


Please cite the Published Version

Ding, Weichao, Luo, Fei, Han, Liangxiu , Gu, Chunhua, Lu, Haifeng and Fuentes, Joel (2020) Adaptive virtual machine consolidation framework based on performance-to-power ratio in cloud data centers. Future Generation Computer Systems, 111. pp. 254-270. ISSN 0167-739X

DOI: <https://doi.org/10.1016/j.future.2020.05.004>

Publisher: Elsevier BV

Version: Accepted Version

Downloaded from: <https://e-space.mmu.ac.uk/625701/>

Usage rights:  In Copyright

Additional Information: This is an Author Accepted Manuscript of a paper accepted for publication in Future Generation Computer Systems, published by and copyright Elsevier.

Enquiries:

If you have questions about this document, contact openresearch@mmu.ac.uk. Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)

Adaptive Virtual Machine Consolidation Framework Based on Performance-to-Power Ratio in Cloud Data Centers

WEICHAO DING¹, FEI LUO¹, LIANGXIU HAN², CHUNHUA GU¹, HAIFENG LU¹ and JOEL FUENTES³

¹School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China

²School of Computing, Mathematics and Digital Technologies, Manchester Metropolitan University, UK

³Department of Computer Science and Information Technology, Universidad del Bío-Bío, Chillán, Chile

Corresponding author: Fei Luo (e-mail: luof@ecust.edu.cn).

This work was supported by the National Natural Science Foundation of China (Grant NO.61472139), the Newton Fund Institutional Links of UK (Grant ID 332438911).

ABSTRACT Efficient resource management in a Cloud data center relies on minimizing energy consumption and utilizing physical resource efficiently while maintaining the *service-level agreement* (SLA) at its highest level. To achieve this goal, dynamically consolidating *virtual machines* (VMs) is considered a promising method, because it eliminates the hotspots resulting from overloaded hosts and switches the underloaded hosts to sleep mode through the live migration of VMs. However, during the consolidation, each VM migration consumes additional resource, leading to performance degradation and SLA violation. To address this issue, this study proposes a novel adaptive *performance-to-power-ratio* (PPR)-aware dynamic VM consolidation framework based on both the predicted resource utilization and PPR of the heterogeneous hosts to resolve the trade-off of performance and energy. The proposed framework consists of four stages: (1) host overload detection based on *residual available computing capacity*; (2) selection of the appropriate VMs for migration from the overloaded hosts based on *minimum data transfer*; (3) host underload detection based on multi-criteria Z-score approach; (4) allocating the VMs selected for migration from the overloaded and underloaded hosts based on the modified *power-aware best-fit decreasing* algorithm. To validate the reliability and scalability of the proposed method, we performed experimental evaluation in both real and simulated environments. The experimental results demonstrate that the proposed approach can reduce the energy consumption effectively and ensure maximal conformity to the *quality of service* (QoS) requirements across heterogeneous infrastructures, in comparison with the existing competitive approaches.

INDEX TERMS Cloud computing, Dynamic consolidation, Energy consumption, SLA violation

I. INTRODUCTION

The rapid growth of cloud computing applications has led to a dramatic increase in the operation costs of cloud data centers, because of the increasing energy consumption and environmental pollutants [1], [2]. The energy consumption of data centers increased by 56% worldwide from 2005–2010, accounting for 1.3% of total electricity use [3]. A recent study [4] shows that electricity demand by data centers in 2018 was an estimated 198 billion kWh, almost 1% of demand for electricity in the world. In the US, it is reported that data center power demand has increased from 29 billion kWh in 2000 to nearly 73 billion kWh in 2020 [5]. This is not only because of the large number of hosts in the data centers, but also because of the low efficiency of power

utilization ratio and inefficient use of resource. For example, it was reported that the average CPU utilization of over 5000 hosts is only about 10-50% of their maximum utilization levels in a data center during a 6-month period [6]. In addition, even completely idle hosts consume approximately 70% of their peak power [7]. Therefore, there is an urgent need for new solutions on how to manage under- or overloaded hosts and satisfy the *service-level agreement* (SLA) requirements for improving energy efficiency and reducing operational costs in data centers.

One effective way to maintain the *quality of service* (QoS) while improving energy efficiency is dynamic *virtual-machine* (VM) consolidation using the live migration technique [8], [9], whereby VMs are migrated from one host

to another, for instance, from an underloaded host to another while ensuring minimum downtime and no suspension or performance degradation; in this framework, idle hosts are switched to the sleep or a low-power mode, based on their actual resource requirements. However, it has been reported that the aggressive migration process can cause performance degradation and SLA violation [39], due to the dynamic resource usage patterns in service applications.

In our previous work [40], we proposed an energy-aware VM consolidation method, named BS-MISR, to achieve the balance between energy consumption and performance. However, the BS-MISR had several limitations: (1) it employed the current value of resource utilization, which may result in inaccurate prediction of application load and cause unnecessary VM migration; (2) BS-MISR mainly considered factors such as CPU utilization and memory, but not the performance-to-power ratio (PPR) factor in relation to the various heterogeneous hosts; (3) BS-MISR only focused on the two stages of VM selection and VM placement in the process of resource consolidation.

As a matter of fact, a high level of PPR can help reduce energy consumption without degrading the performance of the cloud data center. Figure 1 illustrates the impact of low and high PPR under two scenarios: (1) consolidation of VMs on hosts with low PPR; and (2) consolidation of VMs on hosts with high PPR. In Figure 1, there are four hosts; two are HP ProLiant DL360 G7 (Server1 and Server2), and the other two are HP ProLiant DL360 G9 (Server3 and Server4). Each host has a VM. The average PPR of the HP ProLiant DL360 G7 and HP ProLiant DL360 G9 are 3329 and 10118, respectively. Assuming that each VM requires the same amount of resource, the workload of HP ProLiant DL360 G7 and host HP ProLiant DL360 G9 are 15% and 5%, respectively. When the CPU utilization alone is taken into consideration in the process of VM consolidation, Server3 and Server4 will be detected as under-loaded, and to minimize the number of active hosts, all the VMs will be consolidated on Host Server1 or Server2, making the energy consumption of all the hosts equal to 142 W. In contrast, if considering both the PPR and the CPU utilization of the different hosts, the system will consolidate all the VMs in Server3 or Server4, and the total energy consumption would be 101 W (energy consumption and PPR values of the hosts are provided by SPECpower benchmark [10]). Although the number of active hosts in both approaches is the same, the energy consumption in the second approach is 28.9% lower than that of the first approach. Thus, it is evident that the PPR is an influential factor in the context of energy-aware resource management.

In this study, by taking both factors, namely predicted resource utilization and PPR of heterogeneous hosts, into account, we propose a novel adaptive framework for VM consolidation that resolves the trade-off between the energy consumption and application performance. The main contributions of this paper are as follows:

- A novel workload prediction model built on the *moving average* (MA) and the *interquartile range* (IQR)

techniques to predict the resource usage in decision-making. The proposed model improves the prediction accuracy by considering the degree of dispersion of resource utilization, and it reduces the frequency of VM migrations caused by abruptly changing the workload.

- A novel host overload detection algorithm based on the *residual available computing capacity* (RACC) evaluation model that can set adaptive CPU upper thresholds for heterogeneous hosts, according to the PPR of the host.
- A multi-criteria host underload detection algorithm using Z-score [26] technique by considering both the hosts' CPU utilization and power efficiency.
- A VM selection approach, called *Minimum Data Transfer* (MDT), based on dynamic programming, which can deal with the resource shortage effectively, and minimize the data transfer during VM migrations.

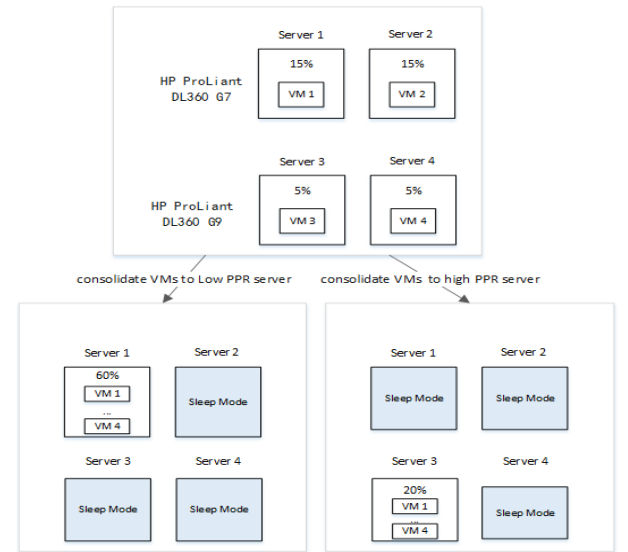


FIGURE 1 VM Consolidation on Different PPR Hosts

The remainder of this paper is organized as follows. In Section II, the related works are discussed. The proposed VM consolidation framework is introduced in Section III. The key stages of the VM consolidation framework design are proposed in Section IV. Evaluation metrics including energy consumption, *SLA violations* (SLAV), *Energy-SLA violations* (ESV), and the total amount of transmitted data are discussed in detail in Section V. The simulation setup and performance evaluation are described in Section VI. Finally, Section VII concludes the paper and highlights the future work.

II. RELATED WORK

There is a wide range of studies that address the *VM consolidation problem* (VMCP) for balancing energy and performance in large cloud data centers. In this section, previous literature regarding the energy-aware management is discussed.

Some studies that use heuristics to solve the VMCP are [9], [11], [14], [17], [19]. For example, Beloglazov et al. in [9] proposed several metrics to rank hosts using an adaptive

upper bound method based on the statistical analysis of historical CPU data. They divided the process of VM consolidation into four steps: (1) host overload detection, (2) host underload detection, (3) selection of VMs from overloaded hosts and (4) finding new placement for the VMs. The authors also proposed several adaptive heuristics for all the steps: *median absolute deviation* (MAD), IQR, and *local regression* (LR) for Step 1; *simple method* (SM), considering only the CPU utilization for Step 2; *minimum migration time* (MMT), random choice and *maximum correlation* (MC) for Step 3; *power-aware best-fit decreasing* (PABFD) [12] for Step 4. Their experimental results showed that the best combination of all the above methods is (LR/MMT/SM/PABFD); however, the underload detection algorithm only accounts for the CPU utilization of the physical host without considering the characteristics (such as quantity, memory utilization, etc.) of the VMs, which can lead to an increase in the number of VMs migrated and amount of data transmission in the consolidation process.

Building upon the work in [9], Horri A et al. [14] proposed two novel heuristics for underload detection and VM placement. In the detection phase of the underloaded host, the authors extended the SM by considering the number of VMs on the host, thereby reducing the SLA violations. In the VM placement phase, they introduced an efficient QoS-aware algorithm based on both the host utilization and minimum correlation between the VMs. Experimental results showed that the above algorithms significantly outperformed LR/MMT/SM/PABFD [9]. Similarly, an online resource management policy that reduces energy consumption and minimizes the number of migrations in cloud environments was proposed in [17]. The authors introduced a new prediction approach to selecting the VMs by exploiting the MA technique for host overload detection and multi-criteria decision making algorithms. They also proposed an effective technique for calculating the weights of the CPU, RAM, and bandwidth. However, their approach carries the risk of resource losses because, in the overload detection phase, they assumed that the weights of the resource in different dimensions were the same, without considering the heterogeneity of physical resource.

A novel VM consolidation approach to obtaining energy, QoS, and temperature balance in cloud environments was proposed in [19]. Unlike other research, they categorized the hosts based on operations per second per watt; then, they calculated the temperature of the optimum host that had a workload of 90% in the data center and designated this temperature as the upper threshold for other hosts. However, the host's temperature is related not only to the workload but also to the lifetime of the hardware. Thus, as the hardware, especially the fan, ages, the temperature of the CPU will rise sharply. Therefore, it is not ideal to use temperature as the threshold when performing thermal-aware consolidation practically.

