	The upper threshold (static)
LW_THR	The lower threshold (dynamic)
W_SIZE	The size of time window

3.1 Problem Definition

For clearity, we first list all the symbols that may be frequently used in Table 1.

Servers account for the major proportion of energy consumption in cloud data centers [28]. It has been observed that a server's power is a function of CPU utilization [29], [30], [31]. The power of other components (e.g., memory and disk) is usually counted in server idle power as they make much less contribution to the total power consumption. So a PM's power can be expressed as (1):

$$p(t) = power(u(t)), \tag{1}$$

where $u(t)$ is CPU utilization at time t and power is the unique power function of the host.

3.1.1 Metrics

A server's energy consumption within a period T (from t_s to t_e) is:

$$E = \int_{t_s}^{t_e} p(t)a(t)\ dt, \tag{2}$$

where $a(t)$ denotes the activity of the server at time t. $a(t) = 1$ if the server is active, otherwise $a(t) = 0$. However, it is impossible to apply (2) directly. Instead, we sample resource usage at fixed intervals (Δt) and use formula (3) to estimate a server's energy consumption. With $T = \frac{t_e - t_s}{\Delta t} - 1$, we define the total energy consumption of a host as (3) and that of the whole cluster as (4):

$$E = \sum_{t=0}^{T} p(t)a(t)\ \Delta t \tag{3}$$

$$total\_E = \sum_{i=1}^{n}\sum_{t=0}^{T} p_i(t)a_i(t)\ \Delta t, \tag{4}$$

where n denotes the number of machines in the set H. Here we consider a limited time span to calculate the energy consumption and assume that every job or application in the VM has a definite running time. Thus $(T+1)\cdot\Delta t$ is the finishing time of the last VM in the set V. Performance should also be warranted in cloud. Here we define average run time of VMs as (5):

$$avg\_RT = \frac{1}{m}\sum_{j=1}^{m} FT_j - ST_j, \tag{5}$$

where FT_j and ST_j are the finishing time and start time of VM j, respectively. A VM's run time depends on its resource share (e.g., CRUs) as well as workload characteristics. While we aim to optimize the performance of all VM provisioning requests, we analyze it from the perspective of jobs because average run time is the most intuitive metric.

3.1.2 Constraints

An obvious constraint is that every server is either active or powered off at a given moment:

$$a_i(t)\ \in\ \{0,\ 1\},\ \forall host_i \in H. \tag{6}$$

Besides, a VM can only be hosted by one server at any moment. For each VM in the set V, let variable $h_{i,j}$ denote the placement of VM j on host i at time t, we have constraints (7) and (8):

$$h_{i,j}(t) = \begin{cases} 1, & vm_j\ is\ running\ on\ host_i \\ 0, & otherwise \end{cases} \tag{7}$$

$$\sum_{i=1}^{n} h_{i,j}(t) = 1, \forall vm_j \in V. \tag{8}$$

Resources allocated to VMs by a host machine should be limited to its available amount of resources. Let $d_{j,r}(t)$ stand for VM_j's demand for resource r at time t and $res_{r,i}$ the total amount of resource r on host i, respectively. So a feasible allocation plan should meet constraint (9):

$$\sum_{j=1}^{m} h_{i,j}(t)d_{j,r}(t)\ <\ res_{r,i}\ , \forall r \in R\ \forall host_i \in H, \tag{9}$$

where R is the set of resource types. We mainly consider CPU, memory, disk space and bandwidth in our work. Note that we do not consider makespan or execution time as a constraint because actual time consumption is often unpredictable while users' specification of resource requirements makes it unnecessary to set this limitation.

3.1.3 Target

The problem of VM consolidation within a cluster for achieving improvement in both performance and energy reservation can be formulated as a multi-objective optimization:

$$min\ (total\_E,\ avg\_RT), \quad s.t.\ \ (6)(7)(8)\ and\ (9),$$

The problem is similar to but more complicated than the classical bin-packing problem which is NP-hard. Therefore we do not resort to deterministic solutions. Instead, we in this paper adopt a heuristic-based approach based on the observation that consolidating load on PMs with high peak power efficiency usually leads to lower energy consumption and shorter VM run time. Besides, we use soft bounds instead of strict constraints on resource usage due to the elastic nature of cloud services. Resource budgets are incorporated in QoS metrics for measuring service quality.

3.2 QoS Metrics

(1) *SLA violations.* Hosts have limited amount of compute resource (i.e., CRUs), memory size, storage and bandwidth. We define the average violation rate of Service-Level Agreement (SLA) at host level as (10):

$$host\_SLAV = \frac{\sum_{i=1}^{n}\sum_{t=0}^{T} overload_i(t)}{n(T+1)}, \tag{10}$$

where $overload_i(t) = 1$ indicates $host_i$ is overloaded at moment *t* (host utilization is check at fixed intervals), otherwise $overload_i(t) = 0$. Over-utilization of a host usually causes resource contention between the VMs allocated to it. Consequently VM(s) on the host may suffer performance degradation due to insufficient resource share. We in turn define the average SLA violation rate at VM level as below:

$$vm\_SLAV = \frac{\sum_{j=1}^{m}\sum_{t=0}^{T} underprov_j(t)}{m(T+1)}, \tag{11}$$

where $underprov_j(t) = 1$ indicates that VM_j is running with sufficient resources required, otherwise $underprov_j(t) = 0$.

(2) *Number of VM migrations*. The number of VM migrations makes a great impact on system's performance and energy efficiency. Frequent migration of VMs between PMs may bring about overheads on resource usage as well as system instability. We consider the total number of VM migrations within the entire cluster defined as below:

$$num\_MGT = \sum_{j=1}^{m} vm\_MGT_j, \tag{12}$$

where vm_MGT_j denotes the number of times that VM_j is migrated from one host to another.

(3) *Scheduling overhead*. The scheduler itself incurs overhead no matter it works in batch-mode or in an online manner. In this paper, we average the scheduling overheads in the placement and reallocation of each VM to indicate the scheduler's efficiency.

4 Peak Efficiency Aware Scheduling

To achieve dual improvement in energy conservation and system performance, we in this paper propose a Peak Efficiency Aware Scheduling (PEAS) strategy for virtual machines in cloud services. Instead of simply maximizing resource utilization, PEAS focuses on running PMs in their peak power efficiency. PEAS consists of two algorithms: PEAP and PEACR. Peak Efficiency Aware Placement (PEAP) algorithm is designed for VM placement while Peak Efficiency Aware Cost-efficient Reallocation (PEACR) algorithm is applied to balancing workload via VM migration.

