A Combined Trend Virtual Machine Consolidation Strategy for Cloud Data Centers

Article *in* IEEE Transactions on Computers · January 2024

DOI: 10.1109/TC.2024.3416734

5 authors, including:

Zhen Zhang
Jinan University (Guangzhou, China)
44 PUBLICATIONS **255** CITATIONS

SEE PROFILE

Yuhui Deng
Jinan University (Guangzhou, China)
143 PUBLICATIONS **1,258** CITATIONS

SEE PROFILE

Geyong Min
University of Exeter
682 PUBLICATIONS **14,967** CITATIONS

SEE PROFILE

A Combined Trend Virtual Machine Consolidation Strategy for Cloud Data Centers

Yuxuan Chen, Zhen Zhang, Yuhui Deng, *Member, IEEE,* Geyong Min, *Member, IEEE,* and Lin Cui, *Member, IEEE,*

***Abstract*—Virtual machine (VM) consolidation strategies are widely used in cloud data centers (CDC) to optimize resource utilization and reduce total energy consumption. Although existing strategies consider current and future resource utilization, the impact of sudden bursts in historical resource utilization on the hosts has been underestimated in uncertain future periods. Insufficient analysis of historical resource utilization may increase the risk of host overloading and Service Level Agreement Violation (SLAV). By defining historical and future trends based on resource utilization, we propose a novel combined trend VM consolidation (CTVMC) strategy which can effectively reduce energy consumption and SLAV. The VMs with the largest combined trend are selected for migration to prevent host overloading. Based on the temporal locality and prediction technique, CTVMC then employs the past, present, and future resource utilization to filter candidate hosts, and identifies the most complementary host to place VM using combined trends. We conduct extensive simulation experiments with PlanetLab Trace and Google Cluster Trace in the CloudSim simulator. Compared with the well-known strategies, CTVMC strategy using the PlanetLab Trace can reduce the number of migrations by over 72.39%, SLAV by over 75.85%, and ESV (a combined metric that judges the trade-off between energy consumption and SLAV) by over 81.54%. According to the Google Cluster Trace, our strategy can reduce the number of migrations by over 61.51%, SLAV by over 37.37%, and ESV by over 35.30%.**

***Index Terms*—Cloud Data Center, Virtual Machine, Service Level Agreement, Energy Consumption, Temporal Locality**

I. Introduction

Cloud computing is a mode for users to access massive computing resources of cloud service providers on demand [1]. The cloud service providers, such as Amazon, Microsoft, and Google, had established cloud data centers around the world to provide low latency, scalable, and flexible services to cloud users on a pay-as-you-go model. Cloud users request physical resources in the form of VM instances to run different types of applications. To improve the resource utilization of physical machines (PMs) in cloud data centers and isolate applications of different users, virtualization technologies such as Xen and KVM, have been introduced to meet the growing demand for cloud services. With the rapid growth of cloud data centers, the operational costs of cloud service providers have become higher, and resource management has also become more complex.

The energy consumptions of cloud data centers are very high. By virtue of the model proposed in [2], the total energy consumption of data centers is expected to grow from 286 TWh to 321 TWh in 2030. This is not only because the number of active hosts is large, but also because the usage of physical resources is inefficient. After a six-month observation of a cloud data center, the average CPU utilization of more than 5000 hosts in the cloud data center was only 10% - 50% [3]. Therefore, reducing the significant increase in energy consumption of cloud data centers is critical.

To provide users with reliable cloud services, the cloud data center needs to maintain the performance of VMs. Therefore, a service level agreement (SLA) has been reached between cloud service providers and users [4]. The SLA consists of multiple system performance metrics, such as system throughput, response time, and downtime ratio [5]. By reducing the SLAV in the cloud data center, the VM performance can be maintained within an acceptable range. Due to the uncertainty of the number of resources requested by cloud users, the workloads on the VMs are highly dynamic, which will increase the risk of PM overloading. When the PM is overloaded, SLAV will occur, and the competition of VMs for physical resources will lead to significant performance degradation [6], [7]. To avoid SLAV, VM live migration technology [8] is widely used in the VM consolidation strategy to reallocate VMs from the overloaded PMs to normal PMs. However, the cost of VM migration is high, which will cause additional energy consumption and VM performance degradation. Therefore, the trade-off between energy consumption and system performance is unavoidable in VM consolidation strategies.