Several studies have formulated the VMCP as a variable-size bin-packing problem [15], [18], [27], where hosts are conceived as bins and VMs are conceived as items. For

instance, [15] considered the VMCP as a three-dimension bin-packing problem and solved it using four variants of the best-fit decreasing algorithm. In contrast to existing studies, the authors applied exact algorithms to VMCP. Although the devised models and exact algorithms can solve large-scale instances, they cannot capture future mega-scale data centers. H. Inkwon et al. [18] considered the VMCP as a multi-capacity stochastic bin-packing problem and efficiently solved it using a heuristic method. The authors treated the resource demands of VMs as random variables with known means and standard deviations and circumvented performance bottlenecks by calculating their correlation. However, the authors created a correlation matrix using a random function and assumed that there was no pattern in the correlation matrix; thus, their approach is unsuitable for specific applications that have target service models.

Some other studies used bio-inspired and nature-inspired algorithms, such as PSO [16], [27], GA [20], and ACO [30], [31]. The authors in [20] proposed a hybrid approach of GA and PSO called HGAPSO for resource allocation and VMs migration, which can not only save the energy consumption and minimize the wastage of resource but also avoid SLA violation at the cloud data center. Zhihua Li et al. [21] developed a discrete differential evolution algorithm to search for the global optimum solution for VM placement. N. Sharma et al. [27] focused on the key goals of multi-objective VM allocation based on PSO and VM migration to reduce energy consumption, resource wastage, and SLA violations. However, these intelligent evolutionary algorithms are sensitive to the parameters and require artificial adjustment according to the system status. Therefore, the cost of parameter optimization is high, and orchestrating better trade-off between multiple objectives is time-consuming, which is unsuitable for the large-scale *infrastructure-as-a-service* (IaaS) cloud platform.

A few schemes applying predictive models, and various host overload approaches have been proposed [28], [29], [42]. F. Fahimeh et al. [28] proposed a regression-based model to approximate the future CPU and memory utilization of VMs and PMs for energy-aware VM consolidation. The authors in [29] developed a deep learning-based adaptive window size selection method, dynamically limiting the sliding window size to capture the trend for the latest resource utilization, and then build an estimation model for each trend period. However, it is necessary to improve the prediction accuracy of those aforementioned methods through the simulation-based learning that cannot be implemented by IaaS cloud providers, such as Amazon EC2.

It is necessary to reiterate at this juncture that the major weakness of all the aforementioned approaches is their failure to take the variation in the PPRs of the heterogeneous infrastructures in resource management into consideration. To address this limitation, we propose a novel adaptive framework for dynamic VM consolidation based on both the predicted utilization of resource and the PPR of the hosts. The proposed framework exploits the prediction model to determine the overloaded hosts based on the RACC

evaluation model, performs the VM selection based on the minimum data-transfer algorithm, detects the underloaded hosts using the Z-score technique. Finally, during the live migration, the proposed VM placement algorithm gathers VMs from candidate hosts to be mapped in a certain manner in order to minimize the number of active low-PPR hosts and maximize the resource utilization of the high-PPR hosts.

III. THE PROPOSED FRAMEWORK FOR VM CONSOLIDATION

The proposed dynamic VM consolidation framework is shown in Fig. 2. It consists of four layers including: user request, global manager, cluster manager, and the shared cloud storage layers. Different from our previous work [40], we have constructed the workload prediction module in the global manager, and developed a new RACC model as a separate module to support host status detection. In addition, we designed the VM migration module in the cluster manager, which enhanced the decoupling between the global manager and cluster manager.

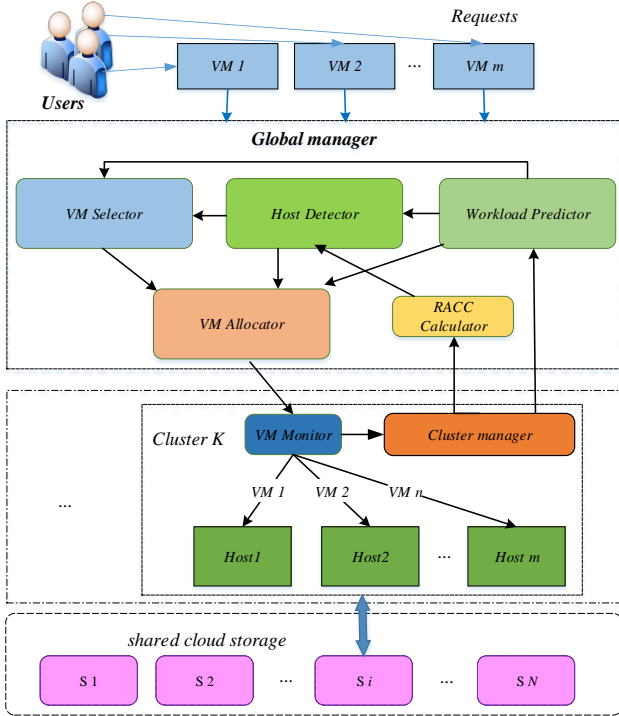


FIGURE 2 Proposed Dynamic VM Consolidation Framework (Adapted from our previous work [40])

- User request layer: The user request layer is an interface between the cloud service provider and consumers. It is responsible for receiving requests and establishing the SLA based on the consumer's preferences. If the SLA negotiation is successful, the user request layer sends the consumer requests to the global manager layer.
- Global manager layer: It consists of four key components. The *workload predictor* gathers the information from the *cluster manager* and predicts the resource utilization of

each physical host based on the developed prediction model. The *host detector* determines when a host is overloaded or underloaded, based on the proposed detection algorithms. The *VM selector* determines the VMs that should be migrated from an overloaded host based on the proposed VM selection algorithm. The *VM allocator* issues commands for the optimization of the VMs' placement, based on the proposed VM placement algorithm.

- Cluster manager layer: The *cluster manager* continuously monitors the resource utilization of the hosts in the cluster and resizes the VM according to their resource needs. The *VM Monitor* (VMM) resizes and migrates the VMs to other hosts.
- The shared cloud storage layer: A shared cloud storage system saves data, facilitates data-sharing between the all hosts, and provides users with a shared cloud storage resource; it also enables rapid live migration of the VMs.

We adopt the VM consolidation procedure defined in [9] that splits the VMCP in cloud data centers into four stages:

- (1) Overload detection: The *host detector* examines the physical hosts one by one, and using the overloading detection algorithm, determines whether a host is overloaded;
- (2) VM selection: If it is detected that a host is overloaded, the *VM selector* selects the VMs to be migrated from the overloaded host using the VM selection algorithm to eliminate hotspots;
- (3) VM placement: Once the list of VMs to be migrated from the overloaded hosts is compiled, the *VM allocator* is invoked to find a new placement for the VMs to be migrated based on the VM placement algorithm;
- (4) Underload detection: The *host detector* determines underloaded hosts using the underload detection algorithm, and the *VM allocator* finds a new placement for all the VMs from the underloaded hosts.

Owing to the heterogeneity of the cloud resource and variety of applications in the cloud environment, the workload on the hosts changes dynamically over time. It is critical to develop accurate workload prediction models for effective resource management and allocation. Therefore, the main difference between our VM consolidation framework and the one proposed in [9] is that the values of resource utilization used in the algorithms of VM consolidation are predicted using the *workload predictor*, rather than based on the current usage.

Through the four steps above, the VM consolidation framework can return a combined migration map that includes the information about the VM placement. Assuming that it takes one byte to record the mapping relationship between each VM and a physical machine, the scheduling interval of the consolidation procedure is T_c , and the data collection period of the *workload predictor* is T_d . Each load record includes five fields: timestamp, server ID, CPU, memory, and bandwidth. The data size of each load record occupies c bytes, and the additional communication

complexity generated by the consolidation framework is $O(n + cml \frac{T_c}{T_d})$, where l denotes the length of the historical load data, and m and n represent the number of servers and VMs, respectively.

IV. DESIGN OF KEY COMPONENTS OF PROPOSED FRAMEWORK

A. WORKLOAD PREDICTOR

Workload predictor is used for predicting usage of all resource types of physical hosts based on the proposed prediction model in this work.

As there is a strong correlation between the change in host utilization and time [23], we have proposed a workload prediction model called *weighted moving average* (WMA) that is predicated on the MA [24] and the IQR techniques. The MA technique is a well-known time-series prediction technique for various applications [24], [25]. However, the default MA technique uses a simple linear model for forecasting based on the current values [17]. Furthermore, it is highly sensitive to noise and instantaneous spikes. To address this issue, the IQR (a statistic measure of variability involving the division of data into quantiles, that is, four equal parts: Q_1 , Q_2 , Q_3 and Q_4) is used to increase the robustness of the model.

Mathematically, the WMA can be represented as follows:

$$WMA := \langle U_t^r, PF, \hat{u}_{t+1}^r \rangle$$

- (1) U_t^r : The input of the parameter of the WMA that is a set of utilization history of resource r (such as CPU, memory, and bandwidth) at time t . Let u_t^r represent the utilization of resource r at time t ; then the input parameter of the WMA can be formulated as $U_t^r = \{u_{t-l+1}^r, \dots, u_{t-1}^r, u_t^r\}$, where l denotes the history length.
- (2) \hat{u}_{t+1}^r : The output parameter of the WMA representing the predicted utilization of resource r at time $t + 1$.
- (3) PF: The prediction formula of the WMA. To enhance the robustness and prediction accuracy of the WMA, we first rank the set U_t^r in ascending order and find the first, second, and third quartiles of U_t^r , respectively. Then, we divide the values of U_t^r into a high-value set, H , and a low-value set, L , using the second quartile Q_2 . Finally, the final predicted utilization value \hat{u}_{t+1}^r is computed by calculating the average of two separate sets with different weight values. The model formulation of WMA is defined as follows:

$$\hat{u}_{t+1}^r = k \times \frac{\sum_{i=t-l+1}^t \{u_i^r \mid u_i^r \geq Q_2\}}{|H|} + (1-k) \times \frac{\sum_{i=t-l+1}^t \{u_i^r \mid u_i^r < Q_2\}}{|L|} \quad (1)$$

$$k = \frac{IQR}{Q_4 - Q_0} = \frac{Q_3 - Q_1}{Q_4 - Q_0} \quad (2)$$

where the coefficient k represents the weight of the higher values exceeding Q_2 in U_t^r , u_i^r is the i -th utilization value of r in the historical dataset, and Q_4 and Q_0 represent the maximum and minimum value of the time-series, respectively.

According to (1) and (2), we can observe that the IQR is directly proportional to the weight of H , and by extension, \hat{u}_{t+1}^r , as well. This is because a high IQR corresponds to a greater dispersion of the CPU utilization, thus increasing the likelihood of the CPU utilization to reach 100%. Therefore, the main idea of the proposed WMA is to predict the workload of the hosts precisely, depending on the historical degree of dispersion of resource utilization.

When the WMA is running, the *workload predictor* needs to communicate with the *cluster manager* to obtain the historical load data of each server. The computational bottleneck of the prediction model lies in sorting the historical load data. Because the length of the historical data is relatively small, the quicksort algorithm is adopted, and its worst-case time complexity and space complexity are $O(l^2)$ and $O(l)$, respectively, where l denotes the history length. Assuming that the data size of each load record occupies c bytes, then the communication complexity of the WMA can be formulated as $O(c * l * m)$, where m denotes the number of the servers in the cloud platform. In this study, since the value of the load data includes five fields (timestamp, server ID, CPU, memory and bandwidth), each field can be expressed as a real number, which usually occupies a small fixed number of bytes in the computer. This means that the data size c can be expressed as a fixed constant, and the final communication complexity can thus be expressed as $O(l * m)$.

Regarding the history length, l , in real scenarios, although increasing the length of the historical load data will be beneficial to the accuracy of load forecasting, it can be seen from the above complexity analysis that the computational complexity of WMA will increase exponentially with the increase in l ; therefore, we suggest that l should not be excessively large (30 in our experiment). With respect to data size, as analyzed previously, the data size of the load can be expressed as a fixed constant; therefore, it will not have a significant impact on the communication complexity of the WMA. In addition, when the server number of the data center is too large, we can consider implementing WMA load-balancing in a distributed mode.

B. HOST DETECTOR

The *host detector* is responsible for determining when a host is overloaded or underloaded based on the proposed detection algorithms.

