Hindawi Publishing Corporation
Scientific Programming
Volume 2016, Article ID 6208358, 13 pages
http://dx.doi.org/10.1155/2016/6208358

Research Article

Task Classification Based Energy-Aware Consolidation in Clouds

HeeSeok Choi,[1] JongBeom Lim,[2] Heonchang Yu,[1] and EunYoung Lee[3]

[1]*Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea*
[2]*IT Convergence Education Center, Dongguk University, Seoul, Republic of Korea*
[3]*Department of Computer Science, Dongduk Women's University, Seoul, Republic of Korea*

Correspondence should be addressed to EunYoung Lee; elee@dongduk.ac.kr

Received 22 January 2016; Accepted 3 August 2016

Academic Editor: Zhihui Du

We consider a cloud data center, in which the service provider supplies virtual machines (VMs) on hosts or physical machines (PMs) to its subscribers for computation in an on-demand fashion. For the cloud data center, we propose a task consolidation algorithm based on task classification (i.e., computation-intensive and data-intensive) and resource utilization (e.g., CPU and RAM). Furthermore, we design a VM consolidation algorithm to balance task execution time and energy consumption without violating a predefined service level agreement (SLA). Unlike the existing research on VM consolidation or scheduling that applies none or single threshold schemes, we focus on a double threshold (upper and lower) scheme, which is used for VM consolidation. More specifically, when a host operates with resource utilization below the lower threshold, all the VMs on the host will be scheduled to be migrated to other hosts and then the host will be powered down, while when a host operates with resource utilization above the upper threshold, a VM will be migrated to avoid using 100% of resource utilization. Based on experimental performance evaluations with real-world traces, we prove that our task classification based energy-aware consolidation algorithm (TCEA) achieves a significant energy reduction without incurring predefined SLA violations.

1. Introduction

Nowadays, cloud computing has become an efficient paradigm of offering computational capabilities as a service based on a pay-as-you-go model [1] and many studies have been conducted in diverse cloud computing research areas, such as fault tolerance and quality of service (QoS) [2, 3]. Meanwhile, virtualization has been touted as a revolutionary technology to transform cloud data centers (e.g., Amazon's elastic compute cloud and Google's compute engine) [4]. By taking advantage of the virtualization technology, running cloud applications on virtual machines (VMs) has become an efficient solution of consolidating data centers because the utilization rate of data centers has been found to be low, typically ranging from 10 to 20 percent [5]. In other words, a single host (physical machine) can run multiple VMs simultaneously and VMs can be relocated dynamically by live migration operations, leading to high resource utilization. Another issue of data centers is high energy consumption, which results in substantial carbon dioxide emissions (about 2 percent of the global emissions). A typical data center consumes as much energy as 25,000 households do [6]. In this regard, an efficient energy consumption strategy in nonvirtualization environments (smart grids) has been carried out [7].

As the virtualization technology [8, 9] has become popular widely, organizations or companies began to build their own private cloud data centers using commodity hardware. In this regard, there exists a need for designing more efficient and effective VM consolidation techniques to reduce energy consumption in cloud data centers. The simplest way to achieve energy reduction in cloud computing environments is to minimize the number of physical machines (PMs) by allocating more VMs in a PM. However, this solution may lead to a high degree of service level agreement (SLA) violations when each VM requires the host's limited resources. Moreover, the relationship between CPU utilization and power consumption is not linear as shown in Figure 1. The power consumption of CPU increases more than linearly as utilization increases. More importantly, when the CPU utilization is above 90%, the power consumption jumps up quickly due to the architectural design and turbo boost

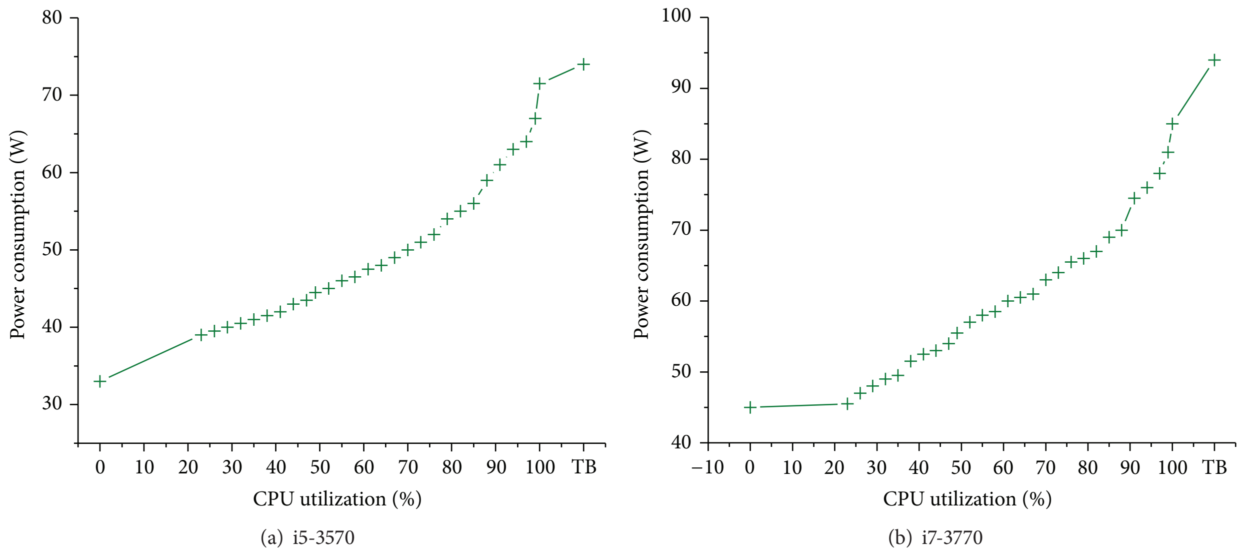


Figure 1: Energy consumption of i5 and i7 CPUs (TB indicates turbo boost).

feature. In other words, the performance to power ratio [10] exhibits sublinear growth, and therefore, just putting many VMs to a PM utilizing 100% of CPU is not always the best solution in terms of performance, energy consumption, and SLA violations. We take Intel i5 and i7 CPUs in our experiments, rather than server class CPUs in Figure 1, because, for small and medium sized companies, using commodity hardware like Intel i5 or i7 to build a private cloud is more affordable and accessible [11].

In this paper, we present a new VM and task consolidation mechanism in cloud computing environments. The proposed method is based on task classification, in which we divide cloud tasks into two categories: computation-intensive and data-intensive tasks. A computation-intensive task refers to a computation-bounded application program. Such applications devote most of their execution time to fulfill computational requirements as opposed to I/O and typically require small volumes of data, while a data-intensive task refers an I/O-bounded application with a need to process large volumes of data. Such applications devote most of their processing time to I/O, movement, and manipulation of data. The basic idea of our approach is twofold. One is that when we need to migrate cloud tasks due to a migration policy, we favor a computation-intensive task for migration rather than a data-intensive task since the migration time for computation-intensive tasks is shorter than that of data-intensive tasks. In order to migrate data-intensive tasks, it is necessary to move data for processing as well, and this transferring of data generates communication overheads. Then, we prefer the target VM with no computation-intensive tasks because data-intensive tasks consume less CPU resources, thereby providing a comfortable executing environment for the computation-intensive task. The other is to use a double threshold approach (i.e., upper threshold and lower threshold) for VM migrations and optimization. When a VM's utilization is either above the upper threshold or below the lower threshold, the VM is scheduled for migration. Our double threshold approach is different from previous work in that no algorithm is proposed to use the upper and lower thresholds simultaneously in an effective way to the best of our knowledge. With an extensive measurement observation, we identified that there is much room for optimization by balancing performance and energy consumption.

Our work differs from traditional scheduling algorithms in the literature by designing and implementing a novel consolidation mechanism based on a task classification approach. We develop corresponding task scheduling and VM allocation algorithms for cloud tasks executed in virtualized data centers.