In VM consolidation, simultaneous optimizing energy consumption and SLAV is an NP problem. That means, it is difficult to find an optimal resource allocation strategy to optimize energy consumption and SLAV, simultaneously. As an NP problem, many consolidation strategies have tried to find better solutions to optimize the energy consumption and SLAV simultaneously [9]–[13]. Traditional VM consolidation strategies [14]–[16] are proposed to reduce the energy cost and optimize resource utilization as much as possible. However, such VM consolidation strategies usually lead to high SLAV, because the potential risk of host overloading in the future is ignored. In recent VM consolidation strategies [5], [17]–[19], the dynamic adaptability of the workload has been considered. By using the prediction technique, these strategies try to prevent the host from being overloaded again in the future

Y. Chen, Z. Zhang(corresponding author), Y. Deng(corresponding author), and L. Cui are with the College of Information Science and Technology, Jinan University, Guangzhou, China, 510632. E-mail: xychen@stu2020.jnu.edu.cn, zzhang@jnu.edu.cn, tyhdeng@jnu.edu.cn, tcuilin@jnu.edu.c.

G. Min is with the College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter EX4 4QF, United Kingdom. E-mail: g.min@exeter.ac.uk.

based on the current and future resource utilizations, thus reducing the SLAV.

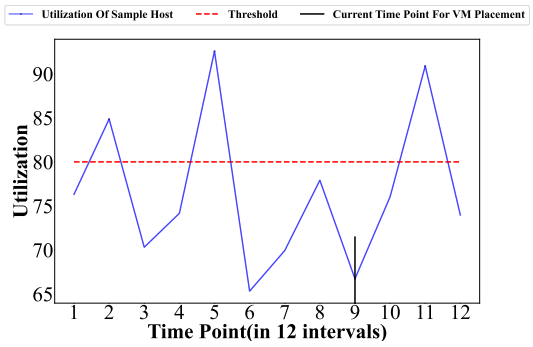


Fig. 1: Resource utilization pattern of sample host in PlanetLab Trace

Figure 1 illustrates an example of VM placement using current and future resource utilization on PlanetLab Trace [20]. The red dotted line indicates that a VM is placed at the current time point 9 and the three red dots represent the time point when the host is overloaded. It can be seen that the VM is placed because the host is not overloaded at the current time point 9 and the next time point 10. However, the host is overloaded at time point 11, which cannot be expected by the existing prediction-based strategies. Though such a random and discontinuous burst is difficult to predict, the signature of temporal locality can be detected from the workload. The host that is overloaded in the future time point 11 is also overloaded in the historical time points 2 and 5. According to the principle of temporal locality [21], if an instruction in a program was executed, it may be executed again soon. This principle applies to the hosts running in the cloud data centers, meaning that a host that has been overloaded in the near past may also be overloaded in the near future.

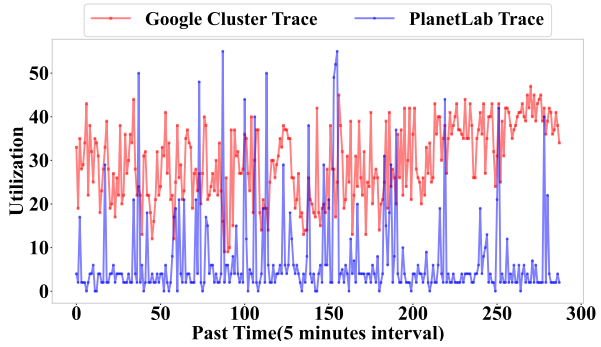


Fig. 2: The resource utilizations of the traces of PlanetLab and Google Cluster

By analyzing the traces of PlanetLab and Google Cluster [25], we sample the resource utilization for every 5 minutes. For simplicity, we only give a simple of the resource utilizations of one random selected host in one day as shown in Figure 2. We found that the resource utilizations of the hosts exhibit significant fluctuations. However, the sudden bursts in historical resource utilization are not effectively utilized in current VM consolidation strategies. Only predicting future resource utilization makes it difficult to capture random bursts in host workloads that may occur during a future period.