1) OVERLOAD DETECTION ALGORITHM

The key of the overload detection algorithm is to determine a reasonable upper threshold for the host, based on the current RACC of the host. Because the CPU characteristics of each

host, such as frequency and number of cores, varies, it is unreasonable to evaluate the RACC of the host relying solely on CPU utilization. In this work, we determine the RACC based on the power consumption and PPR as follows:

$$RACC_{H_i} = (P_{H_i}(100\%) - P_{H_i}(U_{H_i})) \times PPR_{H_i} \quad (3)$$

where U_{H_i} represents the current CPU utilization of host H_i ; $P_{H_i}(100\%)$ and $P_{H_i}(U_{H_i})$ represent the energy consumption when the host CPU utilization is full and U_{H_i} , respectively; PPR_{H_i} represents the PPR of H_i . All the parameters in Eq. (3) can be obtained by conducting experiments on the host using SPECpower_ssj2008 [10].

Based on the RACC calculation formula, Eq. (3), we propose a novel host overload detection algorithm, named *RACC-aware Threshold* (RACCT), that uses the RACC, instead of the CPU utilization, as the upper threshold of the host. The pseudo-code of the RACCT is shown in Algorithm 1. First, it classifies all the hosts in the data center based on the PPR, and finds the optimum host H_{opt} with the maximum PPR value; then, the algorithm sets a fixed value of upper CPU utilization threshold, $UT_{H_{opt}}$, for the H_{opt} , and calculates the $RACC_{H_{opt}}$, according to Eq. (3), when the H_{opt} load is $UT_{H_{opt}}$. Finally, using the $RACC_{H_{opt}}$ as the upper threshold for all the hosts in the data center, that is, we perform a RACC-aware consolidation on another host, if it is detected that the RACC of a host is lower than $RACC_{H_{opt}}$, the host will be regarded as overloaded. Considering the strong computing capacity of the H_{opt} and the advantage of the prediction model (WMA), that is, the ability to effectively eliminate the instantaneous spikes, it is suggested that $UT_{H_{opt}}$ be set to close to 100% to fully maximize the computing resource of H_{opt} . The PPR of the host is inversely proportional to the upper utilization threshold, because a lower performance increases the likelihood of the CPU utilization reaching 100%, thus resulting in SLA violations. The complexity of the host overload detection is $O(m)$, where m is the number of servers.

Algorithm 1: Host overload detection

Input: Host List
Output: Overload Host List

1. $PPR_{H_{opt}} \leftarrow MIN$
2. For each host in Host List do {
3. If $PPR_{host} > PPR_{H_{opt}}$ {
4. $PPR_{H_{opt}} = PPR_{host}$
5. Optimum Host = host
6. }
7. }
8. $RACC_{H_{opt}} \rightarrow$ Calculate RACC of Optimum Host
 When the Load is $UT_{H_{opt}}$
9. For each host in Host List do {
10. $RACC_{host} \leftarrow$ Calculate RACC of host at time t
11. If $RACC_{host} < RACC_{H_{opt}}$ {
12. host is Overload
13. Add host to Overload Host List
14. }
15. }
16. Return Overload Host List

2) UNDERLOAD DETECTION ALGORITHM

The simplest and commonly used method for determining underloaded hosts is called the simple method [9], [17]; it is based on the minimum CPU utilization. However, as mentioned in earlier sections, due to heterogeneous hosts with different CPU capacity and power efficiency, determining underloaded hosts depends on not only CPU utilization but also the PPR of a host.

In this study, we propose a *multi-criteria underloaded host detection* (MCUHD) algorithm based on the Z-score [26] that scores each host based on CPU utilization and PPR, and the host with the highest score is detected as an underloaded host. The formula is defined in Eq. (4), where both the U_{H_i} and the PPR_{H_i} represent the cost criterion (a lower value is better), \bar{U} and \overline{PPR} represent the mean value of the CPU utilization and PPR of all hosts in the data center, respectively.

$$Score_{H_i} = - \frac{U_{H_i} - \bar{U}}{\sqrt{\frac{1}{N} \sum_{j=1}^N (U_{H_j} - \bar{U})^2}} - \frac{PPR_{H_i} - \overline{PPR}}{\sqrt{\frac{1}{N} \sum_{j=1}^N (PPR_{H_j} - \overline{PPR})^2}} \quad (4)$$

The pseudo-code of the MCUHD is shown in Algorithm 2; its complexity is $O(m)$, where m is the number of hosts. The benefits of the MCUHD is its ability to detect underload host with minimum CPU utilization and PPR, to ensure the minimization of energy consumption and performance degradation.

Algorithm 2: Host underload detection

Input: Host List
Output: Under loaded Host

1. Highest Score $\leftarrow MIN$
2. For each host in Host List do {
3. If Overload Host List Contains host {
4. Continue
5. }
6. If Switched-Off Host List Contains host {
7. Continue
8. }
9. Score_{host} \leftarrow score the host according to Eq. (4)
10. If Score_{host} > Highest Score {
11. Highest Score = Score_{host}
12. Under loaded Host = host
13. }
14. }
15. Return Under loaded Host

C. VM SELECTOR

Once an overloaded host is detected, the *VM selector* will select VMs from that host and migrate them to other hosts. Most existing approaches are heuristics based on the greedy strategy, often leading to suboptimal results.

In this study, we formulate the *VM selection problem* (VMSP) as a *0-1 knapsack problem* (0-1KP) and develop a VM selection algorithm, called MDT, based on dynamic

programming to obtain the optimal solution in polynomial time.

1) PROBLEM FORMULATION

The objective of the VM selection algorithm is to quickly eliminate the hotspots in the overloaded host while minimizing the duration of live migration, which is dependent on the amount of data transfer [9]. Therefore, the constraint optimization model for the VMSP can be defined as follows:

$$\min \sum_{i=1}^n v_i x_i$$

s.t. (5)

$$\sum_{i=1}^n w_i - \sum_{i=1}^n w_i x_i \leq C, \quad x_i \in \{0, 1\}, 1 \leq i \leq n$$

where C is the upper CPU-utilization threshold of the overloaded host that can be calculated using Eq. (3); x_i represents the question of whether or not to select the vm_i and migrate from the overloaded host; $x_i = 1$ indicates “move”, and $x_i = 0$ indicates “retain”; w_i and v_i are the CPU and memory utilization of vm_i , respectively.

Lemma 1. If $y_i = 1 - x_i$, then $\min \sum_{i=1}^n v_i x_i$ is equivalent to

$$\max \sum_{i=1}^n v_i y_i.$$

Proof. Because $\sum_{i=1}^n v_i x_i + \sum_{i=1}^n v_i y_i = \sum_{i=1}^n v_i$, and $\sum_{i=1}^n v_i$ is a

constant representing the total amount of memory used by the VMs that reside on the overloaded host. Therefore, when

$\sum_{i=1}^n v_i x_i$ takes the minimum value, $\sum_{i=1}^n v_i y_i$ gets the

maximum value, and vice versa. Therefore, $\min \sum_{i=1}^n v_i x_i$ is

$$\text{equivalent to } \max \sum_{i=1}^n v_i y_i.$$

According to Lemma 1, the constraint optimization model of VMSP can be rewritten as follows:

$$\max \sum_{i=1}^n v_i y_i$$

s.t. (6)

$$\sum_{i=1}^n w_i y_i \leq C, \quad y_i \in \{0, 1\}, 1 \leq i \leq n$$

where can be formally described as a 0-1KP: given the knapsack capacity C ($C > 0$), the weight and value of the i -th item w_i and v_i ($w_i > 0$, $v_i > 0$, $1 \leq i \leq n$), it is necessary to find out a n -element 0-1 vector $Y = \{y_1, y_2, \dots, y_n\}$, $y_i \in \{0, 1\}$, $1 \leq i \leq n$, which makes

$\sum_{i=1}^n w_i y_i \leq C$; it is also necessary to maximize $\sum_{i=1}^n v_i y_i$. By solving 0-1KP, vector Y can be obtained, and the final set of migratable VMs can be achieved through $\bar{1} - Y$.

2) THE PROPOSED ALGORITHM FOR VM SELECTION

We adopt the dynamic programming method introduced in [41] to solve the 0-1KP, and on this basis, propose our VM selection algorithm, named MDT. In the MDT, the VMs are conceived as items; the size of the CPU and memory requested by the VM are conceived as the weight and value of the item, respectively; the upper CPU utilization threshold of the host is conceived as the knapsack capacity. To facilitate the solution of the 0-1KP, we referred to the load data set in CoMon project [38] and discretized the resource utilization of the VM and server into positive integers within the range of 0 to 100 by the rounding method. To obtain the 0-1 vector, Y , the MDT uses matrix $V[n][C]$ to record the placement status of the items in a recursive process; the calculation formula of the elements in $V[n][C]$ is defined as shown in Eq. (7).

$$V[i][j] = \begin{cases} 0, & i=0 \text{ or } j=0 \\ V[i-1][j], & j < w_i \\ \max \{V[i-1][j], V[i-1][j-w_i] + v_i\}, & j \geq w_i \end{cases} \quad (7)$$

where $V[i][j]$ represents the maximum value of the first i items loaded into the knapsack with a capacity of j . Finally, the VMs to be migrated are the items that could not be loaded into the knapsack. This approach not only ensures the overloaded hosts quickly eliminates the hotspot incurred by the CPU shortage but also can minimize the total memory consumed by the VMs to be migrated. In other words, we restore an overloaded host back to the normal state with the minimum amount of data transfer.

Algorithm 3: VM selection

Input: Overloaded Host
Output: Migratable VM List

1. $VM\ List \leftarrow$ Get VMs From Overloaded Host
2. $N \leftarrow$ Obtain Size of VM List
3. $C \leftarrow$ Calculate Upper Threshold for Overloaded Host
4. $Y = \{y_i | y_i = 0, 1 \leq i \leq N\}$
5. For $i = 1$ to N do {
6. For $j = 0$ to C do {
7. If $j < w_i$
8. $V[i][j] = V[i-1][j]$
9. Else
10. $V[i][j] = \max(V[i-1][j], V[i-1][j-w_i] + v_i)$
11. }
12. }
13. $j = C$
14. For $i = N$ to 1 {
15. If $V[i][j] > V[i-1][j]$ {
16. $y_{i-1} = 1$
17. $j = j - w_i$
18. }
19. }
20. For each y_i in Y {
21. If $1 - y_i = 1$
22. Add i -th VM in $VM\ List$ to Migratable VM List
23. }
24. Return Migratable VM List

The pseudo-code of the MDT is shown in Algorithm 3, and its time and space complexity is $O(n \times C)$, where n is

the number of the items (i.e. VMs), and C is the capacity of the knapsack.

D. VM ALLOCATOR

The objective of the *VM allocator* is to find a new placement for the VMs to be migrated to minimize energy consumption due to allocation.

The VM placement can be seen as a multi-capacity bin-packing problem [9], [27]. One popular policy used in [9] is the PABFD, which sorts all the VM in decreasing order of their current CPU utilizations and then allocates each VM to a host that triggers the least increase in the power consumption following the allocation. However, there is a potential problem in this approach. As mentioned earlier, the lower the PPR of the host is, the lower the performance, and the more likely it is to become overloaded over the next time series. The existing PABFD approach does not account for the host's PPR when allocating the VM; thus, it is possible to assign a VM to a host with a low PPR but a high workload, which can increase the frequency of migration. To obtain the optimal VM placement, we extend the existing PABFD approach by considering the PPR of heterogeneous hosts, allocating a VM to a host with a higher PPR while ensuring the least increase in energy consumption. The core idea behind this is that the high-PPR hosts consume less energy per operation, compared to the others, and they have greater resource capacity, which can decrease the frequency of live migration [19].

The modified algorithm of PABFD is shown in Algorithm 4. The complexity of the algorithm is $O(n \times m)$, where m is the number of hosts and n is the number of VMs that have to be allocated.

Algorithm 4: VM placement

Input: Host List, VM List
Output: Migration Map

1. Host Categorized Based on PPR
2. $N \leftarrow$ Obtain Number of Host Category
3. For each VM in VM List {
4. Minimum Power \leftarrow Max
5. Allocated Host \leftarrow NULL
6. For $n=N$ to 1 do {
7. For each host in Category n do {
8. If host has Enough Resource for VM {
9. Power \leftarrow Estimate Power Consumption caused by this allocation
10. If Power < Minimum Power {
11. Minimum Power = Power
12. Allocated Host = host
13. }
14. }
15. }
16. If Allocated Host \neq NULL {
17. Add (VM, Allocated Host) to Migration Map
18. Break
19. }
20. }
21. }
22. Return Migration Map

V. EVALUATION METRICS

To evaluate the efficiency of the proposed algorithms based on four commonly used metrics: energy consumption, live migration cost, SLAV, and ESV.