4.1 Peak Power Efficiency

Inspired by [9] and [32], we define server power efficiency as the ratio of its performance to power consumption (similar to the Power per Watt metric in [51]). It is a function of CPU utilization with the assumption that both performance and power of a running PM depend on CPU utilization:

$$peff(u) = \frac{ps(u)}{power(u)}, \tag{13}$$

where $ps(u)$ is the processing speed when the PM's current utilization is *u*. Here we use MIPS (Million Instructions Per Second) to measure its performance. With definition (13), we can further determine the optimal utilization u_opt given a series of CPU utilization levels and the corresponding performance and power data:

$$u_{opt} = \underset{u}{argmax}\ \frac{ps(u)}{power(u)}, u = u_1, u_2, \ldots, u_z, \tag{14}$$

where $\{u_1, u_2, \ldots, u_z \mid 0 \leq u_k \leq 1\}$ is a set of CPU utilization values observed. Then for a PM we define its Peak Power Efficiency as:

$$peak\_peff = peff(u_{opt}). \tag{15}$$

So the point is that we can optimize VM placement and consolidation with the knowledge of every PM's peak power efficiency and optimal utilization. We then derive the optimal number of CRUs to be offered by a PM as (16):

$$CRUs\_offer = \left\lfloor \frac{ps(1)}{CRU\_UNIT} max\{u_{opt} - u, 0\} \right\rceil, \tag{16}$$

where $ps(1)$ is the full capacity of a host (in MIPS) and CRU_UNIT is a constant equal to 1000 MIPS in this paper. Formula (16) implies that optimally the number of CRUs a host needs to offer equals to the product of its full capacity in CRU and the utilization margin $(u_{opt} - u)$ before attaining its peak power efficiency. The value is rounded and would be zero if the machine is already running above its optimal utilization obtained via (14).

A group of Optimal Offer Lists is used to quickly spot the best PM that has a $CRUs\_offer$ just enough for the VM. When there is no available host in the corresponding optimal offer list, the scheduler will fall back to search the Sorted Host List for a suitable host (Fig. 2).

4.2 Virtual Machine Placement

We first introduce the Peak Efficiency Aware Placement (PEAP) algorithm for allocating VMs based on the principles of PEAS.

The structures we use in Fig. 2 must be updated at any moment an allocation operation occurs. *SH* is originally sorted by $peak\_peff$(based on power and utilization information from the monitoring module), but the order of a host in *SH* may change when it runs overloaded or is powered off. Hosts above their optimal utilizations will be moved to the tail. Besides, the optimal offer lists must be rebuilt before allocation as the load on hosts change from time to time. These operations are included in the method $buildLists()$,

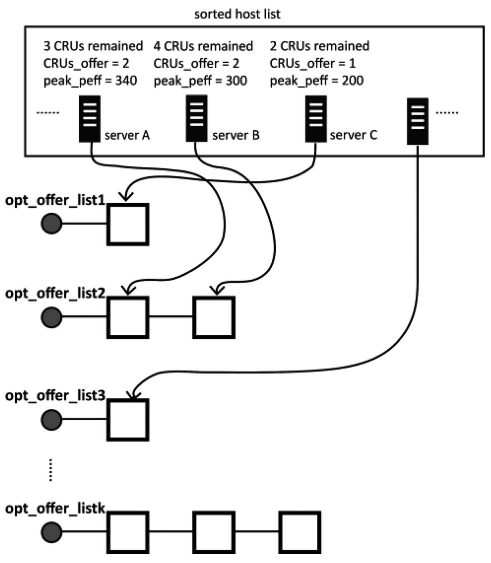


Fig. 2. Organizing physical resources using Optimal Offer Lists and the Sorted Host List.

where the host list is first sorted (line 1), followed by adding hosts to the offer lists according to their $CRUs\_offer$ (Fig. 2).

For a VM, the best host for it should be power-efficient and more importantly, likely to attain its peak efficiency after the allocation. This is the key idea of the PEAP algorithm. The pseudo-code of PEAP algorithm is shown in Algorithm 1.

Algorithm 1. Peak Efficiency Aware Placement (PEAP)

```
    Method name: peapAllocaiton
    Input: SH, v, MAX_C
    Ouput: allocMap
 1: buildLists(SH, MAX_C)
 2: $r \leftarrow CRUs\_VM_v$
 3: if $L_r$ is not empty then                              // case 1
 4:   for each $s$ in $L_r$ do
 5:     if s has sufficient resources then
 6:       allocate $v$ to $s$
 7:       allocMap.add(s,v)
 8:       $L_r$.remove(s)
 9:       return
10:     end if
11:   end for
12: end if
13: for $s$ in SH do                                        // case 2
14:   if $s$ has sufficient resources then
15:     $c\_old \leftarrow CRUs\_offer_s$
16:     allocate $v$ to $s$
17:     allocMap.add(s,v)
18:     $c\_new \leftarrow CRUs\_offer_s$
19:     $L_{c\_old}$.remove(s)
20:     $L_{c\_new}$.add(s)
21:     return
22:   end if
23: end for
24: start up a new host $s$                                 // case 3
25: allocate $v$ to $s$
26: allocMap.add(s,v)
27: $c \leftarrow CRUs\_offer_s$
28: $L_c$.add(s)
```

VMs are put into a FIFO queue and PEAP schedules each of them in an online manner. There are three cases in the PEAP algorithm: searching for a host in the corresponding optimal offer list (lines 3–10), searching in the sorted host list (lines 13–23) and activating a new host (lines 24–28). In the first case, the scheduler tries to spot a "perfect" server for the VM v based on its CRUs demand (line 3), whereas in the second case it turns to seek a host in *SH*. This may happen when there is no suitable host in the optimal offer list (L_r). A new host will be activated in case no active machine can currently host the VM (the third case). Note that every time a VM is allocated to a host, the corresponding offer list needs to be updated.

For example, assume that a VM arrives with maximum CRUs requirement equal to two and we have three servers (A, B and C) with three, four and two CRUs remaining (while their optimal CRUs to offer are two, two and one), respectively. With previous VM consolidation methods, the scheduler is likely to allocate the new VM to server C in order to make it run at full capacity. However, it leads to a sub-optimal consolidation plan regarding productivity of the whole cluster. As shown in Fig. 2, server A will attain its optimal power efficiency (defined as (15) in Section 4.1) after giving out two CRUs (with one CRU left afterwards), while that number for C is one. Meanwhile, note that server A $(\text{peak\_peff} = 340)$ is far more power efficient than server C $(\text{peak\_peff} = 200)$ when they both reach optimal power efficiency. It means that the best plan is to place the VM requiring two CRUs on server A rather than server C. Our strategy, in this example, maintains a number of ordered lists (Fig. 2) and in this case selects the first server available from the second list, say, server A. This results in the most productive state for server A while preventing performance degradation for server C.