The major contributions of this paper are summarized as follows:

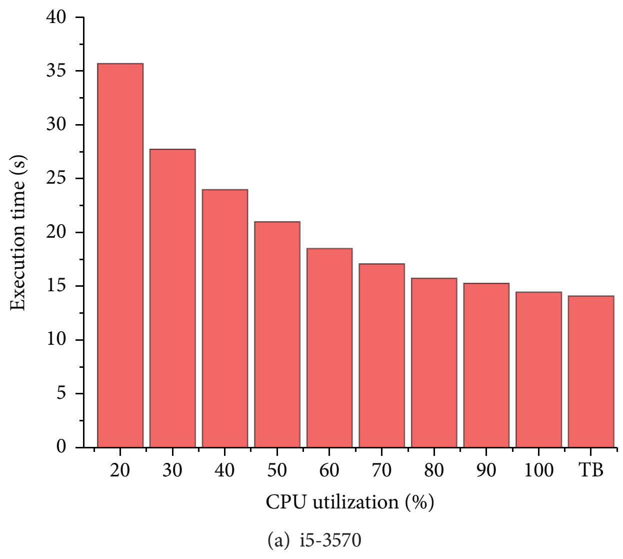
(i) We designed an energy-aware cloud data center consolidation mechanism based on task classification, while preserving performance and SLA guarantee.

(ii) We developed a cloud task scheduling and VM allocation algorithms that solve problems about when and how to migrate tasks and VMs in an energy efficient way.

(iii) We formulated a double threshold algorithm for further optimization to improve the performance to power ratio.

(iv) We undertook a comprehensive analysis and performance evaluation based on real-world workload traces.

The rest of this paper is organized as follows. Section 2 describes our research motivation and our intuition for consolidation in virtualized clouds. In Section 3, the task classification based energy-aware consolidation scheduling mechanism and the main principles behind it are presented. The experiments and performance analysis are given in Section 4. The related work in the literature is summarized in Section 5. Finally, Section 6 concludes the paper.



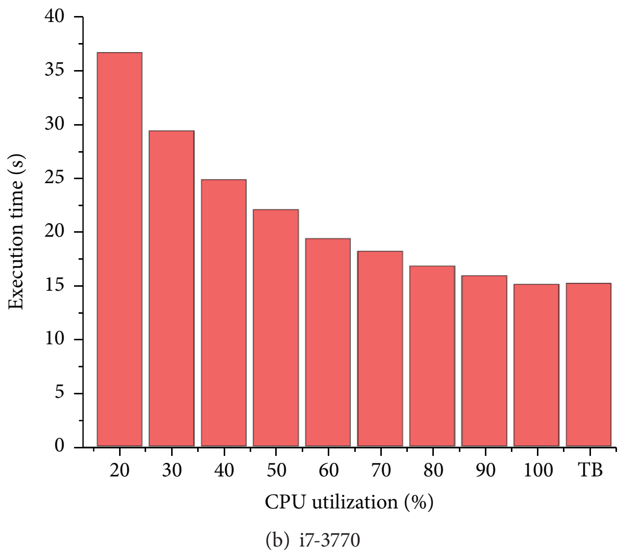


FIGURE 2: Energy consumption and execution time of matrix multiplication of i5 and i7 CPUs.

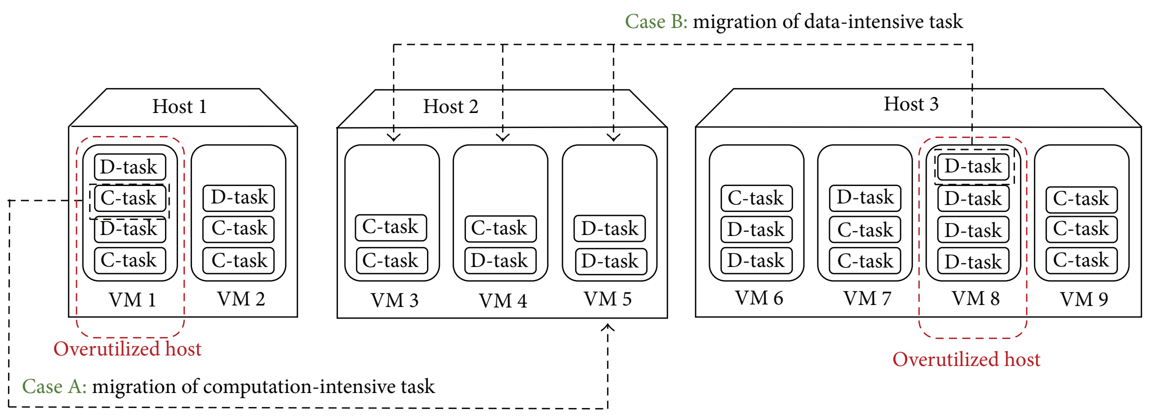


FIGURE 3: An illustrative example of TCEA.

2. Motivation and the Basic Idea

As the virtualization technology has been widely used, it is easily possible to construct a private cloud computing environment with open-source infrastructure as a service (IaaS) solutions and commodity hardware (e.g., desktop-level CPUs and peripherals). Figure 2 shows execution time of a matrix multiplication benchmark program and its performance to power ratio with CPU utilization for Intel i5-3570 and i7-3770 CPUs. With CPU utilization below 50%, the performance gain from the CPUs is noticeable as CPU utilization increases as the performance to power ratio indicates. However, when CPU utilization is above 50% the performance to power ratio grows sublinearly. This means that using high CPU utilization is not always an energy-efficient way to perform tasks. Even when we use a turbo boost feature, one of dynamic voltage and frequency scaling (DVFS) techniques, the performance gain of high frequency of CPU operations is not big considering the performance to power ratio.

Hence, we devise another approach using a threshold of CPU utilization so that a host that manages a couple of VMs does not exceed a predefined CPU utilization threshold. When a host exceeds the threshold, our consolidation algorithm determines to migrate one of the tasks or VMs on the host to another as depicted in Figure 3. Each task is categorized as C-task (computation-intensive task) or D-task (data-intensive task) and is assumed to use 25% of resources or utilization for a VM for simplicity in this example. Note that the task categorization mechanism of C-task and D-task is explained in the next section. Assuming that the threshold is 75% for a VM, tasks in VM 1 and VM 8 should be migrated to underutilized VMs. For Case A, in which there are C-tasks and D-tasks in a VM, our consolidation algorithm chooses a C-task to be migrated and preferentially selects a target VM with no C-tasks since migrating a C-task takes much shorter time compared to a D-task and migrating a D-task introduces a major I/O bottleneck in the host. For Case B, in which there are only D-tasks but C-tasks, we only consider underutilized

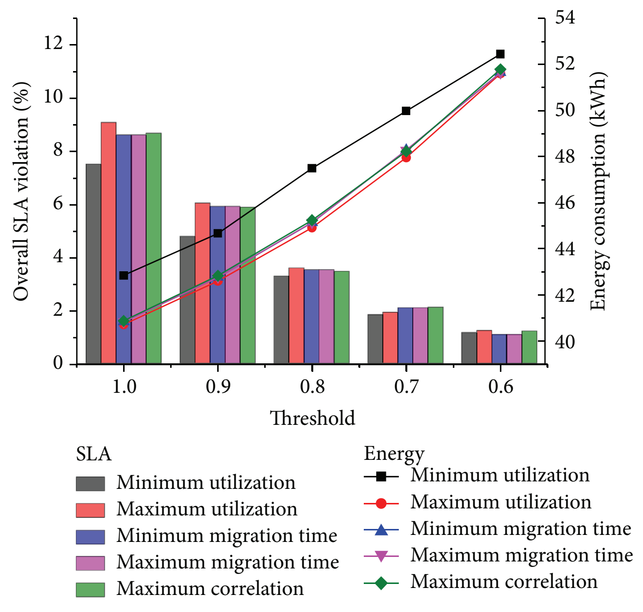


Figure 4: Energy consumption and SLA violations with threshold and migration policies.

VMs for target, disregarding the category of tasks running on the target VM. For task migration, there are many prevalent software and management technologies, such as openMosix, which is a Linux kernel extension that allows processes to migrate to other nodes seamlessly.