A. ENERGY CONSUMPTION

With the development of multi-core CPUs and virtualization, the power consumption ratio of CPUs is constantly decreasing [9]. In addition, modern hosts are typically equipped with large memory that tend to dominate the power consumption in a host [22]. This fact, combined with the difficulty of modeling power consumption in modern data centers, makes building precisely analytical models a complex research problem [9]. Therefore, instead of using an analytical model of power consumption by a server, we utilize real data on power consumption provided by the results of the SPECpower benchmark [10], as shown in Table II.

However, because the workload of the server is not always exactly at levels that are integer multiples of 0.1, it is necessary to propose a reasonable estimation method to obtain the energy consumption of the server functioning at any utilization level. Without loss of generality, we assume that power consumption increases linearly between two adjacent workload values in Table II. Consequently, linear interpolation can be used to approximate the energy consumption of the server under any workload, as shown in Eq.(8).

$$P(U) = \frac{P(U_h) - P(U_l)}{U_h - U_l} (U - U_l) + P(U_l) \quad (8)$$

In Eq.(8), U is the given CPU utilization, and U_l and U_h are two adjacent workload values in Table II satisfy the equation, $U_l \leq U \leq U_h$ and $U_h - U_l = 0.1$, $P(U_l)$ and $P(U_h)$ are the energy consumption of the host when the workload is U_l and U_h , respectively.

$$E_{server} = \int_{T_f}^{T_l} P(U(t)) dt$$

$$= \sum_{n=1}^{N-1} \left[\frac{P(U(T_s + (n-1)*T_p)) + \frac{P(U(T_s + n*T_p))}{2} - \frac{P(U(T_s + (n-1)*T_p))}{2}}{2} \right] * T_p \quad (9)$$

According to Eq.(8), the total energy consumption of the server during the active period can be calculated using Eq.(9), where $U(t)$ represents the CPU utilized by the server at time t ; N ($N \geq 2$) is the number of workload samples taken when the server is running; T_p is the sampling period; and T_f and T_l represent the time of the first and last sampling, respectively, in conformity with $T_l = T_f + (N - 1)T_p$.

Finally, the energy consumption of the data center during operation can be expressed as Eq.(10), where m is the number of hosts in the datacenter, d is the number of days over which the experiment is performed, $E_{server}(i, j)$ represents the energy consumption of the i -th server on the j -th day.

$$E_{datacenter} = \sum_{i=1}^m \sum_{j=1}^d E_{server}(i, j) \quad (10)$$

B. LIVE MIGRATION COST

The live migration of VMs makes it possible to transfer a VM from the source node to the destination node with minimum downtime and no suspension. Although the live migration is transparent to end users, it can cause performance degradation of applications running in the VM [13]. To quantify the cost of live migrations, the migration time and performance degradation experienced by a VM_j are given in (11) and (12), as proposed in [9], and the total transmitted data during the entire migration is formulated as Eq. (13).

$$T_{m_j} = \frac{M_j}{B_j} \quad (11)$$

$$C_{d_j} = 0.1 \times \int_{t_0}^{t_0 + T_{m_j}} u_j(t) dt \quad (12)$$

$$Data_m = \sum_{j=1}^n M_j \quad (13)$$

where T_{m_j} is the time taken to complete the migration, C_{d_j} is the total performance degradation by VM_j , $Data_m$ is the total amount of transmitted data, t_0 is the commencement time of migration, $u_j(t)$ is the CPU utilization of VM_j ; M_j is the memory consumption of VM_j ; B_j is the available network bandwidth; and n is the number of migrated VMs.

C. SLA VIOLATION

The QoS is an extremely important indicator for assessing the cloud platform's efficiency. Owing to the different applications run by VMs applied by different users and their varied demands on bandwidth, response time, and throughput, it is necessary to define a load-independent metric to evaluate the SLAV for the VMs deployed in the data center [17]. In this study, a modified version of the SLAV metric proposed in [9] is defined in Eq. (14), which is calculated by combining the *SLA violation time per active host* (SLATAH) and *performance degradation due to migrations* (PDM), as defined in Eq. (15).

$$SLAV = SLATAH \times PDM \quad (14)$$

$$SLATAH = \frac{1}{N} \sum_{i=1}^N \frac{T_{s_i}}{T_{a_i}}, \quad PDM = \frac{1}{Q} \sum_{j=1}^Q \frac{C_{d_j}}{C_{r_j}} \quad (15)$$

In the default SLATAH introduced in [9], T_{s_i} is defined as the total time during which the capacity of the host i is taxed to the maximum, leading to an SLA violation. In a real scenario, however, it is possible that the workload taxes the physical node's capacity to the maximum. but the VM is properly provisioned. In addition, oversubscription is a common practice in cloud computing; it enables the providers to allocate more resource to users beyond the actual capacity of their hosts, provided that they do not break SLAs. On account of this, we modify the T_{s_i} in Eq. (15) into the total time during which allocated resource to the VM i is lower than the user's requested resource when the host is maximally utilized. T_{a_i} is the total time of the host i being in the active state; Q is the number of VMs; C_{d_j} is the estimate of the performance degradation of the VM_j caused by migrations; C_{r_j} is the total CPU capacity demanded by the VM j during its lifetime; and N is the number of hosts.

D. ENERGY-SLAV

The objective of the dynamic VM consolidation is to minimize energy consumption and SLA violations. To evaluate the comprehensive performance of the proposed algorithms, a metric introduced in [9] that is the product of the energy consumption and SLAV is adopted, as shown in Eq. (16).

$$ESV = E_{datacenter} \times SLAV \quad (16)$$

VI. EXPERIMENTAL EVALUATION

To confirm the effectiveness of the proposed method, we have performed experimental evaluations in both real and simulated environments. In the real environment, we implemented our algorithms on a small-scale cloud platform built with OpenStack [32], with the aim of proving its reliability and superiority in real infrastructure. In the simulated environment, we built a large-scale simulation cloud platform using CloudSim [33] and experimented with our algorithms using real workload datasets to evaluate the scalability of the proposed algorithms in large-scale cloud datacenters.

A. EXPERIMENTAL SETUP

1) EXPERIMENTAL SETTINGS — VALIDATION ON OPENSTACK IN REAL CLOUD ENVIRONMENT

To evaluate the reliability and superiority of the proposed method in the real cloud datacenter, a heterogeneous cloud platform based on OpenStack and Ceph [34] is constructed; it consists of one control nodes, six computing nodes, and three storage nodes, as shown in Figure 3. The components of the global manager layer are deployed on the control node, the *cluster manager* resides on each computing node, and the shared cloud storage system is built based on Ceph in the storage node.

The functionality of each component in the proposed framework is implemented by invoking OpenStack's Restful API using Java language, and the scheduling interval of the resource allocation algorithms is set to 10 min. The *cluster manager* collects the host resource usage information using

OSHI [35], a free JNA-based operating system and hardware information library for Java, and it samples the VM resource usage information through the ceilometer [36], the native resource monitoring component of OpenStack. The data acquisition intervals of the physical machines and the VM are 30 s and 3 min, respectively.

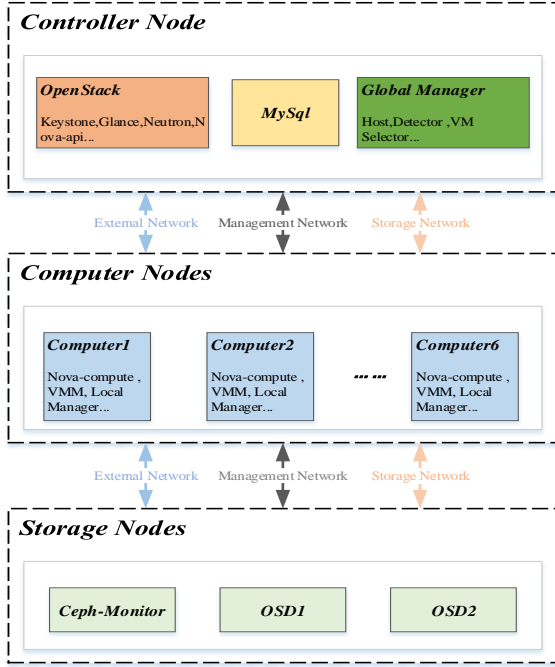


FIGURE 3 Cloud Platform Architecture in Real Environment

TABLE I
CHARACTERISTICS OF HOSTS

Host	CPU model (Intel)	Cores	Frequency (MHz)	RAM (GB)	PPR
Computer1/2	E5-2660	40	2200	256	5244
Computer3/4	E5-2470	24	2300	32	3501
Computer5/6	E5-2609	12	2000	128	2019

TABLE II
POWER CONSUMPTION OF HOSTS FOR DIFFERENT LOAD LEVELS IN WATTS

Load	Computer1/2	Computer3/4	Computer5/6
Idle	97.1	52.7	99.6
10%	107	61.5	138
20%	118	66.9	153
30%	128	72.9	169
40%	138	80.8	187
50%	151	90.8	209
60%	163	102	231
70%	176	112	249
80%	189	119	278
90%	217	143	311
100%	232	156	325

The characteristics of the computing nodes are presented in Table I. The energy consumption data of the physical machines, as shown in Table II, are obtained by performing experiments on the host with SPECpower_ssj2008 [10]. In Table III, three types of VMs specifications that correspond to Alibaba ECS VMs, are used (memory was reduced because of bandwidth limitations in the experimental environment).

TABLE III
SPECIFICATIONS OF VMs IN REAL ENVIRONMENT

Flavor Type	Med	Small	Micro
Cores	4	3	2
Memory (MB)	1024	528	256
Disk (G)	30	15	10

To create opportunities for dynamic consolidation, a random amount of CPU pressure was imposed on the VM at each moment of state-changes using LinuxStress [37]. Because the exponential distribution is frequently used to model the interval between the state-changes in continuous processes, we sample from an exponential distribution with a given rate parameter (in this experiment, we assume that the CPU usage of a VM changes once every 30 s) to determine the time of the next CPU utilization change.

To verify the effectiveness of the proposed algorithms in different resource allocation states of the cloud platform, we divide the experimental process into three phases, each lasting for one day, as shown in Table IV.

Under-Subscription: In this phase, 35 VMs with different specifications are launched, and the number of CPU core requests was 90, less than the total number of CPU cores of the cloud platform.

Adequate-Subscription: In this phase, 56 VMs with different specifications are launched, and there were 152 CPU core requests, equivalent to the total number of CPU cores of the cloud platform.

Over-Subscription: In this phase, 75 VMs with different specifications are launched, and the number of CPU core requests are 200, greater than the total number of CPU cores on the cloud platform.

During the experiment, the history length of the time series used in the WMA policy is 30; the upper threshold of CPU utilization for the optimum host is set to be 96%, and the over-allocation ratio of CPU resource in OpenStack is defined as 1.5. To ensure repeatable experimental environment, the initial placement of the VMs conformed to the default policy of OpenStack.

TABLE IV
EXPERIMENTAL PROCESS IN REAL ENVIRONMENT

Experimental Period	Number and Flavor of VM		Requested CPU Cores	Resource Allocation Phases
	Num	Flavor		
1st day	20	Micro	90	Under-Subscription
	10	Small		
	5	Med		
2nd day	26	Micro	152	Adequate-Subscription
	20	Small		
	10	Med		
3rd day	40	Micro	200	Over-Subscription
	20	Small		
	15	Med		

2) EXPERIMENTAL SETTINGS—VALIDATION ON CLOUDSIM

Scalability is an important metric for evaluating the effectiveness of algorithms. Due to the difficulty of conducting repeatable large-scale experiments using real infrastructures, simulations have been chosen as the realistic way to evaluate the scalability of the proposed algorithms.

CloudSim, a commonly used simulation tool in cloud computing, has been chosen for our simulation. In CloudSim, we first simulated a large-scale computing cluster composed of 900 heterogeneous physical hosts, including 300 HP ProLiant DL360 Gp4, 300 HP ProLiant ML110 G4, and 300 HP ProLiant ML G5, to focus on the performance of the proposed approach in general scenarios. Then, a cloud computing environment with three geographically distributed clusters is constructed to focus on the specific communication overhead of the multiple data centers.