The time complexity of finding a host for one VM instance using PEAP is $O(n)$ since the algorithm will search through *SH* in the worst case. Hence for a batch of VMs, PEAP's time complexity is $O(mn)$ where m is the number of VMs.

4.3 Virtual Machine Reallocation

The scheduler also needs to control the workload on active PMs by redistributing VMs in the cloud. Intuitively, there are two cases that we need to consider for migration. First, it is important to deal with overloaded hosts by shifting load to other machines with sufficient resources. Second, VMs on under-utilized PMs should be moved out for the sake of energy saving. In this section, we present the Peak Efficiency Aware Cost-efficient VM Reallocation (PEACR) algorithm for optimizing VMs' distribution, which is illustrated in Algorithm 2.

The main function of PEACR (Algorithm 2) is to reallocate VMs from overloaded PMs and under-utilized PMs to achieve energy efficiency. First, the algorithm checks the workload on each server. For overloaded hosts, PEACR iteratively selects a VM to migrate out from the host until its utilization decreases to a level below the pre-set upper threshold (lines 4–20). We adopt a heuristic in VM selection that reduces both the number of migrations and cost (line 9) in which we take both VM memory size and available bandwidth into account. For hosts that are under-utilized, the algorithm deactivates them after removing all the VMs

TABLE 2
Server Models

Model	cores	MIPS /core	RAM (MB)	bandwidth (Gbps)	Storage (GB)	$peak\_peff$	opt_u
AcerAR360	16	3000	24576	10	1000	160.0	0.8
IbmX3550	12	2933	12288	10	1000	142.5	1.0
FujitsuPRIMERGY TX1320M3	4	3500	16384	10	1000	282.8	0.6
HuaweiFusion CH121V3	32	3000	524288	10	1000	205.9	0.7
DellPowerEdge R630	36	2300	65536	10	1000	297.0	0.8
HPProLiant DL360G7	12	3070	16384	10	1000	172.7	0.9
HuaweiRH2288 HV2	8	2400	49152	10	1000	140.1	1.0

running on it (lines 23–26) to save power. All removed VMs are added to the set V_{out}. Finally, the placement algorithm PEAP is called to find hosts for the instances in V_{out}. Jobs in the VMs to be migrated will be dumped into memory (as a part of m_size_v) and then restored from it upon the finish of migration.

Algorithm 2. Peak Efficiency Aware Cost-efficient Reallocation (PEACR)

```
Method name: PEACR
Input: H, MAX_C, SH, W_SIZE, UP_THR
Output: $V_{out}$
 1: $V_{out} \leftarrow \emptyset$
 2: $LW\_THR \leftarrow thresholdALIQ()$
 3: for each $s$ in $H$ do
 4:     $rctload_s \leftarrow loadMovingAvg(s, W\_SIZE)$
 5:     while $rctload_s > UP\_THR$ do
 6:         $CRUs\_over_s \leftarrow (u_s - u\_opt_s) * CRU\_max_s$
 7:         $min\_ratio \leftarrow$ MAX_INT
 8:         for each $v$ in $s.VMs$ do
 9:             $ratio \leftarrow |CRUs\_over_s - CRUs\_VM_v| * m\_size_v/bw_s$
10:         if $ratio < min\_ratio$ then
11:             $min\_ratio \leftarrow ratio$
12:             $v_{out} \leftarrow v$
13:         end if
14:     end for
15:     $V_{out}$.add(v_out)
16:     deallocate v_out from $s$
17:     if $u_s < UP\_THR$ then
18:         break
19:     end if
20: end while
23: if $rctLoad_s < LW\_THR$ then
24:     $V_{out}$.add(s.VMs)
25:     deallocate all VMs from $s$
26:     deactivate $s$
26: end if
27: end for
28: for each $v$ in $V_{out}$ do
29: peapAllocation(SH, v, MAX_C)
30: end for
```

Static lower threshold has a major problem that it cannot adapt to the change of workload across the cluster. Therefore, PEACR uses a dynamic lower threshold to detect under-utilized PMs (line 2). Our basic idea is to roughly estimate the resource margin in the whole cluster and set the lower bound of utilization accordingly. Using a statistical implementation, we average the utilization of PMs whose workloads are within the inter-quartile range ($ALIQ$) and set the lower bound LW_THR as $UP\_THR - ALIQ$. For instance, assume UP_THR is set to 0.9 and the utilizations of five servers (from $s1$ to $s5$) are $0.1, 0.2, 0.6, 0.7$ and 0.9, respectively. Then we have $ALIQ = (0.2 + 0.6 + 0.7)/3 = 0.5$ and $LW\_THR = 0.9 - 0.5 = 0.4$. The $thresholdALIQ()$ method returns 0.4 and consequently, $s1$ and $s2$ will be regarded as under-utilized.

We also use an refined method to calculate PM's load by weighting the workload within recent time windows (line 4). More specifically, $loadMovingAvg()$ returns the weighted summation of the PM's utilization in a number of W_SIZE windows with predefined values of weights given in (17):

$$\omega_t = \frac{e^{-t}(1-e^{-1})}{1-e^{-W\_SIZE}},\ t = 0, 1, \ldots, W_{SIZE} - 1, \tag{17}$$

where t is the order counted from the current window to early ones. For example, suppose $W\_SIZE = 3$ and the workload sampled in the most recent windows are 0.7, 0.9 and 0.5, respectively. As a result, $loadMovingAvg()$ returns an average value approximately equal to $0.7 \times 0.67 + 0.9 \times 0.24 + 0.5 \times 0.09 = 0.73$.

To analyze time complexity of PEACR (Algorithm 2), we consider two worst cases: case 1, in which all the hosts are under-utilized (though it is unlikely to happen as we use the adaptive thresholding method) and all VMs need to be reallocated, and case 2, in which all hosts are overloaded and some VMs need to be reallocated on each host. The theoretical time complexity, in case 1, is $O(n + mn)$ where the first term is about checking utilization of each host while the second is complexity of re-allocation using PEAS. For case 2, we assume averagely one half of the VMs need to be migrated. Thus, the theoretical complexity equals to the combination of picking VMs to migrate plus reallocation, which is $O(nm + mn/2)$.