TABLE I: Temporal locality characteristics in VM consolidation strategies

Traces	VM consolidation strategies	M	N	$(N/M)(\%)$
PlanetLab	PABFD [22]	14368	51746	62.06%
	FFD [23]	21970	15105	68.75%
	UPV-MC [24]	15390	4978	32.35%
	PEAS [9]	16747	11449	68.36%
	PAVMP [11]	21835	14949	68.46%
	CTVMC	4919	928	18.87%
Google Cluster	PABFD [22]	32477	18358	56.53%
	FFD [23]	39582	24617	62.19%
	UPC-MC [24]	27405	8151	29.74%
	PEAS [9]	16101	9271	57.58%
	PAVMP [11]	41006	26099	63.65%
	CTVMC	8679	2226	25.65%

As shown in Table I, we stimulated six VM consolidation strategies with 800 PMs and 1600 VMs including FFD, PABFD, UPV-MC, PEAS, PAVMP, and our proposed CTVMC. The parameter M denotes the total number of overloaded in PMs. The parameter N denotes the number of overloaded after placing VMs into one PM on the condition that the PM is already overloaded in the near past. Thus, the parameter N/M represents the degree of temporal locality characteristics in resource utilization in the CDC. From the data of N/M, we can see that all the VM consolidation strategies present temporal locality characteristics. That means, a lot of hosts, which have been overloaded recently, may also be overloaded in the near future after placing VMs. Obviously,the parameters of M, N, and N/M in CTVMC are the smallest among these six consolidation strategies. That means, the CTVMC can greatly reduce the impact of sudden burst of the resource utilization by using the temporal locality characteristics.

Based on the above analysis, we propose CTVMC, a novel VM consolidation strategy that fully considers the impact of sudden bursts in historical resource utilization on the host. We divide the change value of host resource utilization by the number of VMs run on the PM to obtain the normalized value as the trend in resource utilization of PM. We propose the combined trend in VM selection and VM placement to keep hosts running stably in both the past, present and future. Following the idea of peak load shifting [11], [26], we place VMs on the target host that has the most complementary combined trend, effectively reducing the risk of host overloading.

The main contributions of this paper are as follows:

- To take full advantage of historical resource utilization, we introduce a new criterion of combined trend, which consists of the historical trend and future trend. Based on the combined trend, we propose a novel VM consolidation (CTVMC) strategy that focuses on VM selection and VM placement.
- An original VM selection policy is proposed to select the VM with the largest combined trend to migrate. This policy can eliminate the workload growth trend of the overloaded host and reduce the risk of the host being overloaded again.
- An innovative VM placement policy is developed to find

the most complementary host to VM in the combined trend. Based on the temporal locality and prediction technique, this policy uses the past, present, and future resource utilization to filter hosts. An Adaptive Window Size Selection algorithm is then proposed to select appropriate historical resource utilization to eliminate the expired bursts in workloads.

The remainder of this paper is organized as follows. Section 2 presents the related work on VM consolidation strategies. An overview of the problem formulation is provided in Section 3. The CTVMC strategy is proposed in Section 4. In Section 5, we compare our algorithm with the state-of-the-art algorithms through simulation experiments. Section 6 summarizes the work of this paper and discusses future research directions.

II. Related works

This section presents the VM consolidation strategies in cloud data centers, which can be classified into two categories: (1) Traditional VM consolidation strategies, and (2) Prediction-based VM consolidation strategies.