On the other hand, choosing a proper threshold value is an important factor that influences the overall performance and there is a tradeoff between the threshold value and SLA violation. Figure 4 shows the tradeoff with various migration policies. Obviously, lowering the threshold value leads to lower energy consumption, but it causes SLA violations, meaning a user's request for tasks cannot guarantee to be succeeded in preagreed metrics. In a condensed situation, where there is no host that can afford additional VMs and the ratio of PM to VM is low, it is more desirable to use a higher threshold value, whereas, in a sparse situation, where there are many free hosts available for additional VMs and the ratio of PM to VM is high, it allows having a lower threshold value but it is energy consuming and wastes resources. As far as the latter case is concerned, we use a double threshold approach to reduce energy consumption more, while incurring the overall SLA violation as little as possible. The resource types for a system are CPU, memory, storage, network, and so forth. Among them, CPU is the most dominant factor that influences energy consumption [12]. In this paper, we focus on CPU utilization for migration policies and leave integrating other types of resource into the migration policies as future work.

3. Task Classification Based Energy-Aware Consolidation Algorithm (TCEA)

As shown in Figure 5, we consider a typical cloud data center with a cloud portal. When a user submits a task to the cloud portal, TCEA first performs a task classification process based on configurations of the task and historical logs. The task is categorized as either computation-intensive or data-intensive. Then, with this task classification information, we assign the task to an appropriate VM and consolidate VMs in the data center in an energy-aware way. After that, TCEA periodically checks hosts with a predefined threshold value so that unnecessary hosts are powered down after migrating their VM to others, while maintaining SLA. The detailed description of our proposed algorithms is given below.

(A) Double Threshold Scheme. Our consolidation algorithms are based on the double threshold scheme. In order to save energy consumption of a cloud data center, one may consider using the minimum number of hosts by utilizing CPU as much as possible for VMs. However, this approach is not an energy efficient solution because it disregards the performance to power ratio. Thus, TCEA uses the upper threshold to prevent heating CPUs up. On the other hand, when many of the hosts are easygoing as a whole, it is necessary to minimize active hosts to save superfluous energy consumption by consolidating VMs. For that purpose, we employ the lower threshold. With the lower threshold, TCEA periodically checks hosts and VMs whether it requires VM or task consolidation. For example, if a host operates with CPU utilization below the lower threshold, we migrate VMs on the host to other hosts as long as there are available hosts to accommodate the VMs without restricting VMs' liberty. With these in mind, it is important to choose proper values for the double threshold scheme, that is, the upper threshold and lower threshold, considering the tradeoff between performance and energy consumption. To determine the conditions of suitable threshold values, we conduct several experiments in Section 4.

(B) Task Classifier. Unlike previous work, we consider a task's characteristics in consolidating a cloud data center. Towards this end, we place a task classifier module to categorize tasks into computational-intensive or data-intensive tasks. When a user submits a task, it examines history log files to check whether it has been performed before. If so, TCEA uses the previous classification information without performing the task classification process. If not, it performs the task classification process as shown in Algorithm 1.

The criteria of classifying tasks in the task classifier function are based on the communication to computation ratio [13]. By examining the execution time and task transfer time of a task, it puts the task to the corresponding queue. In other words, when computation time is greater than task transfer time of a task, the task classifier makes the task resident in $taskQueue_{computation}$. Otherwise, the task is considered as data-intensive. The classification information of the task is also stored in the storage for future use.

(C) Task Assignment. The next step after performing the task classification process is to assign tasks to appropriate VMs. When assigning a task, TCEA first tries to find a host whose utilization is relatively low as shown in Algorithm 2. Then, it checks all the VMs in the host by counting the number

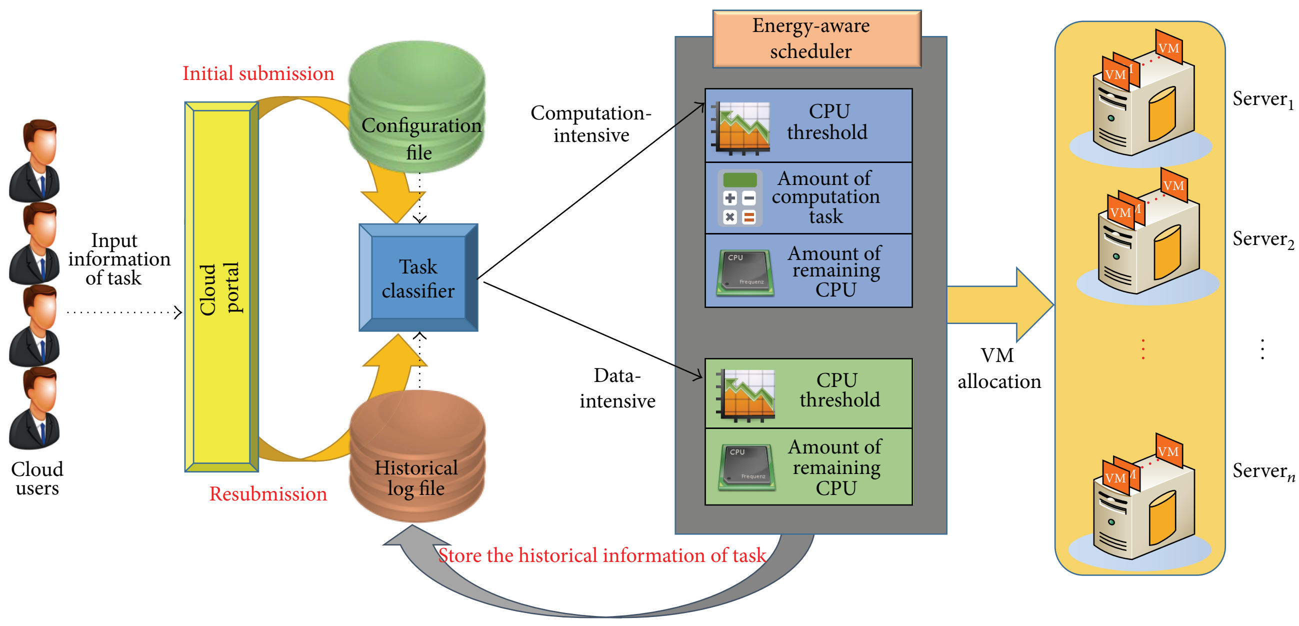


FIGURE 5: System architecture of TCEA.

of computation-intensive tasks. Out of the VMs, a VM that has the least number of computation-intensive tasks can be a candidate when the task is computation-intensive. After iterating this phase, the task assignment function selects a VM for the task.

When the type of a task is data-intensive, TCEA does not care about the types of tasks for finding target VMs. The only consideration is the number of tasks running in VMs. Thus, it finds a VM that runs the minimum number of tasks in order to balance the load. For optimization, the task assignment function migrates a task to another VM. At this stage, we favor computation-intensive tasks for migration because migrating data-intensive tasks is inefficient. In other words, migrating data-intensive tasks takes more time than migrating computation-intensive tasks since it is necessary to move the data of data-intensive tasks as well. When finding an overutilized host, TCEA prefers a VM that runs the largest number of computation-intensive tasks for migration. This is based on the fact that migrating a computation-intensive task is more efficient than migrating a data-intensive task with regard to the number of migrations and utilization shifting. Once a task is chosen for migration, the next step is to choose a target VM. There are two conditions for choosing a target VM. One is CPU utilization and the other is the number of computation-intensive tasks. Among VMs whose host's CPU utilization is low, a VM that runs the least number of computation-intensive tasks will be chosen for the target VM. Then, the task is scheduled to be migrated accordingly.

(D) Consolidation of VMs. For VM consolidation, it is essential to handle and manage VMs and hosts chosen by the double threshold scheme. Algorithm 3 shows the VM consolidation in TCEA in detail. When a host's utilization is above the upper threshold (i.e., overutilized hosts), TCEA chooses a VM to be migrated considering the number of computation-intensive tasks. The more computation tasks a VM has, the more likely the VM is to be a source for migration. Once a source VM is selected, a target host selection phase is performed. Since a source VM will occupy a large portion of utilization, it is preferable to choose a target host whose utilization is relatively low. Therefore, the chosen target host may have fewer numbers of computation-intensive tasks than others. On the other hand, when managing underutilized hosts chosen by the lower threshold, all the VMs in the host will be migrated to hosts whose utilization is normal across the data center. The reason why TCEA chooses normally utilized hosts as migration targets is to exploit the performance to power ratio. Choosing a host of full utilization as a target will result in more energy consumption and consolidation management overheads. For example, when a host becomes overutilized and is chosen as a target host, TCEA will perform redundant load balancing operations.