5 Experimental Evaluation

5.1 Experimental Setup

We conducted extensive experiments using CloudSim 3.0.3, a widely adopted framework for modeling cloud computing infrastructures and services. Similar to [33], we implemented the proposed PEAS strategy and extended the simulation platform with classes and functions to support our experiment. To emulate the heterogeneity of cloud data centers, totally seven models of physical machines are incorporated. The performance and power data are sampled from SPEC.[1] We set twelve types of virtual machines. The settings of

1. SPEC. http://www.spec.org/power_ssj2008/results/

TABLE 3
VM Configuration

type	cores	max CRUs	Mem (MB)	bw (Mbps)	Size (MB)	p
g1.s	1	2.1	1024	100	3000	0.10
g1.m	2	4.2	2048	100	3000	0.12
g2.m	1	2.3	1024	100	3000	0.12
g2.l	2	4.6	1920	100	3000	0.08
g2.xl	4	9.2	4096	100	3000	0.05
g2.2xl	8	18.4	8192	100	3000	0.03
m.l	2	4.6	7808	100	3000	0.03
m.xl	4	9.2	10240	100	3000	0.05
c1.l	2	5.6	1920	100	3000	0.15
c1.2xl	8	22.4	7680	100	3000	0.05
c2.l	2	5.8	1920	100	3000	0.12
c2.xl	4	11.6	3840	100	3000	0.10

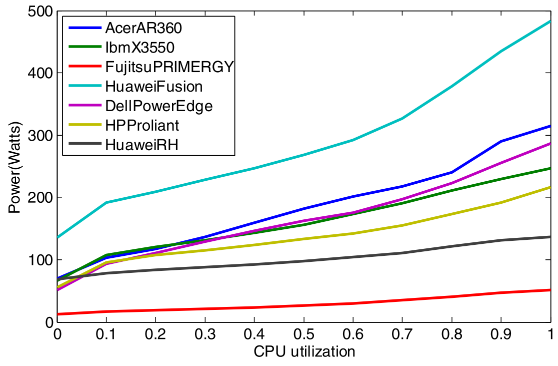


Fig. 3. Power curves of the servers (data from SPEC).

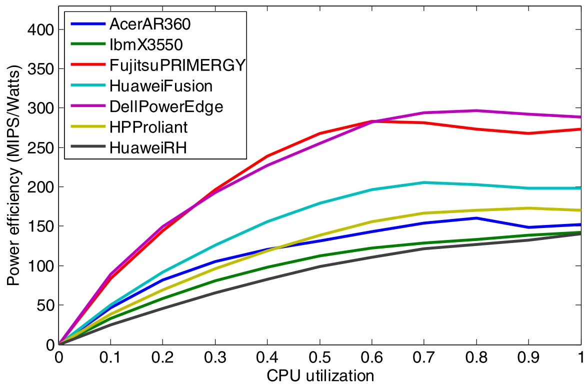


Fig. 4. Servers' power efficiency according to the data from SPEC.

physical machines and VMs are shown in Tables 2 and 3, respectively. Note that a VM's max CRUs is derived from its MIPS under maximum usage and thus can be non-integral. The unit of CRU in this work is 1000 MIPS. The p value (last column of Table 3) for each VM type indicates its popularity, which decides the probability distribution of VM instances as coming requests.

The curves of server power consumption and power efficiency are drawn in Figs. 3 and 4, respectively. From the results we got valuable observations. First, among all the models in our experiment FujitsuPRIMERGY TX1320M3 (red line) and HuaweiRH2288 HV2 (black line) consume the least amount of power. However, the power efficiency of Fujitsu is almost twice as much as that of HuaweiRH. This leads to a conclusion that low-power machines are not always power efficient. Second, two physical servers can have remarkable difference in power efficiency. We can see, for instance, DellPowerEdge R630 and FujitsuPRIMERGY TX1320M3 are obviously superior in power efficiency. It means that given the same amount of power, they are much more productive than other five server models. Last but not least, from Fig. 4 we can see that the power efficiency of physical machines peaks at different CPU utilizations. For example, IbmX3550 are most power efficient when fully utilized (100 percent) while the optimal utilization for HuaweiFusion CH121V3 is 70 percent.

TABLE 4
Cluster Settings

Scale	# of servers	# of VMs
Small	30	150
Medium	100	530
Large	300	1750

TABLE 5
Job Settings

Job	Workload source	Load range	Average
Light	Overlay5	[0.00, 0.35]	0.21
Moderate	Oneswarm	[0.11, 0.64]	0.44
Heavy	P2pns	[0.17, 0.95]	0.81

Considering the complex environment in cloud computing, we set three scales of clusters (Table 4) and three types of jobs with different load characteristics (Table 5). We assume that jobs are all independent and their lengths (measured in million instructions) follow Poisson distribution. We use real-world workload traces in our experiments. Overlay5, Oneswarm and P2pns are public workload traces from PlanetLab[2] VMs, which are contained in Cloudsim 3.0.3 open-source package[3] by default.

5.2 Experimental Results

We evaluated the proposed PEAS strategy against three energy-aware algorithms for comparison, namely PABFD+MM, PABFD+MMT and TVRSM [12]. Both adopting PABFD for VM placement, PABFD+MM and PABFD+MMT use different strategies in offloading VMs from over-utilized hosts. Minimization of Migrations (MM) policy [29] always selects the VM by deallocating which the host's workload will stay below and closest to the upper threshold. Minimal Migration Time (MMT) [11] policy removes the VM that requires the least amount of time to complete migration relatively to other VMs on the host. We choose these three algorithms as baselines as they support online consolidation and prove to be effective in data center energy conservation. The strategy of TVRSM adopts HVRAA in initial placement. For overloaded hosts (hard threshold is used for overload detection), TVRSM picks and migrates VMs out from them according to their Multiple Correlation Coefficient (MCC) [34] factors. For VMs migrated out, TVRSM allocates them to new hosts using a

2. https://www.planet-lab.org/
3. https://github.com/beloglazov/planetlab-workload-traces

TABLE 6
A Summarization of Energy Consumption, Average Job Run Time, and QoS Metrics for VM Consolidation Algorithms (Small Cluster)

Group Name	Algorithm	Energy consumption (kWh)	Avg. run time (min.)	SLAV per host/VM	Migrations	Overhead (ms)	CWEE (MI/Joules)
Small-light	PABFD+MM	18.27	949.17	0.57%/15.87%	555	0.63	49.26
	PABFD+MMT	19.15	976.57	0.36%/18.06%	389	**0.57**	46.99
	TVRSM	17.78	967.83	0.68%/18.23%	**289**	1.83	50.62
	PEAS	**13.95**	**877.07**	**0.42%/9.58%**	664	1.47	**64.52**
Small-moderate	PABFD+MM	14.44	378.47	5.80%/4.62%	1236	0.42	62.33
	PABFD+MMT	14.81	380.60	5.56%/4.20%	1307	**0.31**	60.77
	TVRSM	13.54	378.44	4.57%/7.96%	767	1.69	66.47
	PEAS	**12.34**	**371.71**	**0.31%/0.23%**	**103**	0.32	**72.93**
Small-heavy	PABFD+MM	13.96	222.37	9.41%/2.86%	825	0.43	64.47
	PABFD+MMT	13.98	**222.30**	6.18%/1.19%	686	0.65	64.38
	TVRSM	14.71	222.70	4.52%/1.90%	375	1.98	61.18
	PEAS	**13.11**	222.40	**0.80%/0.26%**	**155**	**0.43**	**68.65**