A. Traditional VM consolidation strategies

The traditional VM consolidation strategies only consider the current resource utilization of the host. Because the workloads of VMs are highly dynamic, the analysis of the workload trends in these strategies is inadequate, and unnecessary migrations may occur. Thus, the overhead, such as additional energy consumption and performance degradation, will be aggravating.

Beloglazov et al. [16] proposed a classical VM placement policy, named PABFD. In PABFD, the VMs are first sorted by CPU utilization and then placed on the host with the smallest energy consumption difference by comparing the difference in host energy consumption generated before and after the VM placement. They also proposed several statistical host overload detection algorithms, such as median absolute deviation (MAD), interquartile range (IQR), and Minimization of Migrations (MM). Keller et al. [23] proposed a kind of First Fit-based (FFD) VM consolidation strategy , which tries to allocate a VM to the first host from the host list that can provide enough resources. The main idea of these strategies is to migrate VMs from overloaded and underloaded hosts, then shut down the idle hosts to reduce energy consumption. Murtazaev et al. [14] proposed a VM consolidation strategy named Sercon, which can continuously migrate VMs from the minimum-load hosts to the maximum-load hosts to release the minimum-load hosts as far as possible. Takouna et al. [27] proposed a robust VM Consolidation strategy that consists of three algorithms: robust host overload detection algorithm, VM selection policy based on minimum signal-to-noise ratio (SNR), and modified best-fit VM placement policy. They also proposed an adaptive historical window selection policy, named FSHW, to select a historical window. Laili et al. [12] proposed an iterative budget algorithm with multi-stage selection to find the migrated VMs and the target hosts. Furthermore, a comprehensive model of the migration cost, energy consumption, and overall communication overhead was proposed to find the best decision for VM adjustment. Lin et al. [9] proposed a Peak Efficiency Aware Scheduling (PEAS) strategy, which consists of PEAP and PEACR, to achieve dual improvement in performance and energy consumption in server clusters. As a VM placement policy, PEAP can find the target host which can maintain the best peak power efficiency after placing the selected VM. PEACR is a VM selection policy that can consider the VM resource utilization, VM memory size, and available bandwidth to select a VM.

All of these strategies have the same drawbacks. When a short-term burst of workload occurred in a host, VM consolidation only using the current resource utilization of the host may lead to many unnecessary VM migrations. Since the workloads of VMs are highly dynamic, some hosts, which currently appear not to be overloaded after placing VMs, may be overloaded again in the future. Once these happen, it may result in additional VM migration and energy consumption.

B. Prediction-based VM consolidation strategies

In the Prediction-based VM consolidation strategies, the time-series forecasting analysis is used to obtain the future resource utilization of the host. It can judge whether the host will be overloaded in the future and avoid unnecessary VM migrations and SLAVs. However, these strategies only rely on the current and future resource utilization to determine whether the host is overloaded. The impact of historical resource utilization exceeding host thresholds is underestimated.

Farahnakian et al. [24] proposed Utilization Prediction-aware VM Consolidation(UP-VMC), which used a regression-based model to obtain the future resource utilization of VMs and PMs. In UP-VMC, the utilization of CPU and memory in the host were combined in the form of a multi-dimensional packing problem, and the SLAVs can be effectively prevented. Similar to UP-VMC, Hieu et al. [19] also adopted a regression-based model to estimate resource utilization. They proposed the PABFD-MUP algorithm for VM placement, which selects the stable host with the least energy increase in the future. Ahmadi et al. [28] combined linear regression with interval estimate to propose the concept of the VM safe zone. In VM safe zone, the future resource utilization of the VM has an upper limit and a lower limit. It can avoid performance degradation by ensuring that the VMs are placed in a safe zone. Zhao et al. [29] built a VM performance model to predict the VM performance after migrating VMs. An ant colony optimization based (ACO-based) algorithm is also proposed to maximize VM performance and minimize the number of active PMs and the total migration cost, simultaneously.