(E) Task Classification Based Energy-Aware Consolidation Algorithm (TCEA). Algorithm 4 covers our overall consolidation and scheduling scheme. Note that the procedure of lines (1)–(6) is triggered upon receipt of a set of tasks and that of lines (7)–(18) is performed periodically. The task classifier function and the task assignment function are responsible for consolidation and management of tasks in TCEA. TCEA monitors VMs and hosts in the cloud data center for status updates. With the predefined values including the upper and lower thresholds, TCEA maintains $list_{upper}$, $list_{normal}$, and $list_{lower}$ of hosts. To balance performance and energy consumption, VMs in $list_{upper}$ and $list_{lower}$ will be migrated to $list_{normal}$. It is worth noting that choosing the proper values of the upper threshold, lower threshold, and the number of

```
(1)  if $task_i$ has no historical log file
(2)      if VM execution time is greater than data movement time
(3)          $taskQueue_{Computation} \leftarrow taskQueue_{Computation} \cup task_i$;
(4)      else
(5)          $taskQueue_{Data} \leftarrow taskQueue_{Data} \cup task_i$;
(6)      end if
(7)  else // The $task_i$ has historical log file
(8)      Retrieve information from the configuration file;
(9)      Classify data type using obtained information;
(10) end if
```

Algorithm 1: *Task_Classifier ( )*.

```
(1)  if $task_i \in taskQueue_{Computation}$
(2)      for all $host_i \in list_{normal}$, $\forall i \in \{1, 2, \ldots, n\}$;
(3)          Find a $host_i$ with the lowest CPU utilization;
(4)              for all $vm_i \in host_i$, $\forall i \in \{1, 2, \ldots, n\}$;
(5)                  Check the number of computation-intensive tasks;
(6)                  Find a $vm_i$ having the least number of computation-intensive tasks;
(7)              end for
(8)      end for
(9)      Assign $task_i$ to $vm_i$
(10) else if $task_i \in taskQueue_{Data}$
(11)     for all $host_i \in list_{normal}$, $\forall i \in \{1, 2, \ldots, n\}$;
(12)         Find a $host_i$ with the lowest CPU utilization;
(13)             for all $vm_i$ $host_i$, $\forall i \in \{1, 2, \ldots, n\}$;
(14)                 Check the number of tasks;
(15)                 Find a $vm_i$ having the least number of tasks;
(16)             end for
(17)     end for
(18)     Assign $task_i$ to $vm_i$;
(19) end if
```

Algorithm 2: *Assign_Task ( )*.

```
(1)  // for over-utilized hosts $\in list_{upper}$
(2)      Find a $host_i$ with the highest CPU utilization $\in list_{upper}$;
(3)      for all $vm_i \in host_i$, $\forall i \in \{1, 2, \ldots, n\}$;
(4)          Check the number of computation-intensive tasks;
(5)          Find a $vm_i$ having the largest number of computation-intensive tasks;
(6)      end for
(7)      for all $host_j \in list_{normal}$;
(8)          Check the number of computation-intensive tasks;
(9)          Find a $host_j$ having the least number of computation-intensive tasks;
(10)     end for
(11)     Migrate $vm_i$ to $host_j$;
(12) // for under-utilized hosts $\in list_{lower}$
(13)     for all $host_j \in list_{lower}$, $\forall j \in \{1, 2, \ldots, n\}$;
(14)         Find a $host_j$ with the lowest CPU utilization;
(15)     end for
(16)     Migrate all VMs $\in host_j$ to $host_k \in list_{normal}$;
(17)     Switch off $host_j$;
```

Algorithm 3: *Consolidate_VM ( )*.

```
(1)  for all $task_i$, where $task_i \in$ Task, $\forall i \in \{1, 2, \ldots, n\}$;
(2)    Task_Classifier ( )
(3)    Assign_Task ( )
(4)  end for
(5)  Update the status of each task;
(6)  Store monitored status information;
(7)  for all $host_i$, where $host_i \in$ Host, $\forall i \in \{1, 2, \ldots, n\}$;
(8)    Monitor the status of host;
(9)      if CPU utilization is higher than $threshold_{upper}$
(10)       $list_{Upper} \leftarrow list_{Upper} \cup host_i$;
(11)     else if CPU utilization is lower than $threshold_{lower}$
(12)       $list_{lower} \leftarrow list_{lower} \cup host_i$;
(13)     else
(14)       $list_{normal} \leftarrow list_{normal} \cup host_i$;
(15)     end if
(16)   Store monitored status information;
(17) end for
(18) Consolidate_VM ( )
```

ALGORITHM 4: Task classification based energy-aware consolidation algorithm.

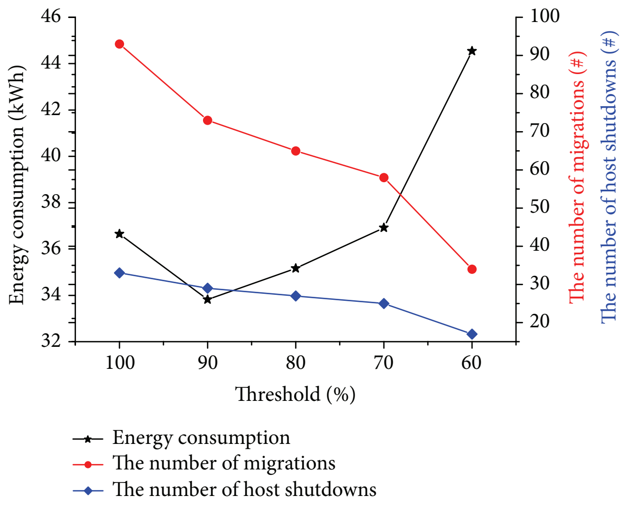


FIGURE 6: Performance results for upper threshold.

VMs to be migrated influences the performance of TCEA. In the next section, we validate TCEA for energy efficiency and performance with these parameters.

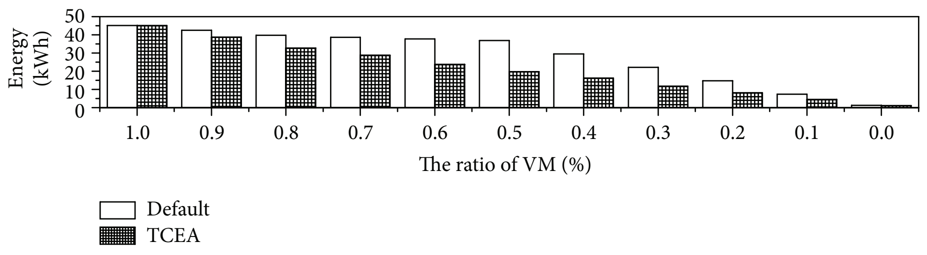
4. Performance Evaluation

In this section, we present experimental results that demonstrate the performance of TCEA for reducing energy consumption by managing VM consolidation while achieving SLA satisfaction. As input, we use real task traces (Intel Netbatch logs [14]) and artifact task logs for a fixed combination of computation-intensive tasks and data-intensive tasks. For experiments, we assume that there are 50 hosts and 100 VMs running in the cloud data center unless specified otherwise. A host is equipped with a quad-core CPU (i7-3770) with 4 GB of RAM and gigabit Ethernet. A user can specify the type of a VM such as the number of vCPU, RAM, and storage capacity. Otherwise, a default VM setting with 1 GB of RAM and 1 vCPU is used.