TABLE 7
A Summarization of Energy Consumption, Average Job Run Time, and QoS Metrics for VM Consolidation Algorithms (Medium Cluster)

Group Name	Algorithm	Energy consumption (kWh)	Avg. run time (min.)	SLAV per host/VM	Migrations	Overhead (ms)	CWEE (MI/Joules)
Medium-light	PABFD+MM	66.95	1055.36	1.35%/24.46%	3998	**0.61**	47.49
	PABFD+MMT	70.66	1133.86	1.23%/29.69%	3878	0.92	45.00
	TVRSM	66.32	1020.15	1.39%/23.66%	**2969**	10.52	47.95
	PEAS	**46.77**	**871.10**	**0.34%/9.28%**	3616	3.45	**67.99**
Medium-moderate	PABFD+MM	48.33	376.97	8.62%/8.25%	5625	0.39	65.80
	PABFD+MMT	50.15	380.20	7.56%/8.04%	5414	0.41	63.41
	TVRSM	48.29	373.78	3.87%/5.29%	1702	11.25	65.85
	PEAS	**42.47**	**373.70**	**0.09%/0.93%**	**201**	**0.38**	**74.88**
Medium-heavy	PABFD+MM	47.91	221.88	12.06%/6.71%	3263	**0.23**	66.37
	PABFD+MMT	48.59	**221.69**	10.30%/3.74%	2992	0.38	65.45
	TVRSM	56.34	222.95	4.67%/2.50%	1258	11.75	56.44
	PEAS	**45.80**	222.26	**0.53%/0.34%**	**423**	0.58	**69.40**

TABLE 8
A Summarization of Energy Consumption, Average Job Run Time, and QoS Metrics for VM Consolidation Algorithms (Large Cluster)

Group Name	Algorithm	Energy consumption (kWh)	Avg. run time (min.)	SLAV per host/VM	Migrations	Overhead (ms)	CWEE (MI/Joules)
Large-light	PABFD+MM	224.77	1088.73	2.60%/28.10%	24401	**1.72**	46.71
	PABFD+MMT	233.66	1147.81	2.42%/31.64%	22772	2.28	44.94
	TVRSM	196.27	**869.15**	0.24%/10.09%	**4515**	114.07	53.49
	PEAS	**152.79**	876.40	**0.28%/9.91%**	14479	8.51	**68.72**
Large-moderate	PABFD+MM	158.54	377.50	10.34%/9.82%	19618	1.17	66.23
	PABFD+MMT	160.62	378.66	8.76%/9.91%	18836	1.09	65.37
	TVRSM	179.85	377.49	5.06%/7.79%	7947	77.91	58.38
	PEAS	**140.34**	**375.31**	**0.01%/1.50%**	**484**	**0.68**	**74.82**
Large-heavy	PABFD+MM	162.79	221.90	12.37%/7.19%	10578	0.74	64.50
	PABFD+MMT	161.91	**221.65**	12.16%/5.93	10264	**0.72**	64.85
	TVRSM	175.09	221.89	7.20%/3.88%	5175	103.92	59.97
	PEAS	**155.24**	222.26	**0.56%/0.47%**	**1624**	0.9	**67.64**

strategy named MPIS (Minimum Power Increasing Strategy), which prioritizes physical machines according to their power increments after allocation. The parameters for TVRSM in our experiment are set to the same as [12].

Without loss of generality, totally nine groups of experiments were conducted to simulate all possible combinations of clusters and workload intensity settings listed in Table 4 and Table 5. We discuss the performance of migration-enabled, energy-aware algorithms (PEAS, PABFD+MM, PABFD+MMT and TVRSM) in respect to energy consumption, energy efficiency, average job run time as well as multiple QoS metrics. We choose average run time as VM performance metric because it clearly reveals the performance impact by different strategies in the case where VMs are running jobs. For other types of requests (Fig. 1), this metric may not be applicable. Experiment results are shown in Tables 6, 7, and 8.

5.2.1 Energy Consumption

By extracting the data from Tables 6, 7 and 8, cluster energy consumptions upon finishing all the jobs with different schedulers are shown in Fig. 5. Among all the consolidation algorithms, PEAS achieves the highest energy efficiency and averagely outperformed PABFD+MM, PABFD+MMT and TVRSM by 15.43, 17.36 and 17.46 percent, respectively. We observe that PEAS saved energy most effectively for VMs under relatively light workload in medium and large clusters. This is because PEAS can best utilize productive PMs in

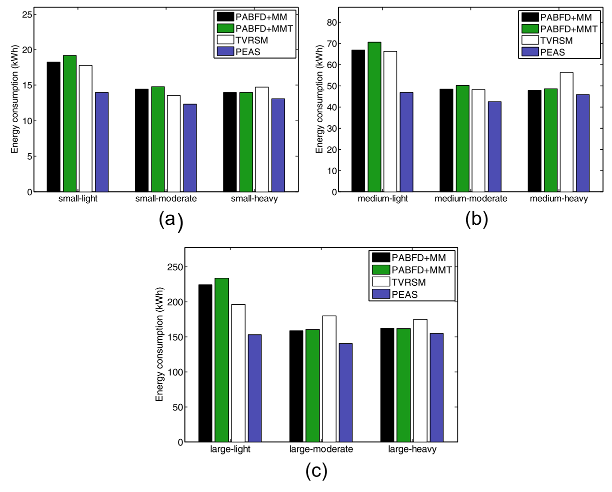


Fig. 5. Comparing VM consolidation strategies in total energy consumption with migration enabled in the (a) small cluster, (b) medium cluster and (c) large cluster.

an environment with ample resources. For the same reason the differences between the algorithms were relatively small in the groups where CPU-intensive jobs were applied.