As a classical time-series forecasting method, ARIMA is adopted by many studies to predict the future resource utilization. By using ARIMA, Fu et al. [11] proposed a policy called PAVMP. In PAVMP, the affinity between the VMs is detected to find the host with the highest affinity to place the selected VM. Through finding the ascending/descending trend of the Google Cluster Trace, Zhou et al. [10] proposed a strategy, named DADTA, which is predicted by using the ARIMA. In DADTA, the thresholds of the hosts can be adaptively adjusted to detect overloaded and underloaded hosts.

As a simple and effective tool, the Markov chain model is widely used in many studies [30], [31] to accurately predict future resource utilization. Hsieh et al. [5] proposed a host overload/underload detection algorithm, which combines the Markov chain model with the grey prediction model to reduce the prediction error. Ranjbari et al. [18] used learning automata to consider the change in VM resource utilization, so as to find out the hosts that may be overloaded.

How to ensure service level agreements provided by cloud manufacturing service providers while reducing host energy consumption is still a challenging problem. Many works try to find trade-off between the energy consumpting and SLAs [32]–[36]. In recent years, the reinforcement learning algorithm is adopted to improve the accuracy of prediction [37]–[40]. However, these strategies cannot be applied to the real cloud data center with highly dynamic workloads, because these strategies either need time-consuming offline training or set many specific parameters for the dataset.

For highly dynamic VM workloads, the predicted values got by the VM consolidation strategies analyzed above may be not correct. The incorrect predictions may lead to wrong judgments during the VM placement process, resulting in unnecessary migration. Additionally, the historical resource utilizations of hosts are underutilized. The far-reaching impact of sudden bursts in historical resource utilization on hosts is downplayed.

III. Problem Formulation

In this section, we first present our system architecture of the cloud data center. Then we provide the relevant definitions of the PMs and VMs. We also define the combined trend of the resource requested by users, which is the key concept in the CTVMC strategy. For clarity, we show the symbols used in this paper in Table II.

TABLE II: Symbol List

Symbol	Definition
M	Number of VMs
N	Number of PMs
V	Set of VMs in the cloud data center
H	Set of PMs in the cloud data center
E	Total energy consumption of cloud data center
t	Current system time in cloud data center
$V_i^C(t)$	Resource capacity of V_i at time t
$V_i^R(t)$	Used resource of V_i at time t
$V_i^U(t)$	Resource utilization of V_i at time t
$H_i^C(t)$	Resource capacity of H_i at time t
$H_i^R(t)$	Used resource of H_i at time t
$H_i^U(t)$	Resource utilization of H_i at time t
U_{H_i}	Resource utilization list of H_i
$maxSize$	Number of resource utilization collected by local manager
P_{H_i}	Number of resource changes of H_i in the past
F_{H_i}	Number of resource changes of H_i in the future
P_{V_i}	Number of resource changes of V_i in the past
F_{V_i}	Number of resource changes of V_i in the future
H_i^T	Combined trend vector after normalization of H_i
V_i^T	Combined trend vector of V_i
$P_{H_i}^{power}$	Energy that H_i consumes at time t

A. System Architecture

A typical system architecture of cloud data center is shown in Figure 3. To effectively manage the VMs and PMs in the cloud data center, we need three essential components, including a local manager, a global manager, and a VM monitor. The local manager continuously monitors the number of resources requested by users and collects the historical resource utilization in the host. The global manager gathers information from local managers to collect the overall situations of PMs and makes VM migration decisions according to the VM consolidation strategy based on the resource utilization of the past, current, and future. The VM monitor is responsible for receiving commands from the global manager and performing the VM migrations.