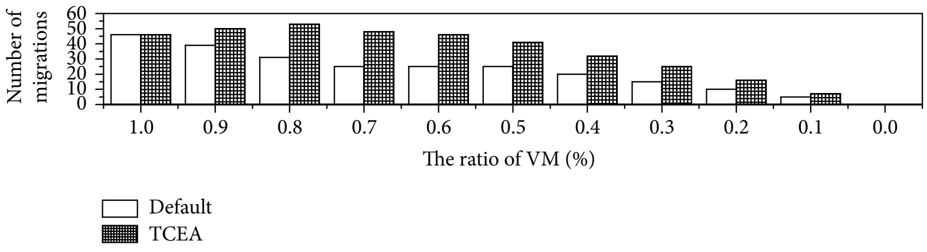
In this experiment, we analyze the runtime of TCEA with varying upper thresholds from 100% to 60%. We conduct this experiment for the real world datasets mentioned above. In Figure 6, x-axis denotes the upper threshold and y-axis represents the energy consumption, the number of VM migrations, and the number of host shutdowns. The number of VM migrations and the number of host shutdowns are constantly going down as the upper threshold decreases. With decreased upper threshold, the available hosts tend to remain alive because VMs should reside in hosts whose utilization is below the upper threshold, and therefore, the number of VM migrations is reduced as well. For energy consumption, 90% is optimal. This indicates that (1) although hosts with 100% of upper threshold maintain more VMs, 100% is not the best threshold due to the performance to power ratio, (2) even though the number of host shutdowns peaks with 100% of upper threshold, the energy reduction of using the lower threshold (90%) dominates that of the number of host shutdowns, and (3) the number of VM migrations decreases with lower upper threshold because the probability of finding satisfactory target VMs gets lower too. For the rest of experiments, we use 90% of upper threshold unless specified otherwise.

For a sparse situation, where there are many free hosts available for additional VMs and the ratio of PM to VM is high, we devise an optimization algorithm to migrate VMs from underutilized hosts to others and shutdown the hosts, thereby reducing energy consumption. To this end, we use a lower threshold such that VMs in a host below the lower threshold are scheduled to be migrated to other hosts, and then the host gets shutdown. Figure 7 shows energy consumption, the number of VM migrations, and the number of host shutdowns with varying lower thresholds (e.g., 0.8 of x-axis means that 20% of hosts are chosen by the lower threshold). Comparing with default (no task classification is performed), TCEA consumes 14.05% less energy on average. When the lower threshold is 50%, the difference between default and TCEA reaches a peak. With respect to energy consumption, the number of VM migrations, and the number of host shutdowns, we use 50% of lower threshold for the rest of experiments unless specified otherwise.

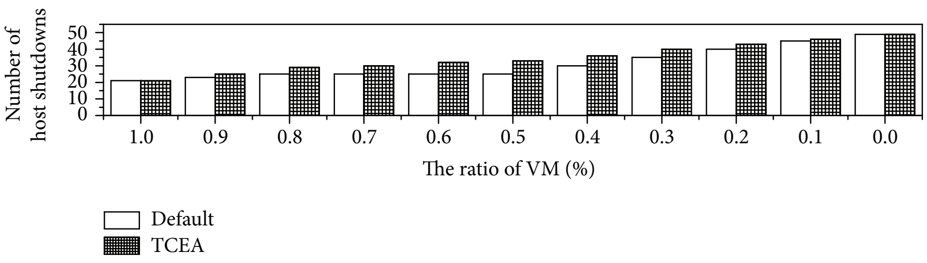
To verify the effectiveness of lower thresholds, we conduct another experiment showing energy consumption, the number of VM migrations, and the number of host shutdowns with VM ratios by increasing the number of VMs and hosts (1x means a default setting of 100 VMs and 50 hosts). Note that, in this experiment, 0.9 of VM ratio means that 10% of hosts whose utilization is below the lower threshold are scheduled to be powered down by migrating their VMs. As shown in Figure 8, around 50% of the VM ratio suits our purpose in terms of energy consumption, the number of VM migrations, the number of host shutdowns, and SLA violations. The ratio below 0.5 leads to SLA violations; therefore we do not use ratio lower than 0.5.



(a) Energy consumption



(b) The number of migrations



(c) The number of host shutdowns

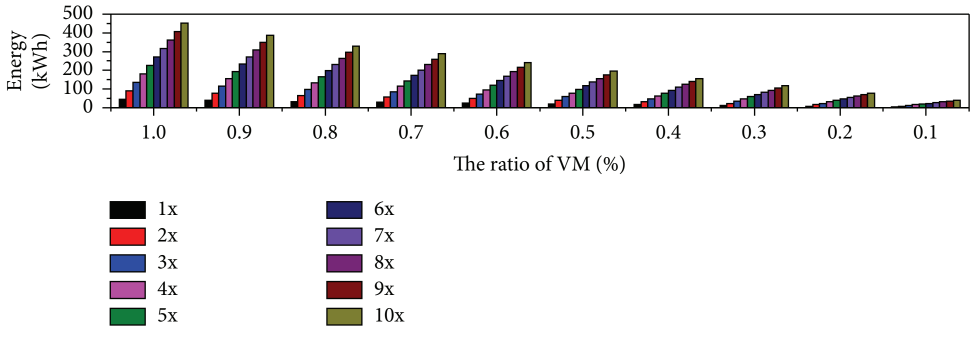
Figure 7: Performance results for lower threshold.

To investigate the respective improvement brought by TCEA's double threshold scheme, we compare the performance of TCEA (double threshold) with the single threshold scheme and default (no threshold and no task classification) setting. In this experiment, we use real task trace logs and artifact task logs for a fixed combination of computation tasks and data-intensive tasks. In Figure 9, "Job" indicates real task traces, *Job_c* indicates only computation-intensive tasks, *Job_d* indicates only data-intensive tasks, and *Job_cd* indicates 50% of computation-intensive tasks and 50% of data-intensive tasks.

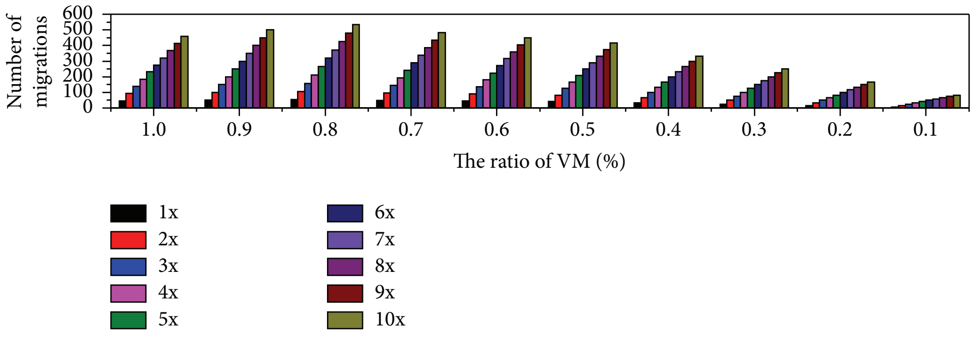
As shown in Figure 9, there is no difference for the results with the default setting (no threshold) in terms of energy consumption because a threshold scheme is not applicable. Nevertheless, we leave them for comparison. The double threshold scheme saves 47.6% of energy compared to the default setting. For the single threshold scheme, there is no big difference between 90% and 100% but there are more VM migration operations with 100% of upper threshold, which leads to overheads. Of job categories (*Job*, *Job_c*, *Job_d*, and *Job_cd*), *Job_d* shows a little performance impact with single threshold because it uses relatively less CPU utilization, and *Job_cd* has performance improvement when the single threshold is above 80%. The result for double threshold shows similar phenomenon when the single threshold is used. However, the double threshold scheme further reduces energy consumption by 14.2% compared to the single threshold scheme.

An important requirement for achieving the optimal performance of virtualized cloud environments is to find the appropriate number of VMs per PM. In such an environment, the ratio of PM to VM affects the overall performance. To validate the effect of the ratio of PM to VM, we compare the threshold schemes (default, single, and double). The double scheme achieves the largest energy reduction, followed by the single scheme and by the default scheme as shown in Figure 10. The double threshold scheme saves energy consumption by 11.3% and 27.2% comparing with single and default, respectively. For the number of VM migrations, there are some points where the double threshold scheme exhibits more VM migrations than the single threshold scheme does, but it stabilizes when the ratio of PM to VM is 1 : 9 or more. In addition, the double threshold scheme always outperforms with respect to the number of host shutdowns.