Inspired by the definition of Workload Power Efficiency [52], we design a holistic metric for measuring the energy efficiency of server clusters. Cluster Workload Energy Efficiency (CWEE) is defined as the ratio of total workload finished to the amount of energy consumed by the cluster

$$\text{CWEE} = \frac{\text{Amount of work finished}}{\text{Cluster energy consumption}}, \tag{18}$$

where the amount of work finished is defined as the sum of compute resource requests from every virtual machines created. Cluster energy consumption is the sum of every single server's consumption. Both are collected using counters throughout the simulation. In our case we use million instructions and Joules to measure the amount of workload and energy consumption, respectively. High CWEE implies a great energy-efficiency state that the cluster is working in.

As shown in the last columns of Tables 6, 7 and 8, PEAS make significant improvement on CWEE, indicating that our strategy, in comparison with other algorithms, is making the cluster more productive given equal amount of energy. The reason behind is that PEAS is more inclined to consolidate VMs on energy-efficient servers whilst maintaining them at peak efficiency.

5.2.2 *Job Run Time*

In this paper, we choose to demonstrate the effectiveness of our strategy by looking at jobs (run in VMs) because a great portion of them (especially batch jobs) are fixed in amount of work. As a result, it is easier to quantify and analyze the VM performance impact through comparing the runtime of the same jobs under different consolidation strategies. Fig. 6 demonstrates the average job run time when we applied different VM consolidation strategies to the specified clusters. When job workload is moderate or heavy, we can observe only tiny differences of a few minutes between PEAS and other algorithms. But in the case that VMs are running light workload, PEAS remarkably reduces average job run time compared

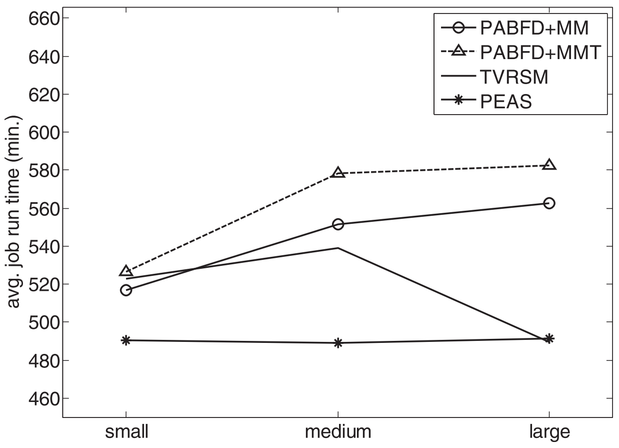


Fig. 6. Comparing VM consolidation algorithms in average job run time grouped by workload intensity.

with PABFD+MM and PABFD+MMT. For example, PEAS shortens the average time to complete a job by 184.26, 262.76 and 149.05 minutes when compared with PABFD+MM, PABFD+MMT and TVRSM in the medium-light group, indicating a performance promotion of the cloud system. Although we cannot evaluate strategies in run time for long running applications, the result infers that VMs are expected to obtain more sufficient share of resources under the scheduling of PEAS.

5.2.3 *QoS Metrics*

From the result shown in Table 6, 7 and 8, we can see that PEAS produces the lowest SLA violation rate, whereas PABFD+MM and PABFD+MMT suffers much higher risk of service quality degradation especially in the medium-light and large-light groups (Tables 7 and 8). In particular, PEAS warrants satisfactory QoS for CPU-intensive jobs (i.e., moderate or heavy workload) in all the specified cluster scales with SLA violations per host/VM below 1.0 percent in most cases.

Although TVRSM triggers fewer VM migrations than PEAS under light workload (group "s-l", "m-l" and "l-l" in Fig. 7), PEAS induces much fewer migration operations than TVRSM, PABFD+MM and PABFD+MMT in other test cases. This is due to the non-aggressive policy we adopt in VM placement. In other words, PEAS hardly consolidates

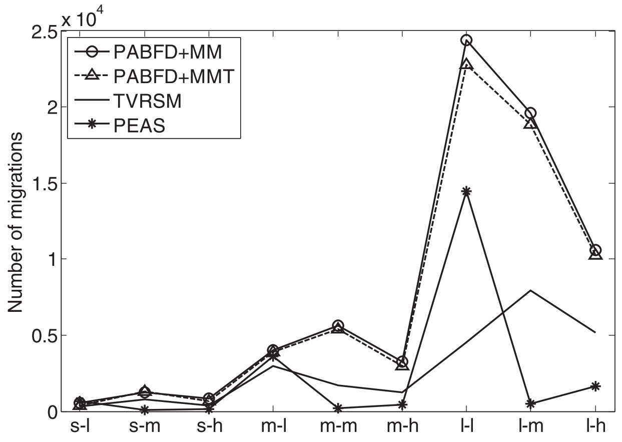


Fig. 7. Number of migrations incurred by PEAS, PABFD+MM, PABFD+MMT and TVRSM.

VMs on hosts that are nearly overloaded, leaving relatively sufficient space for changing workloads. Besides, PEAS is of low time complexity. From Tables 6, 7, and 8, it can be found that PEAS takes only slightly more time than PABFD+MM/MMT in VM allocation, whilst TVRSM produced large scheduling overheads in the large cluster. The result also infers that the overheads by PEAS does not increase dramatically as cloud services scale out.

6 Conclusions

It has been a prominent concern for CSPs to promote cloud services' energy efficiency while improving the service quality in the meantime. In this paper, we first discuss the trade-off between system performance and energy reservation from the perspective of server clusters, present a dynamic VM (re)allocation framework for offering flexible services, and provide problem definition for VM consolidation. Then we propose a novel VM consolidation strategy, namely PEAS, with the aim at achieving dual improvement in performance and energy efficiency. PEAS takes advantage of servers' peak power efficiency by maintaining active machines under their optimal utilizations whenever possible. We evaluated PEAS with several VM placement and migration algorithms/strategies like PABFD and TVRSM. Experimental results show that PEAS effectively reduces cluster energy consumption, shortens average job run time and improves multiple QoS metrics.

In future research, we plan to investigate how to optimize task scheduling and resource provisioning in edge computing and fog computing, which are emerging together with challenges. We will also work on applying machine learning to profiling/predicting job workloads and metering VM-level power consumption [48] in order to enable proactive resource provisioning and VM consolidation in complex cloud environments.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant Nos. 61772205 and 61872084), Guangdong Science and Technology Department (Grant No. 2017B010126002), Guangzhou Science and Technology Program key projects (Grant Nos. 201802010010, 201807010052, 201902010040, and 201907010001), Nansha Science and Technology Projects (Grant No. 2017GJ001), Guangzhou Development Zone Science and Technology (Grant No. 2018GH17), the Fundamental Research Funds for the Central Universities, SCUT (Grant No. 2019ZD26), Major Program of Guangdong Basic and Applied Research (2019B030302002). W. Lin and W. Wu have equally contributed to this article.