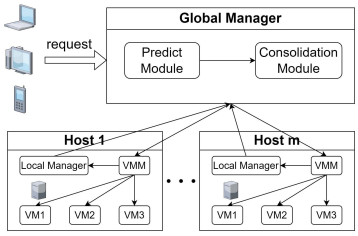


Fig. 3: The System Architecture of cloud data center

B. List of Hosts and VMs

The system is a large-scale cloud data center composed of N heterogeneous PMs, where each PM corresponds to a host. The set of hosts is denoted by $H =< H_1, H_2, \ldots, H_N >$, and H_i represents the i-th host where $i \in [1, N]$. Each host contains different types of resources, such as CPU, memory, network bandwidth, disk size, etc. Closely related to energy consumption, the CPU is the main focus we considered in this paper. In addition, to facilitate measurement, CPU is usually defined as million instructions per second (MIPS). The resource capacity set of hosts is defined as $H^C =< H_1^C, H_2^C, \ldots, H_N^C >$, which are fixed in the system. The number of resources consumed by the hosts at any time t can be defined as a resource usage set $H^R(t) =< H_1^R(t), H_2^R(t), \ldots, H_N^R(t) >$, which is the total resource usage of all running VMs in the PMs. Then the resource utilization of $H_i^U(t)$ at time t is defined as the used number of resources divided by the resource capacity of the considered host in Eq. (1). The historical resource utilization list of H_i is defined as $U_{H_i} = H_i^U(t-1), H_i^U(t-2), \ldots, H_i^U(t-maxSize)$, where $maxSize$ is the number of historical resource utilization collected by local manager.

$$H_i^U(t) = \frac{H_i^R(t)}{H_i^C}. \tag{1}$$

To make full use of the physical resources on the host, each host runs multiple VMs at any time to provide cloud services for different users. The set of VMs is denoted as $V =< V_1, V_2, \ldots, V_M >$, and V_i represents the i-th VM where $i \in [1, M]$. The resource capacity set of the VMs is defined as $V^C =< V_1^C, V_2^C, \ldots, V_M^C >$, and the used resource quantity set at time t is defined as $V^R(t) =< V_1^R(t), V_2^R(t), \ldots, V_M^R(t) >$. The resource utilization of V_i^U at time t is defined in Eq. (2).

$$V_i^U(t) = \frac{V_i^R(t)}{V_i^C}. \tag{2}$$

C. Predict Method

In the proposed strategy, we use the predicted value of resource utilization to calculate the future trend, so we need to find efficient and accurate time series analysis techniques to obtain the future resource utilization of the host.

In CTVMC, we adopt statistical methods to analyze the workloads in real CDCs and discover the temporal locality characteristics in the workloads. The Autoregressive Integrated Moving Average (ARIMA) model is one of linear models for time series analysis which is also constructed based on statistical methods. Compared with Bayesian and Neural networks, the ARIMA model is simple and lightweight, which makes ARIMA be an effective tool to predict the workloads in CDCs [10] [11] [41]. The ARIMA model is composed of the AR model and the MA model, and can only be applied to the time series with significant up-down trends or periodicity. Moreover, if the historical resource utilization is not smooth, we first make it stationary through the i-order difference.

The ARIMA model used in this article is constructed according to a well-known open-source third-party library component from GitHub (https://github.com/signaflo/java-timeseries.git). Two parameters of p and q are contained in the ARIMA model, where p is the number of autoregressive terms, and q is the number of lagged prediction errors in the prediction equation. By using resource utilizations as input, the ARIMA model arranges and combines different values of p, q, and d. The upper bounds of p, q, and d, are 5, 5, and 2, respectively. We find the values of p, q, and d which can minimize Akaike Information Criterion (AIC) through iterative calculation. The final ARIMA model is constructed based on the minimum AIC.