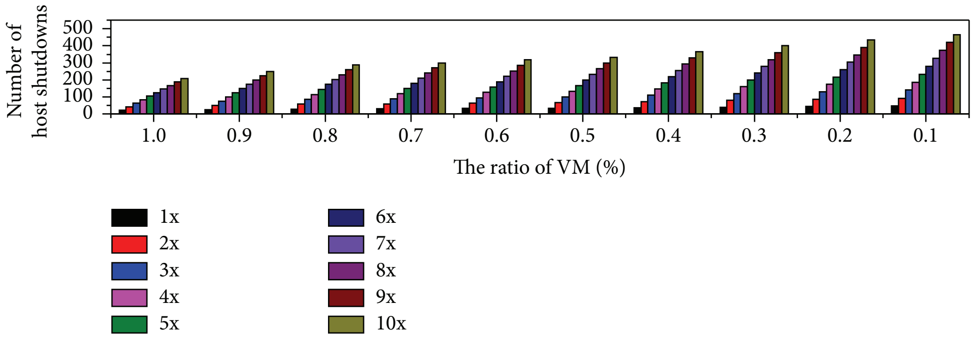
To measure the scalability for the number of PMs and VMs, we increase the number of PMs and VMs from 1 : 2 up to 10 : 20 as shown in Figure 11. As expected, TCEA consumes less energy by 17.9% on average than the default scheme and outnumbers the default scheme in terms of the number of shutdowns. For VM consolidation, TCEA has a higher number of VM migrations. For task scalability, we compare energy consumption by increasing the task log size



(a) Energy consumption



(b) The number of migrations



(c) The number of host shutdowns

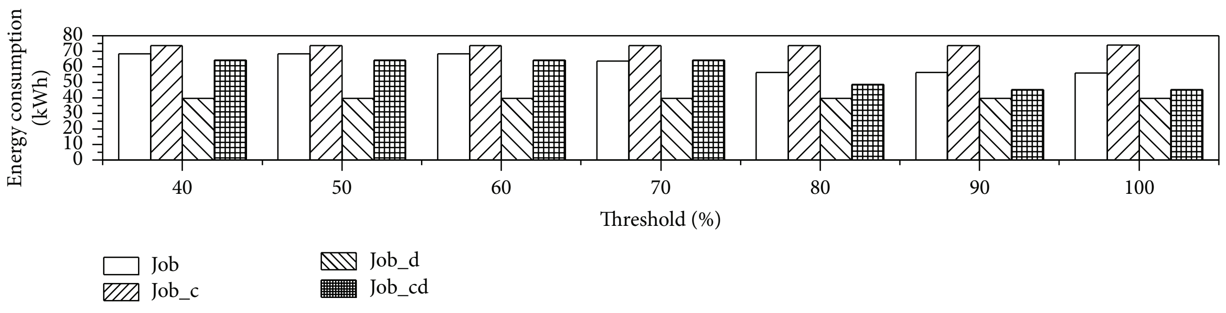
FIGURE 8: Performance scalability for the number of nodes with lower threshold.

up to 10 times as depicted in Figure 12. Comparing with the default scheme, TCEA consumes less energy by 15.8% on average. Obviously, TCEA has more VM migration and host shutdown operations than the default scheme has for VM consolidation.

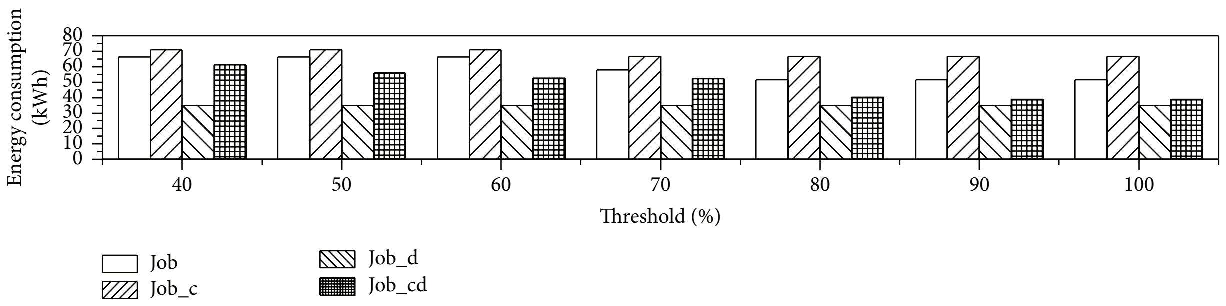
5. Related Work

We summarize the related work across three perspectives: resource allocation and scheduling in data centers and clouds, threshold-based schemes with different objectives, and energy savings in data centers. To balance energy consumption and VM utilization, the authors of [10] used a performance to power ratio. It schedules VM migration dynamically and consolidates servers in clouds. They compared their proposed algorithm with three different algorithms including the DVFS algorithm using real trace log files. The authors of [13] proposed a criterion to divide computation-intensive tasks and data-intensive tasks using a communication to computation ratio. The rationale of this task classification is to employ resource allocation methods based on tasks or workflows to improve performance.

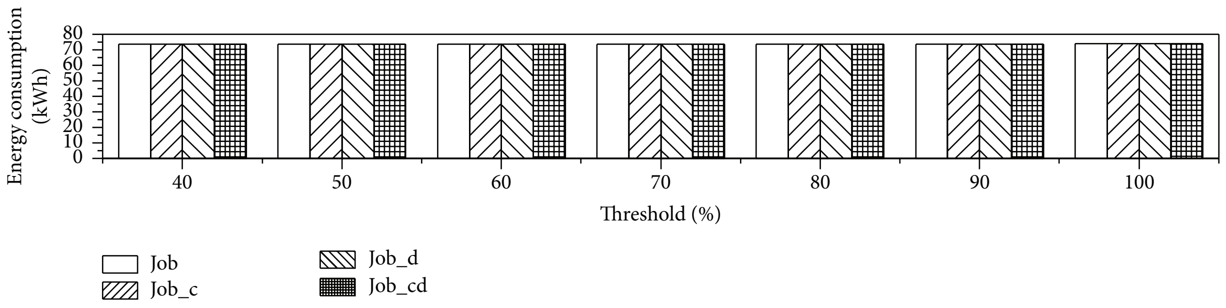
In [15], they developed an energy-aware scheduling to reduce total processing time for VMs in a precedence-constrained condition, while maximizing PM's utilization considering communication costs. In [16], they proposed a prediction algorithm for finding overutilized servers and a best-fit algorithm for hosts and VMs. The results show that the algorithms reduce the number of migration operations, rebooting servers, and energy consumption, while achieving SLA guarantee. A separation mechanism of I/O tasks to perform computation-intensive tasks in a batch in virtualized servers to mitigate virtualization overheads is proposed in



(a) Upper



(b) Upper + lower



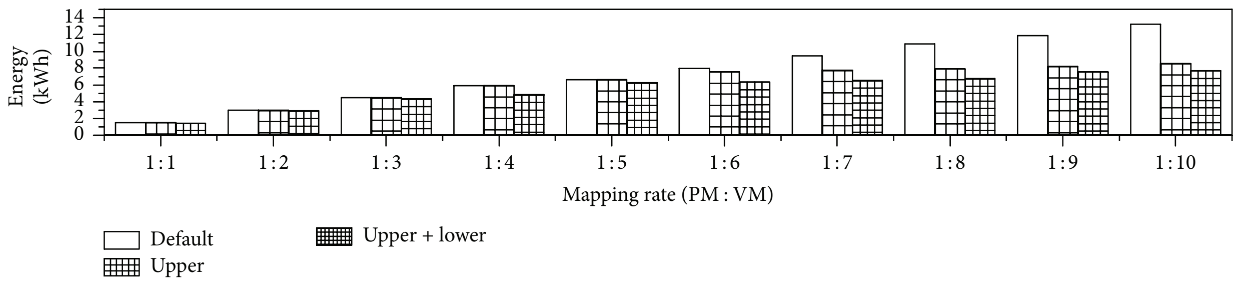
(c) No threshold

Figure 9: Performance comparison with task types and threshold schemes.