References

[1] Z. Zhang and S. Fu, "Characterizing power and energy usage in cloud computing systems," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Technol. Sci.*, 2012, pp. 146–153.

[2] J. Koomey, "Growth in data center electricity use 2005 to 2010," Analytics Press, Oakland, CA, Tech. Rep., 2011.

[3] A. Venkatraman, "Global census shows datacenter power demand grew 63% in 2012," *ComputerWeekly*, Oct. 2012, [Online] Available: http://www.computerweekly.com/news/2240164589/Datacentre-power-demand-grew-63-in-2012-Global-datacentre-census.

[4] F. Juarez, J. Ejarque, and R. M. Badia, "Dynamic energy-aware scheduling for parallel task-based application in cloud computing," *Future Gener. Comput. Syst.*, vol. 78, pp. 257–271, 2016.

[5] L. Teylo, U. de Paula, Y. Frota, D. de Oliveira, and L. M. Drummond, "A hybrid evolutionary algorithm for task scheduling and data assignment of data-intensive scientific workflows on clouds," *Future Gener. Comput. Syst.*, vol. 76, pp. 1–17, 2017.

[6] N. Zhang, X. Yang, M. Zhang, Y. Sun, and K. Long, "A genetic algorithm-based task scheduling for cloud resource crowd-funding model," *Int. J. Commun. Syst.*, vol. 5, pp. 1–10, 2017.

[7] H. Duan, C. Chen, G. Min, and Y. Wu, "Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems," *Future Gener. Comput. Syst.*, vol. 74, pp. 142–150, 2016.

[8] Y. Chen *et al.*, "Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS," *Future Gener. Comput. Syst.*, 2017. [Online]. Available: https://doi.org/10.1016/j.future.2017.09.024

[9] W. Lin, W. Wu, and J. Z. Wang, "A heuristic task scheduling algorithm for heterogeneous virtual clusters," *Scientific Program.*, vol. 2016, pp. 1–10, 2016, doi: 10.1155/2016/7040276.

[10] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *Proc. Int. Conf. High Perform. Comput. Simul.*, 2009, pp. 1–11.

[11] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput. Practice Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[12] W. Zhu, Y. Zhuang, and L. Zhang, "A three-dimensional virtual resource scheduling method for energy saving in cloud computing," *Future Gener. Comput. Syst.*, vol. 69, pp. 66–74, 2017.

[13] K. Li, H. Zheng, and J. Wu, "Migration-based virtual machine placement in cloud systems," in *Proc. IEEE Int. Conf. Cloud Netw.*, 2014, pp. 83–90.

[14] D. Yokoyama, B. Schulze, H. Kloh, M. Bandini, and V. Rebello, "Affinity aware scheduling model of cluster nodes in private clouds," *J. Netw. Comput. Appl.*, vol. 95, pp. 94–104, 2017.

[15] W. Lin, S. Xu, J. Li, L. Xu, and Z. Peng, "Design and theoretical analysis of virtual machine placement algorithm based on peak workload characteristics," *Soft Comput.*, vol. 21, no. 5, pp. 1301–1314, 2017.

[16] M. Uddin, Y. Darabidarabkhani, A. Shah, and J. Memon, "Evaluating power efficient algorithms for efficiency and carbon emissions in cloud data centers: a review," *Renewable Sustain. Energy Rev.*, vol. 51, no. 11, pp. 1553–1563, 2015.

[17] W. Wu, W. Lin, C. Hsu, and L. He, "Energy-efficient hadoop for big data analytics and computing: A systematic review and research insights," *Future Gener. Comput. Syst.*, 86, 1351–1367, 2018, [Online]. Available: https://doi.org/10.1016/j.future.2017.11.010.

[18] H. Rong, H. Zhang, S. Xiao, C. Li, and C. Hu, "Optimizing energy consumption for data centers," *Renewable Sustain. Energy Rev.*, vol. 58, pp. 674–691, 2016.

[19] D. Borgetto, M. Maurer, G. Da-Costa, J. M. Pierson, and I. Brandic, "Energy-efficient and SLA-aware management of IaaS clouds," in *Proc. 3rd Int. Conf. Future Energy Syst.: Where Energy Comput. Commun. Meet*, May 2012, pp. 25–34.

[20] L. Bölöni and D. Turgut, "Value of information based scheduling of cloud computing resources," *Future Gener. Comput. Syst.*, vol. 71, pp. 212–220, 2017.

[21] D. G. Lago, E. R. M. Madeira, and D. Medhi, "Energy-aware virtual machine scheduling on data centers with heterogeneous bandwidths," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 1, pp. 83–98, Jan. 2018.

[22] J. Lartigau, X. Xu, and D. Zhan, "Artificial bee colony optimized scheduling framework based on resource service availability in cloud manufacturing," in *Proc. Int. Conf. Service Sciences*, 2014, pp. 181–186.

[23] K. M. Cho, P. W. Tsai, C. W. Tsai, and C. S. Yang, "A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing," *Neural Comput. Appl.*, vol. 26, no. 6, pp. 1–13, 2015.

[24] A. J. Rubio-Montero, E. Huedo, and R. Mayo-García, "Scheduling multiple virtual environments in cloud federations for distributed calculations," *Future Gener. Comput. Syst.*, vol. 74, pp. 90–103, 2016.

[25] Z. Xiao, P. Liang, Z. Tong, K. Li, S. U Khan, and K. Li, "Self-adaptation and mutual adaptation for distributed scheduling in benevolent clouds," *Concurrency Comput. Practice Experience*, vol. 29, no. 5, 2016, Art. no. e3939.

[26] J. L. Berral *et al.*, "Towards energy-aware scheduling in data centers using machine learning," in *Proc. ACM 1st Int. Conf. Energy-Efficient Comput. Netw.*, 2010, pp. 215–224.

[27] C. Pettey and L. Goasduff, "Gartner says worldwide public cloud services market to grow 18 percent in 2017," [Online] Available: https://www.gartner.com/en/newsroom/press-releases/2017-02-22-gartner-says-worldwide-public-cloud-services-market-to-grow-18-percent-in-2017

[28] A. Berl *et al.*, "Energy-efficient cloud computing," *Comput. J.*, vol. 53, no. 7, pp. 1045–1051, 2010.

[29] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.

[30] C. H. Hsu and S. W. Poole, "Power signature analysis of the SPECpower_ssj2008 benchmark," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw.*, 2011, pp. 227–236.

[31] J. Xu and J. Fortes, "A multi-objective approach to virtual machine management in datacenters," in *Proc. 8th ACM Int. Conf. Auton. Comput.*, 2011, pp. 225–234.