Referring to the state-of-the-art VM consolidation research works [10] [18] [37] [39], we adapt root mean square error (RMSE), mean absolute error (MAE), and Theil to compare the proposed CTVMC with other VM consolidation methods. These three metrics are defined as follows:

$$Theil = \frac{\left[T^{-1}\sum_{i=1}^{T}(R_i - P_i)^2\right]^{\frac{1}{2}}}{\left[T^{-1}\sum_{i=1}^{T}(R_i)^2\right]^{\frac{1}{2}} + \left[T^{-1}\sum_{i=1}^{T}(P_i)^2\right]^{\frac{1}{2}}} \tag{3}$$

$$RMSE = \left[T^{-1}\sum_{i=1}^{T}(R_i - P_i)^2\right]^{\frac{1}{2}} \tag{4}$$

$$MAE = T^{-1}\sum_{i=1}^{T}|R_i - P_i| \tag{5}$$

where R_i and P_i are the true and prediction value of VM resource at time i, and T is the total number of samples

The Theil coefficient is one of the commonly used statistical formulas to measure prediction accuracy [42]. The small value of Theil coefficient indicates good prediction, otherwise it indicates poor prediction. The Theil coefficient is highly sensitive to resource allocation efficiency, making it an ideal analytical tool for workload prediction and VM placement in cloud data centers. The RMSE and MAE are both used to measure the average size of the error. There still remains enduring confusion over their use. Neither of these two metrics is inherently better. RMSE is optimal for Gaussian errors, and MAE is optimal for Laplacian errors.

Notably, we focus on the use of the time series prediction method to calculate future trends, which will be described in the next section. Therefore, it is completely feasible to use other methods with higher accuracy in our strategy. As shown in Table III, ARIMA achieved excellent prediction results in both traces of PlanetLab and Google Cluster, which is suitable for our strategy. Furthermore, we also calculated the average predicted value and provide the average confidence bound (CB) of 95% for each workload trace as shown in Table IV and Table V. From Table IV and Table V, we can see that the CB is small enough to indicate that the predicted value is accurate. Since the predcitive algorithm is accurate enough, the scheduler in CTVMC can directly consolidate the VMs according the predictive algorithm.

TABLE III: The value of three metrics in two different traces

Workload Type	Root Mean Square Error	Mean Absolute Error	Theil
PlanetLab Trace	0.0826	0.0587	0.3919
Google Cluster Trace	0.0553	0.0353	0.3496

D. Definition of Combined Trend

Resource utilization differences between past, present and future hosts can reflect the changing characteristics of resources requested by hosts over time. In this section, we propose the definition of combined trend, which consists of historical trend and future trend. The actual increase or decrease of the resources used by the host in the past can be represented by the historical trend. The requested resources that may be changed by the host in the future can be represented by the future trend. By combining these two trends, we can comprehensively capture the changing characteristics of the workload and make more accurate decisions to stabilize the host. The historical trend and future trend of the H_i can be calculated in Eq. (6) and Eq. (7).

$$P_{H_i} = (H_i^U(t) - H_i^U(t-1)) \times H_i^C; \tag{6}$$

$$F_{H_i} = (H_i^U(t+1) - H_i^U(t)) \times H_i^C; \tag{7}$$

where $H_i^U(t+1)$ can be predicted by using ARIMA and other prediction techniques, $H_i^U(t-1)$ and $H_i^U(t)$ are historical resource utilization that can be obtained directly from the local manager. P_{H_i} is the historical trend which represents the number of resource changes of H_i in the past. F_{H_i} is the future trend which represents the number of resource changes of H_i in the future.

As shown in Eq. (8) and Eq. (9), the definitions of the combined trend of VM are similar to that of the host, except that the configuration of the host is replaced by that of VM.