The specifications of each cluster and characteristics of the hosts are presented in Tables V and VI, respectively. Each host is modeled to have 1 GB/s network bandwidth, and the bandwidth between different clusters is 100 MB/s. The energy consumption data is provided by the results of the SPECpower benchmark [10]. As shown in Table VII, five types of VMs specifications that correspond to Amazon EC2 are used.

TABLE V
SPECIFICATION OF CLUSTERS IN SIMULATED ENVIRONMENT

Cluster	ProLiant Gp4	ProLiant G4	ProLiant G5
Cluster1	0	0	300
Cluster2	100	100	100
Cluster3	300	0	0

TABLE VI
CONFIGURATION OF HOSTS IN SIMULATED ENVIRONMENT

Host	CPU model	Cores	Frequency (MHz)	RAM (GB)
ProLiant Gp4	Intel 930	2	3400	6
ProLiant G4	Intel 3040	2	1860	4
ProLiant G5	Intel 3075	2	2600	4

TABLE VII
SPECIFICATION OF VMs IN SIMULATED ENVIRONMENT

VM Type	Large	Med	Small	Micr o	Nano
Processor (Mips)	2500	2000	1000	500	250
Memory (MB)	2048	2048	1024	1024	512
Bandwidth(GB/s)	1	1	1	1	1

It was important to use a real workload for the simulation experiment; thus, we have used real workload traces with a sampling time of 30 days, as obtained from the CoMon project [38] and Google clusterdata-2011-2 [43] to evaluate the scalability of our approach. In the dataset, the VMs' workload-trace usage data is reported every 5 min from thousands of VMs, and the date of each day is shown in Table VIII. It is worth mentioning that the tasks in Google workload traces in real environment are not running within VM, but in Linux containers. To facilitate simulation, we characterize and cluster the tasks of Google workload traces based on the job ID and machine ID of the tasks and assume that each job consisting of tasks with the same job ID is assigned to run in a separate VM. During the simulations, each VM is randomly assigned a workload trace from one of the VMs from the corresponding day. The history length of the time series used in the WMA technique is equal to 30; the upper threshold of CPU utilization for the optimum host is set to 98%.

TABLE VIII
WORKLOAD DATA CHARACTERISTICS

workload datasets	Date	Number of VMs	Mean (%)	SD (%)
CoMon Project Workload Trace (see [9])	03/03/2011	1052	12.31	17.09
	06/03/2011	898	11.44	16.83
	09/03/2011	1061	10.70	15.57
	22/03/2011	1516	9.26	12.78
	25/03/2011	1078	10.56	14.14
	03/04/2011	1463	12.39	16.55
	09/04/2011	1358	11.12	15.09
	11/04/2011	1233	11.56	15.07
	12/04/2011	1054	11.54	15.15
	20/04/2011	1033	10.43	15.21
Google Workload Trace	1st day	1657	7.22	13.36
	2nd day	1657	6.93	12.64
	3rd day	1657	6.96	12.72
	4th day	1657	6.86	12.52
	5th day	1657	6.93	12.63
	6th day	1657	6.86	12.5
	7th day	1657	7.01	12.9
	8th day	1657	6.99	12.7
	9th day	1657	6.96	12.76
	10th day	1657	6.84	12.46
	11th day	1657	6.9	12.77
	12th day	1657	6.92	12.68
	13th day	1657	6.92	12.57
	14th day	1657	6.89	12.55
	15th day	1657	6.9	12.55
	16th day	1657	6.83	12.44
	17th day	1657	6.83	12.59
	18th day	1657	6.83	12.58
	19th day	1657	6.94	12.66
	20th day	1657	6.97	12.74

B. COMPARISON STUDY AND DISCUSSION

We compare our approach, RACC-MDT, with the existing approaches: (1) *Local regression and minimum migration time policy* (LR-MMT) [9]; (2) *VM-based dynamic threshold and minimum correlation of host utilization policy* (VDT-UMC) [14]; (3) *Dynamic threshold and maximum-fit policy* (DTH-MF) [19]. These approaches are selected, because they are the most popular VM consolidation algorithms. Furthermore, they are similar to our research that considers the four phases of energy-aware consolidation process.

1) VALIDATION ON OPENSTACK

• Energy Consumption Comparison

The comparison results of energy consumption under different resource allocation phases are shown in Figure 4. In details, when the resource allocation of the cloud platform is in the phase of *under-subscription* and *adequate subscription*, the two algorithms, RACC-MDT and DTH-MF, that consider the effect of the PPR of heterogeneous hosts have the lowest energy consumption while the algorithms that do not consider the PPR have a higher energy consumption. When the cloud platform is in the phase of *over-subscription*, the energy consumption of the data center varies slightly under the four approaches.

The reason for the above results is that in the initial process of VM allocation, computer3 and computer4 are configured with the minimum memory resource compared with others in cluster (but the PPRs of computer3 and computer4 are greater than those of computer5 and computer6); therefore, when the cloud platform is in the

phase of *Under-Subscription* and *Adequate-Subscription*, a small number of VMs are allocated in computer3 and computer4, which drastically reduces the CPU utilization of computer3 and computer4. LR-MMT and VDT-UMC take the CPU utilization of physical host and the number of VMs in physical host as evaluation metrics for underloaded host detection. Thus, computer3 and computer4, with their higher PPR values, were detected as the hosts with insufficient load. In contrast, RACC-MDT and DTH-MF account for the CPU utilization and the PPR difference between heterogeneous hosts; computer5 and computer6, with the lower PPRs, will be detected as underloaded hosts, which consequently have helped improve the power efficiency of the entire data center. When the cloud platform is in the phase of *Over-Subscription*, there are many VMs and higher workload in each host. It is difficult to find suitable hosts to reallocate all VMs in the underloaded host. Thus, the four approaches have similar effects in terms of energy-saving.

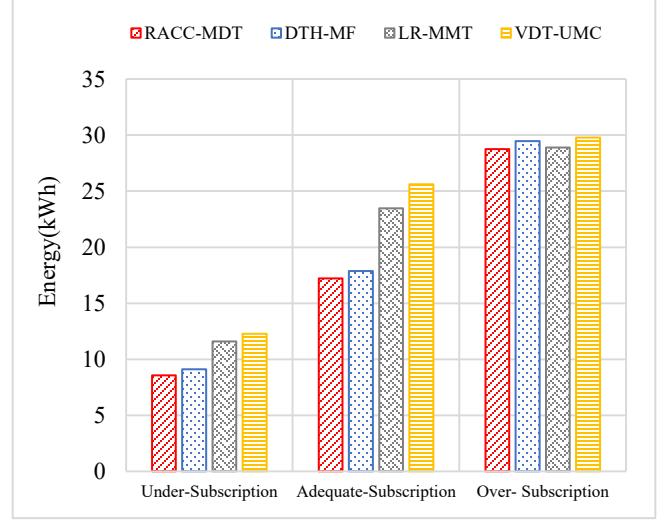


FIGURE 4. Energy Consumption Comparison

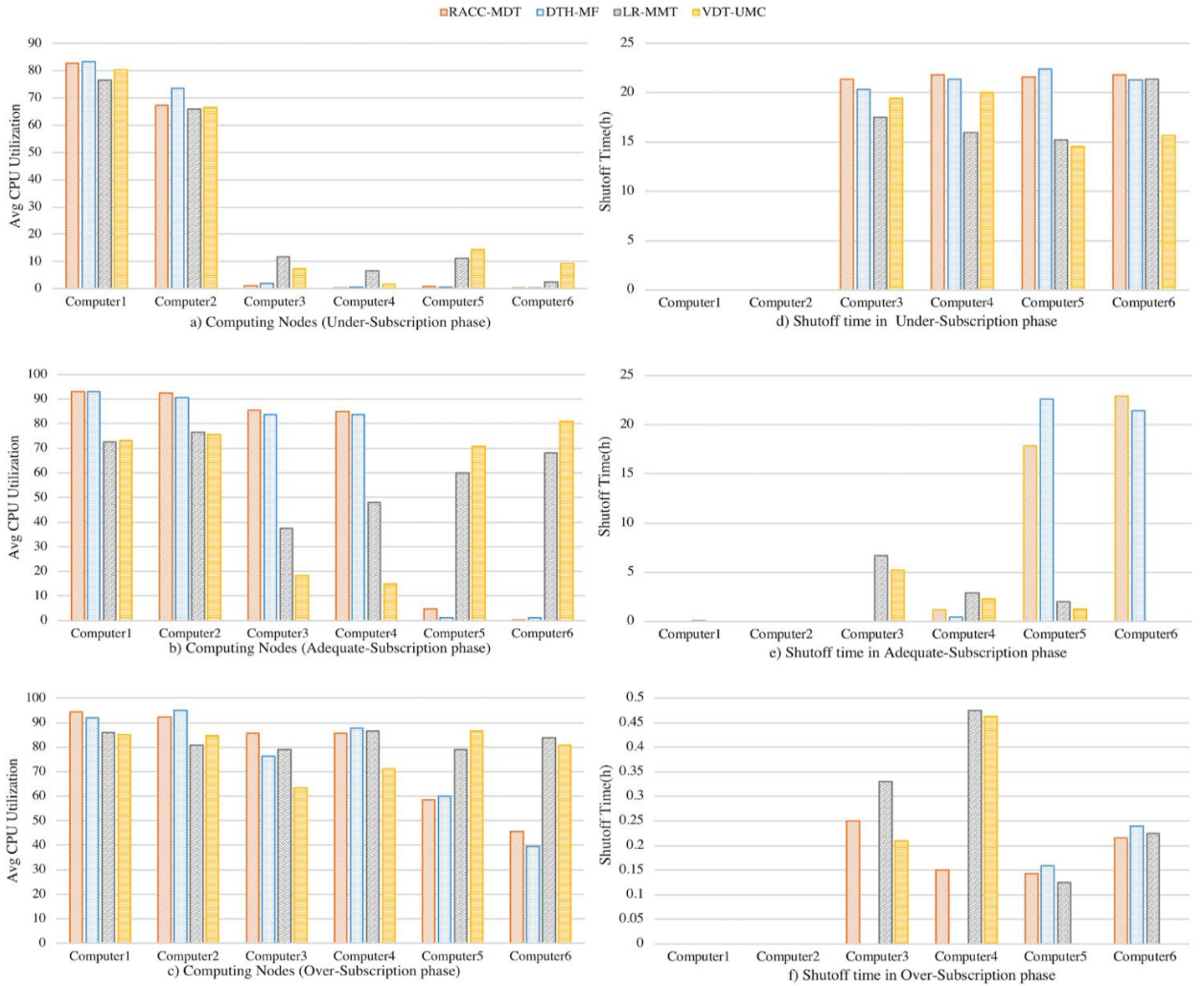


FIGURE 5. Average CPU Utilization and Shutoff Time of Each Computing Node in Different Experimental Phases

The results shown in Figure 5 can be seen as evidence for the above analysis. Figure 5 describes the average CPU utilization and shutdown time of each computing node under

different resource allocation stages. It is found that in each stage of the experiment, under the RACC-MDT and DTH-MF approaches, the resource utilization efficiency and active

time of high PPR hosts are higher than that of low PPR hosts, while the LR-MMT and VDT-UMC approaches only have this characteristic in the *Under-Subscription* phase. Figure 6 describes the average CPU utilization of the whole data center under different workloads. Under the same load, it is found that two algorithms RACC-MDT and DTH-MF considering the effect of the PPR of heterogeneous hosts have the lowest CPU utilization.