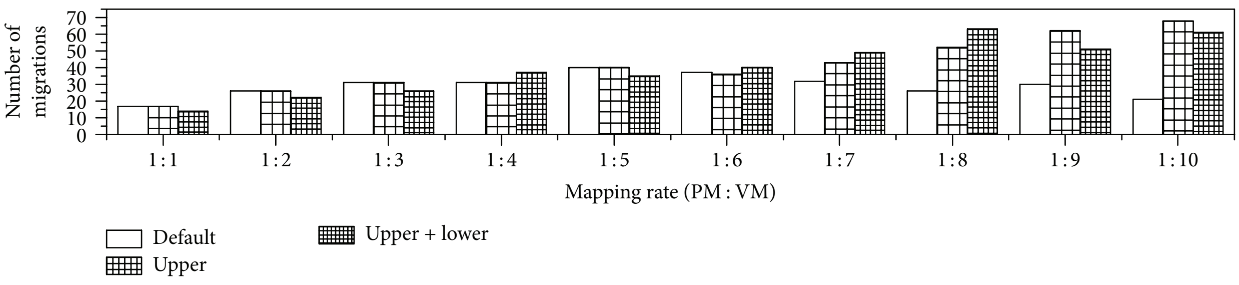
[17]. Because energy consumption and the frequency of SLA violations determine the quality of service [18], in this paper, we balance the tradeoff between energy consumption and SLA violations using the double threshold schemes based on tack classification and none of the abovementioned studies consider the energy saving objectives in the context of task classification.

For data-intensive workflows, where the majority of energy consumption accounts for storing and retrieving data, the authors of [19] consider not using DVFS. Instead, they installed and used an independent node to store data-intensive tasks. They endeavor to reduce energy consumption by minimizing data access and then performed evaluations by increasing the communication to computation ratio. The authors of [20] proposed a VM scheduling algorithm to reduce energy consumption with DVFS. By dynamically adjusting clock frequency and its corresponding voltage, it results in energy reduction in idle and computation stages. In [21], they proposed a scheduling algorithm based on priority and weight with DVFS. It increases servers' resource utilization to reduce energy consumption of the servers. In [22], they used a threshold value to migrate a VM to another host. When a host's utilization is below the threshold value, all the VMs belonging to the host are scheduled to be migrated to other hosts to save idle power consumption. In addition, some VMs are scheduled to be migrated when the host's utilization exceeds a certain threshold value to avoid SLA violations. A service framework that allows monitoring energy consumption and provisioning of VMs to appropriate location in an energy-efficient way is designed in [23].

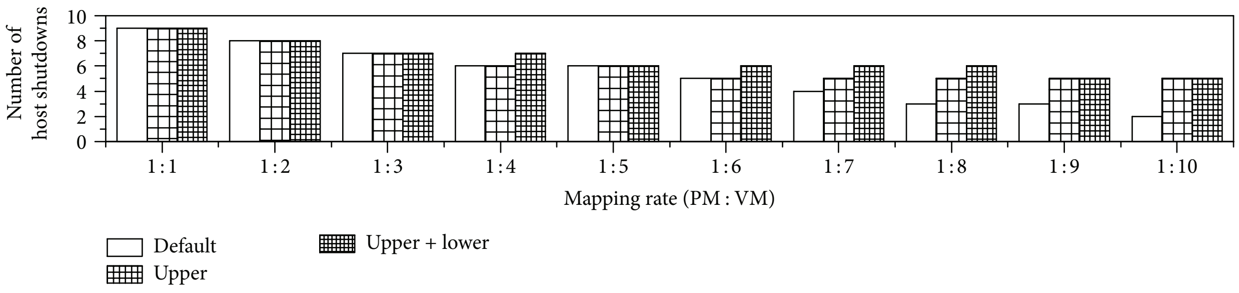
Various CPU consolidation techniques including DVFS, dynamic power shutdown (DPS), and core-level power gating (CPG) are introduced in [24]. The authors of [25] used a threshold value to migrate VMs and consider resource, temperature, and network conditions for optimization. They considered migration time to minimize the number of VMs that are in progress of migration simultaneously. The authors of [26] designed an energy-aware resource allocation heuristic for VMs' initial placement, VM selection policy for migration, and migration policy in virtualized cloud



(a) Energy consumption



(b) The number of migrations



(c) The number of host shutdowns

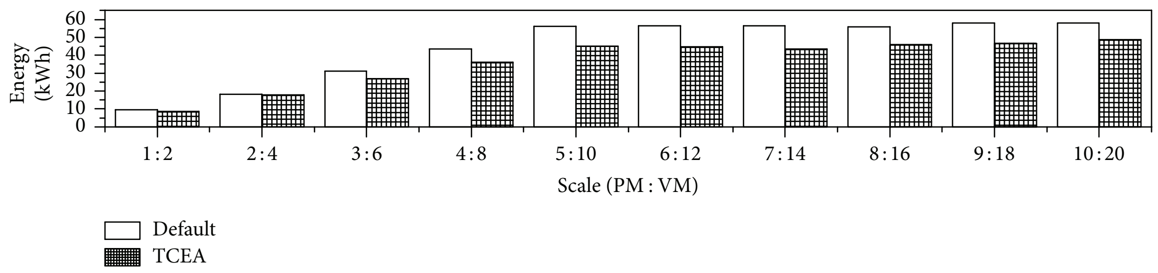
Figure 10: Performance comparison with PM to VM ratio.

computing environments. The authors of [27] developed a resource allocation method at the cloud application level. In the application's perspective, it allocates virtual resources for the application with a threshold-based dynamic resource allocation algorithm to improve resource utilization. In [28], they developed a VM placement algorithm based on the evolutionary game theory. According to their experiments, when the loads of the data center are above 50%, the optimizations are unnecessary.

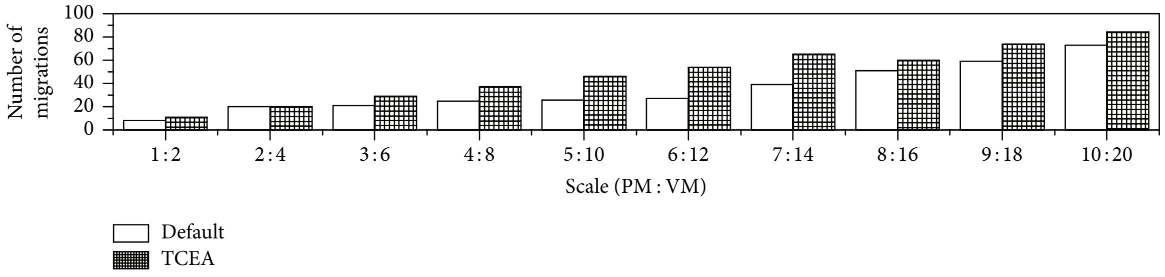
However, the design objective and the implementation methods of these cloud data center schedulers and consolidation algorithms are different from TCEA in terms of the following aspects. First, the target of these cloud data center schedulers is to enforce resource allocation strategy based on fairness or priorities when sharing the resources of large-scale cloud data centers among VMs, while TCEA is aimed at improving both energy consumption and the performance of tasks by dynamically migrating VMs in runtime. Second, we extend a single threshold scheme to further improve the overall performance and energy consumption by incorporating the double threshold scheme and task classification together. Finally, they cannot solve both the maximum utilization problem and the host shutdown problem in an efficient way, while TCEA takes the performance to power ratio into consideration and employs the host shutdown mechanism by migrating VMs on underutilized hosts while maintaining SLA violations.

6. Conclusions

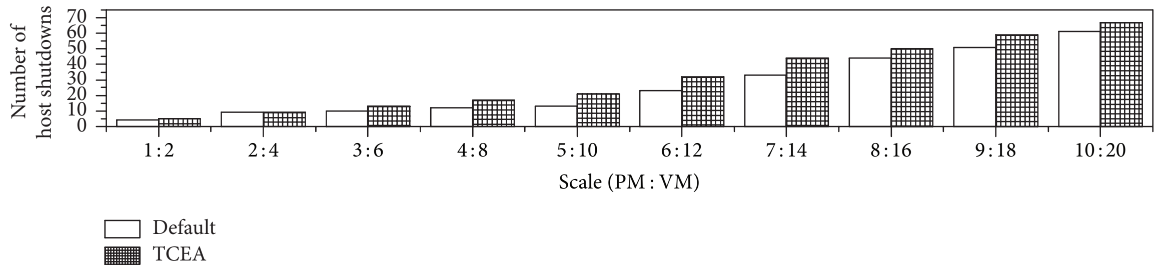
As green IT and its related technologies have received much attention recently, reducing the power consumption of cloud data centers is one of the critical issues to address, thereby reducing the carbon dioxide footprints. In this paper, we propose two consolidation mechanisms for a cloud data center. One is the task consolidation based on task classification (computation-intensive or data-intensive) and the other is the VM consolidation that uses a double threshold scheme (upper and lower). We optimize energy consumption in a virtualized data center not by maximizing resource utilization but by balancing resource utilization of hosts with migrating appropriate VMs. We prove that our task classification based energy-aware consolidation algorithm (TCEA) achieves significant energy reduction without incurring predefined SLA violations.