[32] L. Mashayekhy, M. M. Nejad, D. Grosu, and Q. Zhang, "Energy-aware scheduling of mapreduce jobs for big data applications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 10, pp. 2720–2733, Oct. 2015.

[33] W. Lin, S. Xu, L. He, and J. Li, "Multi-resource scheduling and power simulation for cloud computing," *Inf. Sci.*, vol. 397, pp. 168–186, 2017.

[34] H. Abdi, "Multiple correlation coefficient," *Encyclopedia of Measurement and Statistics*, Thousand Oaks, CA, USA: SAGE, 2007, pp. 648–651.

[35] X. Li, P. Garraghan, X. Jiang, Z. Wu, and J. Xu, "Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 6, pp. 1317–1331, Jun. 2017.

[36] A. Tarafdar, S. Khatua, and R. K. Das, "QoS aware energy efficient VM consolidation techniques for a virtualized data center," in *Proc. IEEE/ACM 11th Int. Conf. Utility Cloud Comput.*, Dec. 2018, pp. 114–123.

[37] K. Chang, S. Park, H. Kong, and W. Kim, "Optimizing energy consumption for a performance-aware cloud data center in the public sector," *Sustain. Comput.: Inform. Syst.*, vol. 20, pp. 34–45, 2018.

[38] I. Hwang and M. Pedram, "Hierarchical, portfolio theory-based virtual machine consolidation in a compute cloud," *IEEE Trans. Services Comput.*, vol. 11, no 1, pp. 63–77, Jan./Feb. 2018.

[39] F. Tao, C. Li, T. W. Liao, and Y. Laili, "BGM-BLA: A new algorithm for dynamic migration of virtual machines in cloud computing," *IEEE Trans. Services Comput.*, vol. 9, no. 6, pp. 910–925, Nov./Dec. 2016.

[40] J. O. Gutierrez-Garcia, and A. Ramirez-Nafarrate, "Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 916–929, Nov./Dec. 2015.

[41] L. Wu, S. K. Garg, S. Versteeg, and R. Buyya, "SLA-based resource provisioning for hosted software-as-a-service applications in cloud computing environments," *IEEE Trans. Services Comput.*, vol. 7, no. 3, pp. 465–485, Jul.–Sep. 2014.

[42] L. Wang, Y. Ma, J. Yan, V. Chang, and A. Y. Zomaya, "pipsCloud: High performance cloud computing for remote sensing big data management and processing," *Future Gener. Comput. Syst.*, vol. 78, pp. 353–368, 2018.

[43] D. B. Muralitharan, S. A. B. Reebha, and D. Saravanan, "Optimization of performance and scheduling of HPC applications in cloud using cloudsim and scheduling approach," in *Proc. Int. Conf. IoT Appl.*, 2017, pp. 1–6.

[44] C. Gao, H. Wang, L. Zhai, Y. Gao, and S. Yi, "An energy-aware ant colony algorithm for network-aware virtual machine placement in cloud computing," in *Proc. IEEE 22nd Int. Conf. Parallel. Distrib. Syst.*, 2016, pp. 669–676.

[45] W. Zhu, Y. Zhuang, and L. Zhang, "A three-dimensional virtual resource scheduling method for energy saving in cloud computing," *Future Gener. Comput. Syst.*, vol. 69, pp. 66–74, 2017.

[46] X. F. Liu, Z. H. Zhan, J. D. Deng, Y. Li, T. Gu, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 113–128, Feb. 2018.

[47] V. D. Reddy, B. Setz, G. S. V. Rao, G. R. Gangadharan, and M. Aiello, "Metrics for sustainable data centers," *IEEE Trans. Sustain. Comput.*, vol. 2, no. 3, pp. 290–303, Jul.-Sep. 2017.

[48] A. Narayan and S. Rao, "Power-aware cloud metering," *IEEE Trans. Services Comput.*, vol. 7, no. 3, pp. 440–451, Jul.-Sep. 2014.

[49] R. Basmadjian, N. Ali, F. Niedermeier, H. De Meer, and G. Giuliani, "A methodology to predict the power consumption of servers in data centres," in *Proc. ACM 2nd Int. Conf. Energy-Efficient Comput. Netw.*, 2011, pp. 1–10.

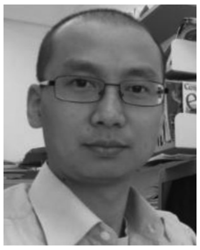
[50] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J. M. Pierson, and A. V. Vasilakos, "Cloud computing: Survey on energy efficiency," *ACM Comput. Surveys*, vol. 47, no. 2, 2015, Art. no. 33.

[51] S. Rivoire, M. A. Shah, P. Ranganathan, C. Kozyrakis, and J. Meza, "Models and metrics to enable energy-efficiency optimizations," *Comput.*, vol. 40, no. 12, pp. 39–48, 2007.

[52] T. Wilde *et al.*, "DWPE, a new data center energy-efficiency metric bridging the gap between infrastructure and workload," in *Proc. Int. Conf. High Perform. Comput. Simul.* 2014, pp. 893–901.

Weiwei Lin received the BS and MS degrees from Nanchang University, Nanchang, China, in 2001 and 2004, respectively, and the PhD degree in computer application from South China University of Technology, Guangzhou, China, in 2007. Currently, he is a professor with the School of Computer Science and Engineering, South China University of Technology. His research interests include distributed systems, cloud computing, big data computing and AI application technologies. He has published more than 80 papers in refereed journals and conference proceedings. He is a senior member of the CCF.

Wentai Wu received the bachelor's and master's degrees in computer science from the South China University of Technology, Taipei, Taiwan, in 2015 and 2018, respectively. Currently, he is working toward the PhD degree supervised by Dr. Ligang He in the University of Warwick, Coventry, United Kingdom. His research interests mainly include parallel and distributed computing, energy-efficient computing, predictive analytics and distributed machine learning. He is a student member of the IEEE.



Ligang He received the PhD degree in computer science from the University of Warwick, Coventry, United Kingdom, and worked as a postdoctoral researcher at the University of Cambridge, Cambridge, UK. From 2006, he worked in the Department of Computer Science at the University of Warwick, Coventry, U.K., as an assistant professor, and then became an associate professor. He is currently a reader with the department. His research interests focus on parallel and distributed processing, cluster, grid, and cloud computing. He has published more than 100 papers in international conferences and journals, such as the *IEEE Transactions on Parallel and Distributed Systems*, IPDPS, CCGrid, and MAS-COTS. He has been a co-chair or a member of the program committee for a number of international conferences, and been the reviewers for many international journals, including *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, etc. He is a member of the IEEE.