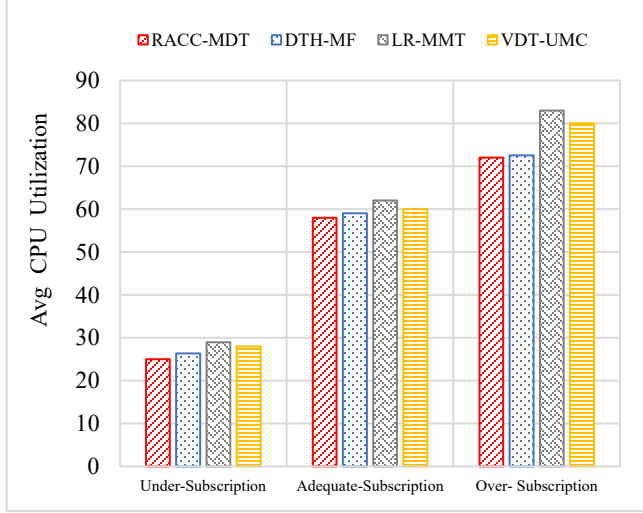


FIGURE 6. Average CPU Utilization of the Data Center Under Different Loads

It should be noted that although the DTH-MF policy accounts for the PPR differences of heterogeneous hosts, it migrates the VM to the hosts with the highest PPR and the lowest resource utilization in the VM allocation stage, which does not guarantee the minimum increment of energy consumption in the data center. The lowest energy consumption belongs to RACC-MDT (18.175 kWh) in comparison with DTH-MF (18.811 kWh), LR-MMT (21.316 kWh) and VDT-UMC (22.556 kWh).

• Live Migration Cost Comparison

The comparison results of the number of VM migrations in different experimental stages are shown in Figure 7. It is found that our approach RACC-MDT outperforms the DTH-MF and LR-MMT; however, the VDT-UMC outperforms it, except in the *over-subscription* phase.

The RACC-MDT outperforms the VDT-UMC because the former takes the number of VMs into account for host underload detection; the RACC-MDT, on the other hand, focuses on the total amount of data transfer when selecting VMs from the overloaded host. This is also evident in Figure 8. Although the number of VM migrations is least under the VDT-UMC algorithm, the amount of data transfer is more than that of the RACC-MDT. Compared with the VDT-UMC, the RACC-MDT reduces the amount of data transfer by an average of 7.97%. Furthermore, compared with the existing approaches, the RACC-MDT reduces data transfer by 29.1%, compared to the DTH-MF, and 24.9%, compared to the LR-MMT.

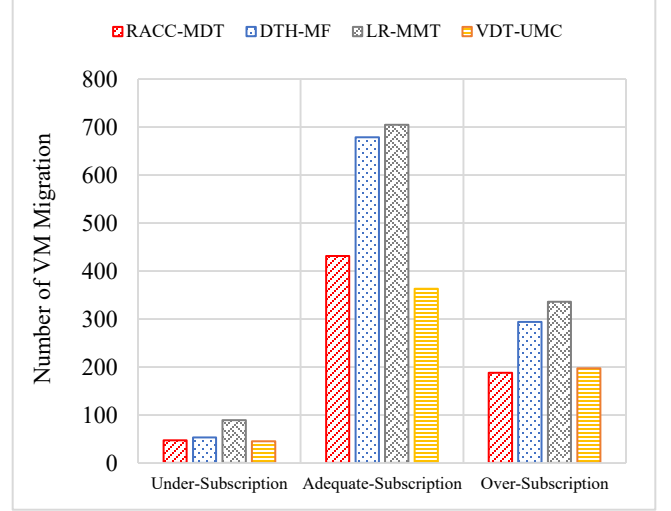


FIGURE 7. Number of VM Migrations Comparison

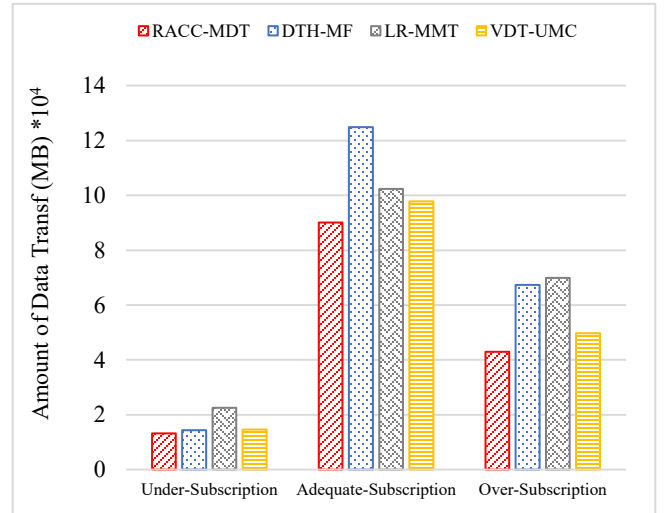


FIGURE 8. Amount of Data Transfer Comparison

• SLA Violation Comparison

Figure 9 demonstrates the comparison results of the SLAV metric. It suggests that in the *under-subscription* phase, the SLAV did not differ significantly across the four approaches; however, in the *adequate-subscription* and *over-subscription* phases, the proposed solution, RACC-MDT, and the VDT-UMC in [14] have the lowest SLAV, compared to the LR-MMT and DTH-MF.

The above results may be attributed to the fact in the *under-subscription* phase, the VMs are few, and the workload on each host is low; therefore, the probability of SLAV occurring is low. However, in the *adequate-subscription* and *over-subscription* phases, as the number of VMs and the workload on each host increase, the effects of

the four approaches on reducing SLA violations, in conformity with the definition of SLAV, become more apparent. From Eq. (14), SLAV is obtained by multiplying the SLATAH and PDM; thus, by reducing one of the variables (SLATAH or PDM), the final answer (SLAV) is also reduced. As shown in Figures 7 and 8, we conclude that the RACC-MDT and VDT-UMC reduced the amount of data transfer and the number of live migration, respectively, the implication of each being that less PDM is obtained. Therefore, the SLAV is reduced.

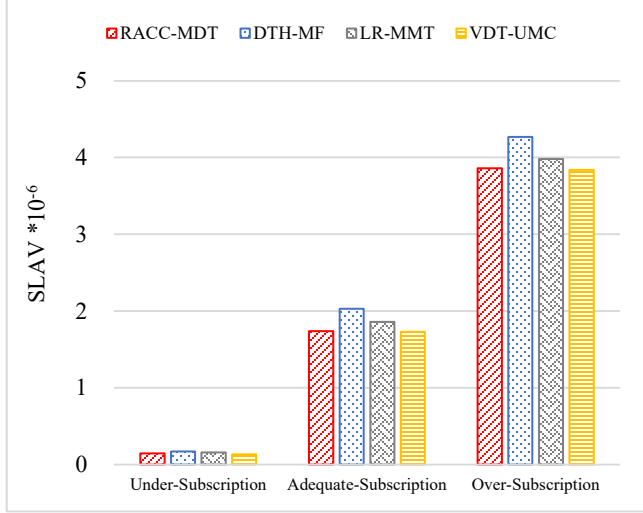


FIGURE 9. SLAV Comparison

• Energy-SLA Violations Comparison

The ESV metric comparison is shown in Figure 10. The value of the ESV metric is inversely proportional to better performance of the approach. It is found that, regardless of the current status of resource allocation on the cloud platform, the proposed RACC-MDT has the lowest results. Compared

with the existing approaches (LR-MMT, DTH-MF, VDT-UMC), the average ESV of the RACC-MDT over three days decreased by 11.4%, 12.7%, and 11.2%, respectively. This observation accords with the Figure 4 and Figure 8. Additionally, the proposed WMA model is more accurate than the existing approaches, which significantly improves the output results.

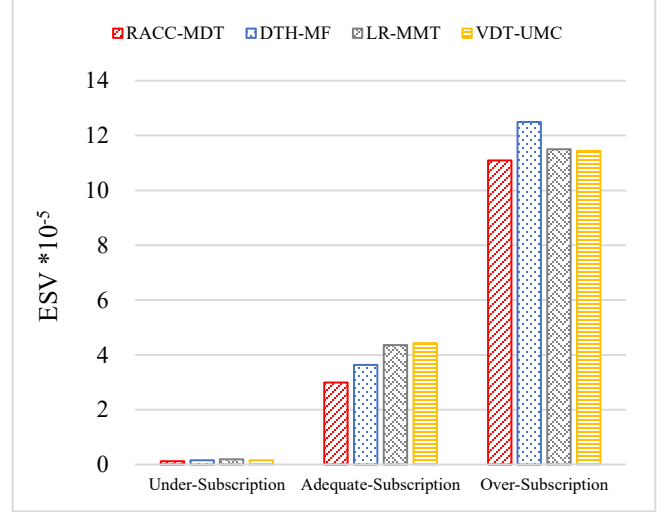


FIGURE 10. ESV Comparison

2) VALIDATION ON CLOUDSIM

The comparison results of the evaluation metrics in the single-cluster simulated environment are shown in Figures 11–16. It is found that the proposed method, RACC-MDT, offers significant advantages for minimizing energy consumption (Figures 11 and 12) and total data transfer (Figure 13), which further proves the effectiveness of the RACC-MDT in improving energy efficiency and reducing data traffic in data centers.

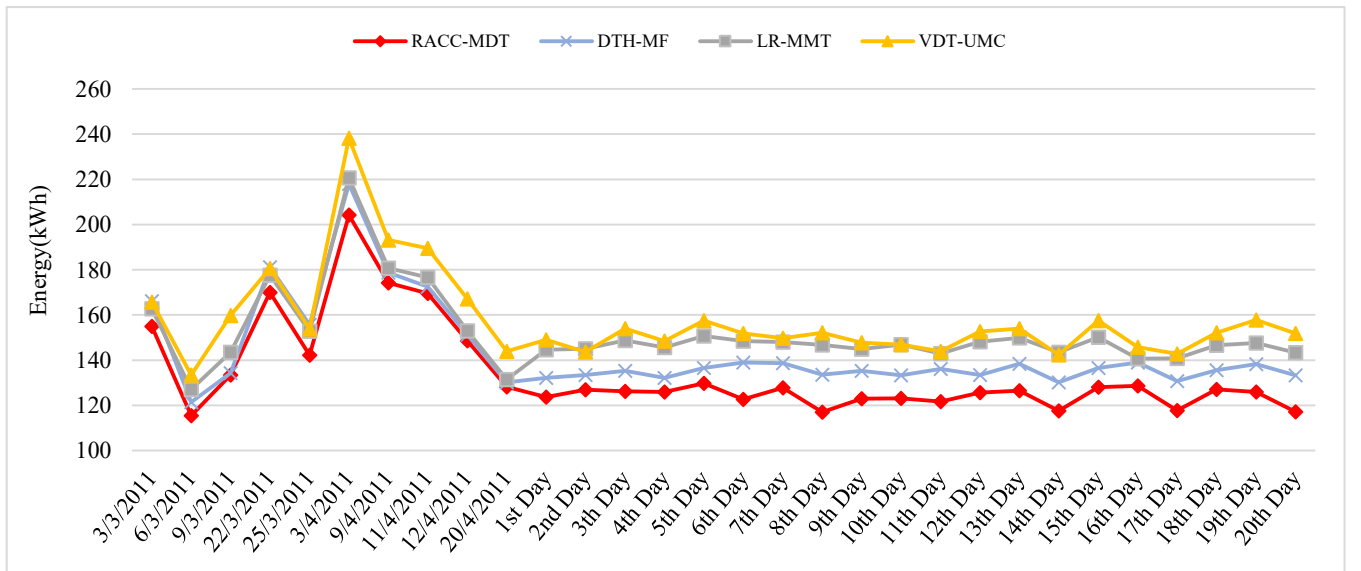


FIGURE 11. Energy Consumption Comparison of Different Approaches in Simulated Environment