(a) Energy consumption



(b) The number of migrations



(c) The number of host shutdowns

Figure 11: Performance comparison with scalability.

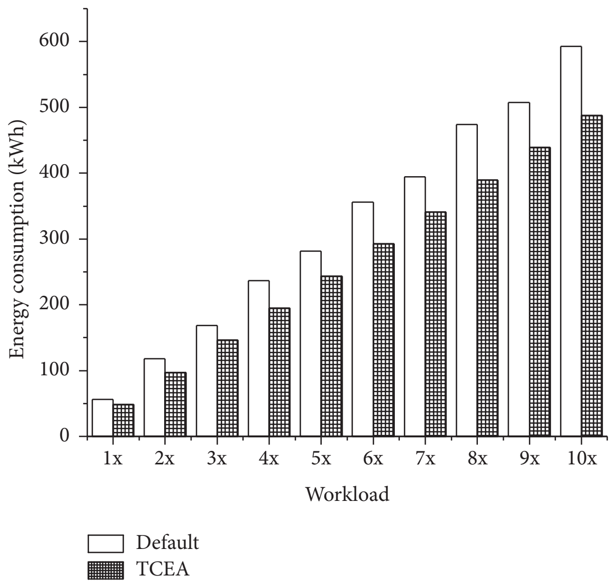


Figure 12: Scalability for the number of tasks.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2015R1C1A2A01054813).

References

[1] W. Ai, K. Li, S. Lan et al., "On elasticity measurement in cloud computing," *Scientific Programming*, vol. 2016, Article ID 7519507, 13 pages, 2016.

[2] J. Lim, T. Suh, J. Gil, and H. Yu, "Scalable and leaderless Byzantine consensus in cloud computing environments," *Information Systems Frontiers*, vol. 16, no. 1, pp. 19–34, 2014.

[3] S. K. Choi, K. S. Chung, and H. Yu, "Fault tolerance and QoS scheduling using CAN in mobile social cloud computing," *Cluster Computing*, vol. 17, no. 3, pp. 911–926, 2014.

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

[5] Y. Wen, X. Zhu, J. J. P. C. Rodrigues, and C. W. Chen, "Cloud mobile media: reflections and outlook," *IEEE Transactions on Multimedia*, vol. 16, no. 4, pp. 885–902, 2014.

[6] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: a survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732–794, 2015.

[7] N. Boumkheld, M. Ghogho, and M. E. Koutbi, "Energy consumption scheduling in a smart grid including uding renewable energy," *Journal of Information Processing Systems*, vol. 11, no. 1, pp. 116–124, 2015.

[8] P. Barham, B. Dragovic, K. Fraser et al., "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[9] I. Habib, "Virtualization with KVM," *Linux Journal*, vol. 2008, no. 166, article 8, 2008.

[10] X. Ruan and H. Chen, "Performance-to-power ratio aware Virtual Machine (VM) allocation in energy-efficient clouds," in *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER '15)*, pp. 264–273, Chicago, Ill, USA, September 2015.

[11] D. Sood, H. Kour, and S. Kumar, "Survey of computing technologies: distributed, utility, cluster, grid and cloud computing," *Journal of Network Communications and Emerging Technologies*, vol. 6, no. 5, pp. 99–102, 2016.

[12] Y. Gao, H. Guan, Z. Qi, B. Wang, and L. Liu, "Quality of service aware power management for virtualized data centers," *Journal of Systems Architecture*, vol. 59, no. 4-5, pp. 245–259, 2013.

[13] W. Guo, W. Sun, W. Hu, and Y. Jin, "Resource allocation strategies for data-intensive workflow-based applications in optical grids," in *Proceedings of the 10th IEEE Singapore International Conference on Communications Systems (ICCS '06)*, pp. 1–5, IEEE, Singapore, November 2006.

[14] O. Shai, E. Shmueli, and D. G. Feitelson, "Heuristics for resource matching in Intel's compute farm," in *Job Scheduling Strategies for Parallel Processing*, vol. 8429, pp. 116–135, Springer, 2013.

[15] V. Ebrahimirad, M. Goudarzi, and A. Rajabi, "Energy-aware scheduling for precedence-constrained parallel virtual machines in virtualized data centers," *Journal of Grid Computing*, vol. 13, no. 2, pp. 233–253, 2015.

[16] J. Huang, K. Wu, and M. Moh, "Dynamic Virtual Machine migration algorithms using enhanced energy consumption model for green cloud data centers," in *Proceedings of the International Conference on High Performance Computing & Simulation (HPCS '14)*, pp. 902–910, Bologna, Italy, July 2014.

[17] P. Xiao, Z. Hu, D. Liu, X. Zhang, and X. Qu, "Energy-efficiency enhanced virtual machine scheduling policy for mixed workloads in cloud environments," *Computers & Electrical Engineering*, vol. 40, no. 5, pp. 1650–1665, 2014.

[18] A. Paya and D. C. Marinescu, "Energy-aware load balancing policies for the cloud ecosystem," in *Proceedings of the 28th IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW '14)*, pp. 823–832, IEEE, Phoenix, Ariz, USA, May 2014.

[19] P. Xiao, Z.-G. Hu, and Y.-P. Zhang, "An energy-aware heuristic scheduling for data-intensive workflows in virtualized datacenters," *Journal of Computer Science and Technology*, vol. 28, no. 6, pp. 948–961, 2013.

[20] G. von Laszewski, L. Wang, A. J. Younge, and X. He, "Power-aware scheduling of virtual machines in DVFS-enabled clusters," in *Proceedings of the 2009 IEEE International Conference on Cluster Computing and Workshops (CLUSTER '09)*, pp. 1–10, IEEE, New Orleans, La, USA, September 2009.

[21] C.-M. Wu, R.-S. Chang, and H.-Y. Chan, "A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters," *Future Generation Computer Systems*, vol. 37, pp. 141–147, 2014.

[22] L. Luo, W. Wu, W. Tsai, D. Di, and F. Zhang, "Simulation of power consumption of cloud data centers," *Simulation Modelling Practice and Theory*, vol. 39, pp. 152–171, 2013.

[23] G. Katsaros, J. Subirats, J. O. Fitó, J. Guitart, P. Gilet, and D. Espling, "A service framework for energy-aware monitoring and VM management in Clouds," *Future Generation Computer Systems*, vol. 29, no. 8, pp. 2077–2091, 2013.

[24] I. Hwang, T. Kam, and M. Pedram, "A study of the effectiveness of CPU consolidation in a virtualized multi-core server system," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED '12)*, pp. 339–344, Redondo Beach, Calif, USA, August 2012.

[25] K. Maurya and R. Sinha, "Energy conscious dynamic provisioning of virtual machines using adaptive migration thresholds in cloud data center," *International Journal of Computer Science and Mobil Computing*, vol. 2, no. 3, pp. 74–82, 2013.

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

[27] W. Lin, J. Z. Wang, C. Liang, and D. Qi, "A threshold-based dynamic resource allocation scheme for cloud computing," *Procedia Engineering*, vol. 23, pp. 695–703, 2011.

[28] Z. Xiao, J. Jiang, Y. Zhu, Z. Ming, S. Zhong, and S. Cai, "A solution of dynamic VMs placement problem for energy consumption optimization based on evolutionary game theory," *Journal of Systems and Software*, vol. 101, pp. 260–272, 2015.