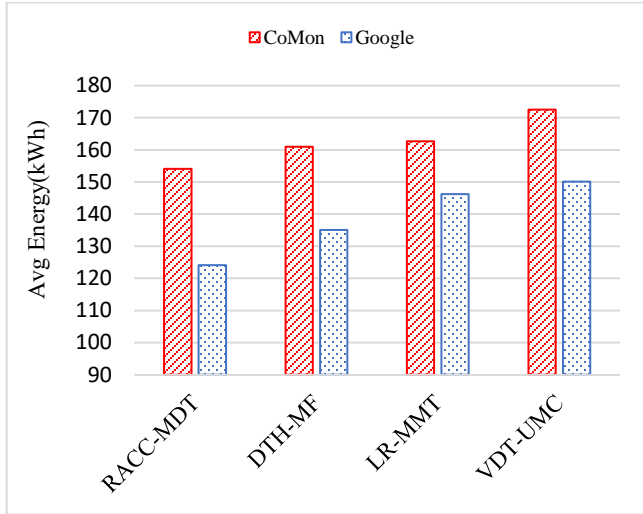


FIGURE 12. Average Energy Consumption of Different Approaches

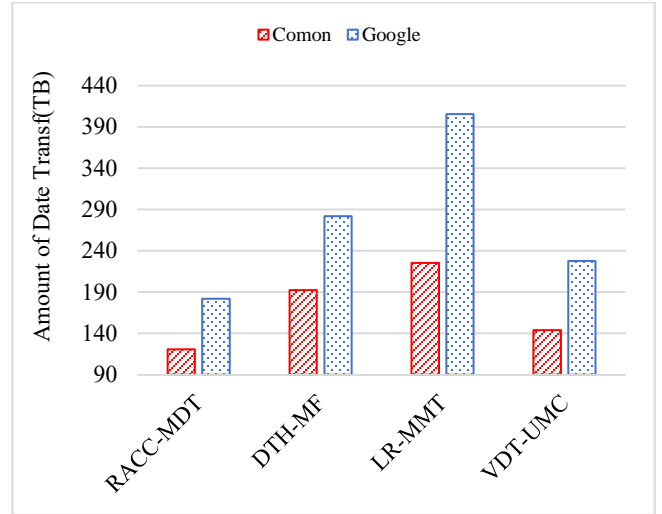


FIGURE 13. Average Data Transfer of Different Approaches

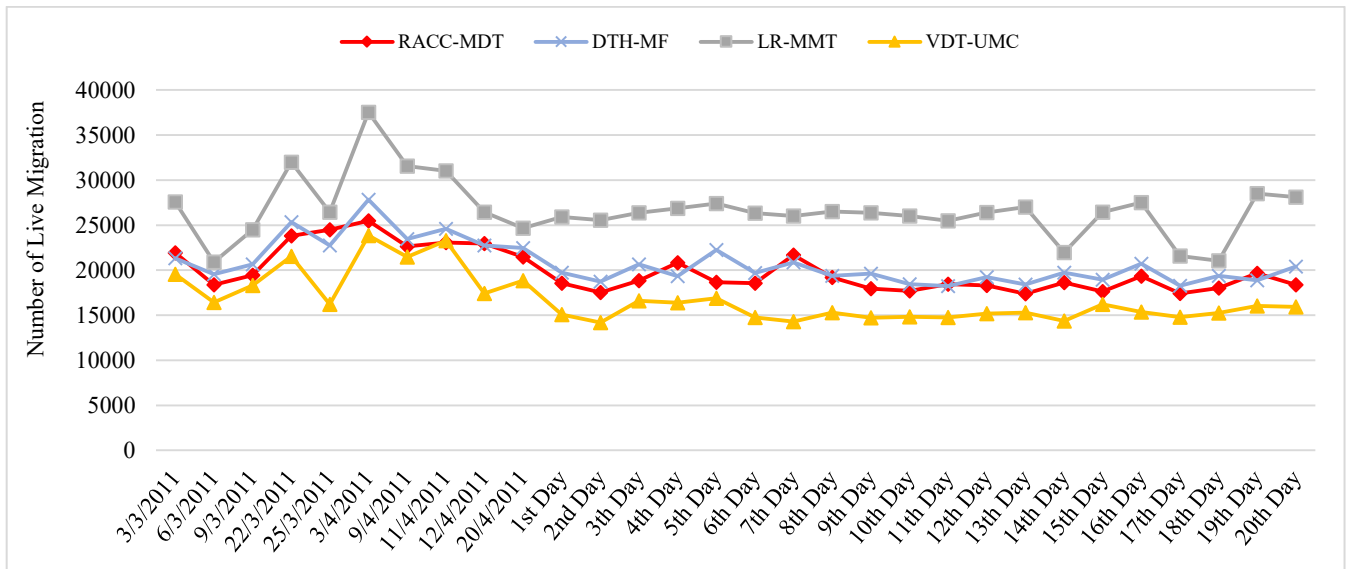


FIGURE 14. Number of Live Migration of Different Approaches in Simulated Environment

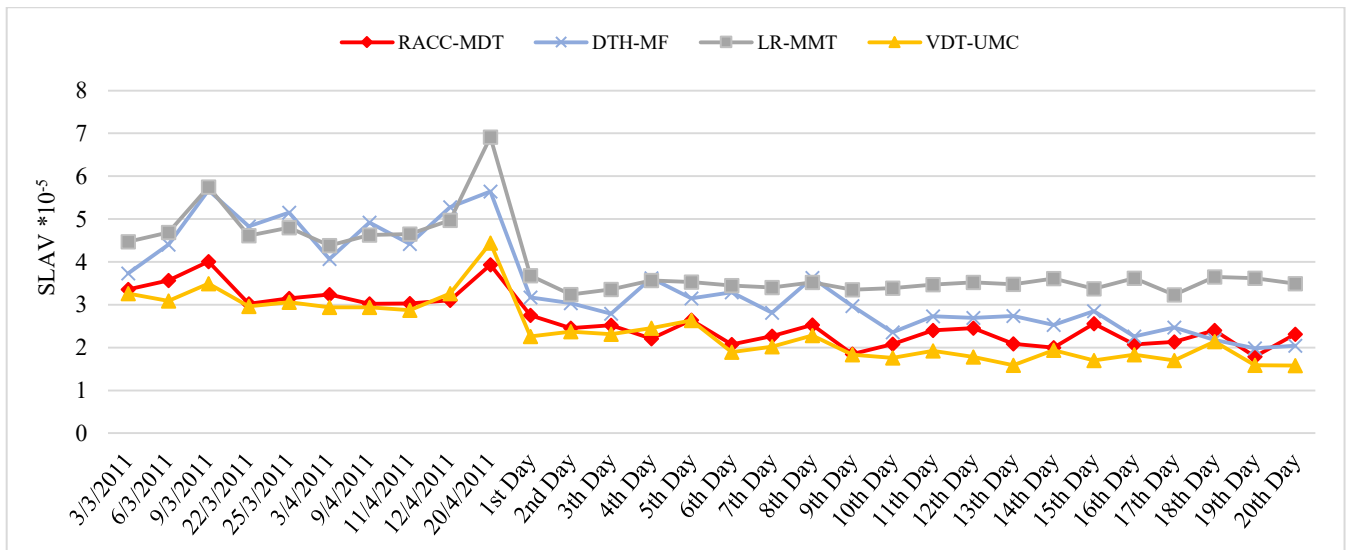


FIGURE 15. SLAV Violations of Different Approaches in Simulated Environment

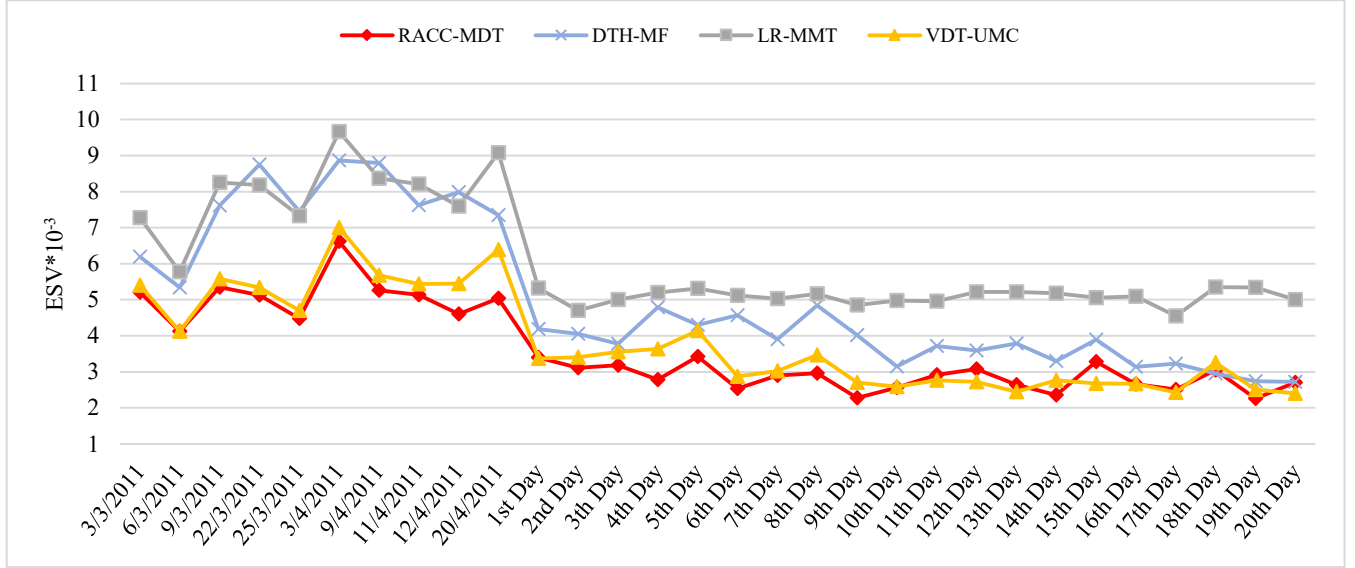


FIGURE 16. ESV of Different Approaches in Simulated Environment

The comparison results of the SLAV (Figure 15) are different from those in the real environment. It is found that, although our approach, the RACC-MDT, is better than the DTH-MF and LR-MMT, it is worse than the VDT-UMC. We attribute this finding to the excessively small resource utilization of each VM in the datasets (as shown in Table VIII, average CPU utilization is far below 15%); thus, the resource use of the data center is inefficient; furthermore, most of the VMs selected to be migrated in the VM selection phase are from underloaded hosts. The VDT-UMC takes the number of VMs into account when detecting underloaded hosts; therefore, it can effectively decrease the number of VMs migration in the consolidation process (this is evident from Figure 14), which consequently reduces the SLAV caused by VM migration. In the real environment, due to the small number of VMs and physical machines, the above characteristics are not obvious.

The comparison results based on the ESV metric (Figure 16) suggests that the proposed RACC-MDT has the least values in most of the conditions. Specifically, compared with the existing LR-MMT, DTH-MF, VDT-UMC, the average ESV of the RACC-MDT in 30 days decreased by 40.88%, 28.66%, and 6.14%, respectively.

TABLE IX
PAIRWISE COMPARISONS OF ESV VALUES BY USING PAIRED *T* TESTS

Approach 1 (ESV*10 ⁻³)	Approach 2 (ESV*10 ⁻³)	95% CI	<i>P</i> value
RACC-MDT (3.56)	LR-MMT (6.03)	(-3.0722, - 2.1789)	2.3*10 ⁻⁸
RACC-MDT (3.56)	DTH-MF(5)	(-2.7667, - 1.4456)	1.5*10 ⁻⁵
RACC-MDT (3.56)	VDT-UMC (3.8)	(-0.5902, - 0.1490)	3.3*10 ⁻⁴
LR-MMT (6.03)	DTH-MF (5)	(0.0892, - 0.9496)	2.1*10 ⁻⁴
LR-MMT (6.03)	VDT-UMC (3.8)	(1.9218, - 2.5902)	4.8*10 ⁻⁹
DTH-MF (5)	VDT-UMC (3.8)	(1.9218, - 2.5902)	5.5*10 ⁻⁵

Table IX shows the results based on the paired *t*-tests of the ESV values under the four approaches. The results show that there is a statistically significant difference between these approaches, and the proposed approach, RACC-MDT, best minimizes the ESV value.

In the geographically distributed cloud computing environment, due to the smaller bandwidth between different clusters, we focus on the communication overhead, including the amount of data transfer between different clusters and SLA violation during the VMs migration.

TABLE X
AMOUNT OF DATA TRANSFER BETWEEN DIFFERENT CLUSTERS

Data Flow Direction	Amount of Data Transfer (TB)			
	RACC-MDT	DTH-MF	LR-MMT	VDT-UMC
Cluster1→2	19.3	21.3	39.4	31.6
Cluster1→3	7.4	10.5	28.7	17.3
Cluster2→1	54.6	48.9	42.9	36.5
Cluster2→3	14.8	20.7	29.6	28.7
Cluster3→1	49.5	61.1	59.3	40.1
Cluster3→2	31.6	47.8	74.2	27.3
Total	177.2	216.3	274.1	181.5

The comparison results of the amount of data transfer between the different clusters are shown in Table X (Cluster *i* → *j* represents the direction of data transmission from cluster *i* to cluster *j*). It is observed that the RACC-MDT approach is manifestly better than the DTH-MF and LR-MMT, and slightly better than the VDT-UMC. More specifically, compared with the existing approaches, the amount of data transfer between the different clusters using the RACC-MDT approach is less by 18.1% (compared to the DTH-MF), 35.3% (compared to the LR-MMT), and 2.6% (compared to the VDT-UMC).

This may be attributed to the fact that our approach, the RACC-MDT, focuses on the total amount of data transfer when selecting the VMs from the overloaded host, and migrating the VMs to a host with higher PPR, which can decrease the frequency of live migration. The amount of data transfer results shown in Cluster1 → 2, Cluster1 → 3 and Cluster2 → 3 buttress the above analysis.

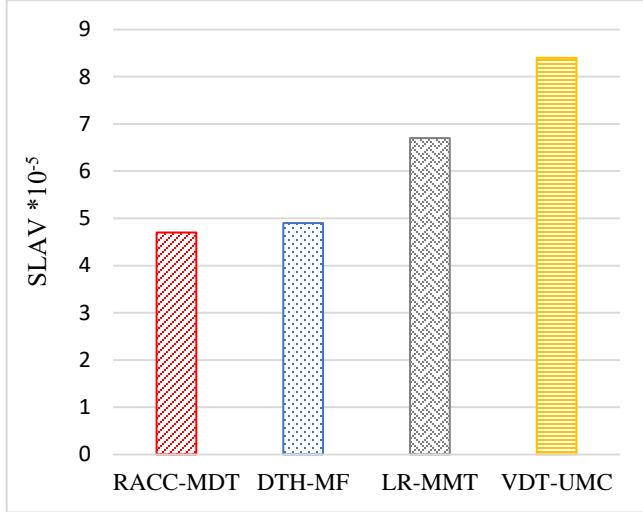


FIGURE 17. SLAV Comparison in Multi-cluster Simulation Environment

Figure 17 demonstrates the comparison results of the SLAV metric in the multi-cluster simulation environment. It suggests that the proposed RACC-MDT has the least values in most of the conditions. Compared with the existing DTH-MF, LR-MMT, and VDT-UMC, the average SLAV of the RACC-MDT in 30 days decreased by 5%, 24%, and 44%, respectively. This result is basically consistent with that in the single cluster environment.

It is noteworthy that we did not analyze energy consumption indicators in the multi-cluster environment. This is because the approach proposed in this paper is based on the inherent characteristics (i.e. PPR) of the hosts, allocating a VM to a host with the higher PPR while ensuring the least increase in energy consumption. The core idea behind this is that the high-PPR hosts consume less energy per operation, compared to the others. Therefore, it is independent of the number of servers in the data center, the geographical distribution, and the type and size of the task.

VII. CONCLUDING REMARKS AND FUTURE DIRECTIONS

This study presents a novel adaptive framework for VM consolidation based on the resource utilization and the consideration of the PPR of heterogeneous hosts to resolve the trade-off between the energy and performance in cloud data centers. The proposed approach can detect overloaded hosts based on the RACCT algorithm, select VMs based on the MDT algorithm, determine the underloaded hosts based on the MCUHD algorithm, and find the VM placement based on the modified algorithm of PABFD algorithm. We have used performance metrics including energy consumption, the

total amount of transmitted data, SLAV, and ESV to evaluate the efficiency of the proposed approach. We have compared our approach with the state-of-the-art existing approaches (LR-MMT, VDT-UMC, DTH-MF), and experimental evaluation has been conducted in both real and simulated environments. The results show that the RACC-MDT is reliable, scalable and can significantly reduce the energy consumption, compared to the existing approaches, while maintaining the SLA violations at a reasonable level.

This study has mainly investigated the general but critical problem of VM consolidation, namely, the trade-off between energy consumption and SLA violation. In a geographically distributed computing environment, the resource cost, SLA definition, and virtualization technology differ significantly across the different clusters. Therefore, the future work will be focused on improving the robustness of the proposed framework against resource consolidation constraints among different clusters.

ACKNOWLEDGMENT

This research was supported by the National Natural Science Foundation of China (Grant NO.61472139) and the Newton Fund Institutional Links under the Newton-Ungku Omar Fund partnership of UK (Grant ID. 332438911, The grant is funded by the UK Department of Business, Energy, and Industrial Strategy (BEIS) and the Malaysian Industry-Government Group for High Technology; it was delivered by the British Council. For further information, please visit www.newtonfund.ac.uk.)

The authors would like to thank the reviewers, who provided constructive comments on the earlier version of this paper.

REFERENCES

- [1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.
- [2] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, "1000 islands: an integrated approach to resource management for virtualized data centers," *Cluster Computing*, vol. 12, no. 1, pp. 45–57, 2008.
- [3] J. Koomey, "Growth in data center electricity use 2005 to 2010," A report by Analytical Press, completed at the request of The New York Times, Sep. 2011.
- [4] A. Ender, "Energy Efficiency in Data Centers," *IEEE ComSoc Technical Committees Newsletter*, Nov. 13, 2019.
- [5] A. Shehabi, S. J. Smith, E. Masanet, et al, "Data center growth in the United States: decoupling the demand for services from electricity use", *Environmental Research Letters*, vol. 13, no. 12, 2018.
- [6] L. A. Barroso, U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, 2007.
- [7] X. Fan, W. D. Weber, L. A. Barroso, "Power provisioning for a warehouse-sized computer," In: *ACM SIGARCH Computer Architecture News*, vol. 35, no. 2, pp. 13–23, 2007.
- [8] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, "Live migration of virtual machines," In: *Proc. of the 2nd USENIX Symp on Networked Systems Design & Implementation (NSDI 2005)*, Boston, MA, USENIX Association, Berkeley
- [9] A. Beloglazov, R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic

- consolidation of virtual machines in cloud data centers,” *Concurr Comput Pract Exp*, vol. 24, no. 13, pp. 1397–1420, 2012.
- [10] Standard Performance Evaluation Corporation. https://www.spec.org/power_ssj2008/results/.
- [11] E. Oikonomou, D. Panagiotou, A. Rouskas, “Energy-aware Management of Virtual Machines in Cloud Data Centers,” *Proceedings of the 16th International Conference on Engineering Applications of Neural Networks (INNS)*, ACM, pp. 1-6, 2015.
- [12] A. Beloglazov, J. Abawajy, 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.
- [13] W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, “Cost of virtual machine live migration in clouds: a performance evaluation,” In: *Cloud computing. Lecture notes in computer science*, vol. 59, no.31, pp. 254–265, 2009.
- [14] A. Horri, M. S. Mozafari, G. Dastghaibfard “Novel resource allocation algorithms to performance and energy efficiency in cloud computing,” *Journal of Supercomputing*, vol. 69, no. 3, pp. 1445–1461, 2014.
- [15] Chen Wei, Zhihua Hu, Yougan Wang, “Exact algorithms for energy-efficient virtual machine placement in data centers,” *Future Generation Computer Systems*, vol. 106, pp. 77–91, 2020.
- [16] H. Li, G. Zhu, C. Cui, H. Tang, Y. Dou, C. He, “Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing,” *Computing*, vol. 98, no. 3, pp. 303-317, 2016.
- [17] E. Ariyanan, H. Taheri, S. Sharifian, “Novel heuristics for consolidation of VMs in cloud data centers using multi-criteria resource management solutions,” *Journal of Supercomputing*, vol. 72, no. 2, pp. 688–717, 2016.
- [18] I. Hwang, M. Pedram, “Hierarchical, portfolio theory-based VM consolidation in a compute cloud,” *IEEE Transactions on Services Computing*, vol. 99, no.1, 2016.
- [19] S. Y. Z. Fard, M. R. Ahmadi, S. Adabi, “A dynamic VM consolidation technique for QoS and energy consumption in cloud environment,” *J Supercomput*, vol. 73, no. 10, pp. 4347–4368, 2017.
- [20] Neeraj Kumar Sharma, G. Ram Mohana Reddy, “Multi-Objective Energy Efficient Virtual Machines Allocation at the Cloud Data Center,” *IEEE Transactions on Services Computing*, vol. 12, no. 1, pp. 158–171, 2019.
- [21] Z. Li, X. Yu, L. Yu, et al, “Energy-efficient and quality-aware VM consolidation method,” *Future Generation Computer Systems*, vol. 102, pp.789–809, 2020.
- [22] L. Minas, B. Ellison, “Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers,” *Intel Press*, vol. 68, no. 3, pp. 154-158, 2009.
- [23] P. A. Dinda, “The statistical properties of host load,” *Languages, Compilers, and Run-Time Systems for Scalable Computers*, vol. 1511, no. 4, pp. 319-334, 1998.
- [24] Z. Xiao, W. Song, Q. Chen, “Dynamic resource allocation using virtual machines for cloud computing environment,” *IEEE Transactions on Parallel & Distributed Systems*, vol. 24, no. 6, pp. 1107–1117, 2013.
- [25] D. Yang, J. Cao, J. Fu, J. Wang, J. Guo, “A pattern fusion model for multi-step-ahead CPU load prediction,” *Journal of Systems and Software*, vol. 86, no. 5, pp. 1257–1266, 2013.
- [26] G. J. Eidleman, “Z scores-A Guide to failure prediction,” *The CPA Journal*, vol. 65, no. 2, pp. 52, 1995.
- [27] N. Sharma, R. M. Guddeti, “Multi-Objective Energy Efficient Virtual Machines Allocation at the Cloud Data Center,” *IEEE Transactions on Services Computing*, 2016.
- [28] F. Fahimeh, P. Tapio, L. Pasi, et al, “Energy-Aware VM Consolidation in Cloud Data Centers Using Utilization Prediction Model,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524–536, 2019.
- [29] B. Shuja-ur-Rehman, I. Waheed, L. B. Josep, C. David, “Adaptive sliding windows for improved estimation of data center resource utilization,” *Future Generation Computer Systems*, vol. 104, pp. 212–224, 2020.
- [30] Y. Gao, H. Guan, Z. Qi, Y. Hou, L. Liu, “A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,” *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230-1242, 2013
- [31] F. Farahnakian, A. Ashraf, T. Pahikkala, P. Liljeberg, J. Plosila, I. Porres, H. Tenhunen, “Using ant colony system to consolidate VMs for green cloud computing,” *IEEE Transactions on Services Computing* vol. 8, no. 2, pp. 187-198, 2015.
- [32] K. Jackson. “OpenStack cloud computing cookbook,” Ehu Es, 2013.
- [33] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. DeRose, R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and Experience*, vol. 41, no. 1, pp. 23-50, 2011.
- [34] S. A. Weil, S. A. Brandt, E. L. Miller, et al, “Ceph: A scalable, high-performance distributed file system,” *Proceedings of the 7th symposium on Operating systems design and implementation*. USENIX Association, pp. 307-320, 2006.
- [35] OSHI. <https://github.com/oshi/oshi>
- [36] M. A. Sharma, M. O. Joshi, “Openstack Ceilometer Data Analytics & Predictions,” *IEEE International Conference on Cloud Computing in Emerging Markets*, pp. 182-183, 2017.
- [37] StressLinux. <https://www.stresslinux.org/>
- [38] K. Park, V. S. Pai, “CoMon: a mostly-scalable monitoring system for PlanetLab,” *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65-74, 2006.
- [39] S.B. Shawa, et al, “Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in cloud data center,” *Computers and Electrical Engineering*, pp. 241-254, 2015.
- [40] Y. Chang, C. Gu, F. Luo, et al. “Energy Efficient Resource Selection and Allocation Strategy for Virtual Machine Consolidation in Cloud Datacenters,” *IEICE Transactions on Information and Systems*, pp. 1816-1827, 2018.
- [41] Martello S , Toth P . Knapsack problems: algorithms and computer implementations. John Wiley & Sons, Inc. 1990.
- [42] A. Beloglazov, R. Buyya, “Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers under Quality of Service Constraints,” *IEEE Transactions on Parallel & Distributed Systems*, vol. 24, no. 7, pp. 1366-1379, 2013.
- [43] Goole Cluster Workload traces. https://github.com/google/cluster-data/blob/master/ClusterData2011_2.md