TABLE IV: The CI of 95% for Plannetlab Trace

Workload	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
Truth value	12.31	11.44	10.70	9.26	10.56	12.39	11.12	11.56	11.54	10.43
Predict value	12.47	11.72	10.76	9.42	10.70	12.64	11.26	11.77	11.71	10.57
CB	(11.18, 13.76)	(10.41, 13.03)	(9.70, 11.83)	(8.68, 10.16)	(9.69, 11.70)	(11.60, 13.68)	(10.36, 12.16)	(10.80, 12.73)	(10.63, 12.78)	(9.59, 11.54)

TABLE V: The CI of 95% for Google Cluster Trace

Workload	G1	G2	G3
Truth value	5.57	11.22	10.76
Predict value	5.70	11.24	10.91
CB	(5.42, 5.99)	(11.22, 11.25)	(10.72, 11.10)

$$P_{V_i} = (V_i^U(t) - V_i^U(t-1)) \times V_i^C. \tag{8}$$

$$F_{V_i} = (V_i^U(t+1) - V_i^U(t)) \times V_i^C. \tag{9}$$

IV. The Proposed Strategy

A typical VM consolidation strategy can be divided into four steps, including host overload detection, host underload detection, VM selection, and VM placement [22]. The main work of the first two steps is to set an appropriate overload threshold and underload threshold for the host. In this paper, we mainly consider the last two steps. By combining historical trend and future trend, we proposed a novel combined trend VM selection policy and VM placement policy, which can eliminate the fluctuation of host resource utilization to the greatest extent over some time and make the host stable. To help readers understand more details of CTVMC, we uploaded the program source code to GitHub[1].

A. VM Selection Policy

During the VM selection process, it is important to select a suitable VM from the host for migration. Because VM migrations are inherently expensive, this results in performance degradation and additional energy consumption. Most existing policies only focus on short-term performance improvement, ignoring whether the overall utilization trend of the VM will have a negative impact on the source host and target host. For example, if the resource utilization of the source host keeps increasing and we select the VM with the shortest migration time as the MMT policy. The source host will likely be overloaded again in the future.

To solve this drawback, we proposed the VM selection policy to select the VM with the largest sum of P_{V_i} and F_{V_i}. If a host is overloaded, that means some VMs in the host are requesting too many resources at once. The VM that requested the most resources from the past to the present has the greatest impact on the host. To minimize the possibility of the host being overloaded again and wipe the growing workload from the overloaded host, we should migrate the VM which requests the largest number of the resource in both the past and future. The pseudo-code of the *VM selection policy* is shown in Algorithm 1. First, we need to traverse every overloaded host, which is detected by the host overload detection policy (line 2). Then, we iterate over each VM in the host and select the VM with the largest combined trend to migrate until the host is no longer overloaded (lines 3-22). After selecting a VM, the host will be checked whether it is still overloaded according to the host overload detection policy to exit the loop (line 19).

Algorithm 1 getMigratedVmList: VM Selection Policy

Input: hostList: The overloaded host list
Output: migratedVmList: The VMs needed to be migrated

```
1:  migratedVmList ← []
2:  for each H_i ∈ hostList do
3:    while true do
4:      vmList ← getMigratableVms(H_i);
5:      selectedVM ← NULL /*VM initially selected for migration is empty*/
6:      MaxTrend ← 0 /*largest combined trend is initialized to 0*/;
7:      for each V_j ∈ vmList do
8:        P_Vj = V_j.pastTrend(); /*calculated in Eq. (8)*/;
9:        F_Vj = V_j.futureTrend(); /*calculated in Eq. (9)*/;
10:       if P_Vj + F_Vi ≥ MaxTrend then
11:         MaxTrend = P_Vj + F_Vi
12:         selectedVM = V_j /*record the VM with the largest combined trend*/
13:       end if
14:     end for
15:     if selectedVM == NULL then
16:       break
17:     end if
18:     migratedVmList.add(selectedVM)
19:     if isHostOverloaded(H_i) == FLASE then
20:       break
21:     end if
22:   end while
23: end for
24: return migratedVmList
```

B. VM Placement Policy

The VM technology is one of the main technologies for resource allocation in CDC, which can greatly improve the efficiency of resource utilization in the host. The VM technology is adapted to configure and manage the resource in CDC. By deploying and migrating VMs, the CDC can provide better service for the users. Therefore, most state-of-the-art VM consolidation strategies consider the number of VMs to a

[1] https://github.com/dscLabJNU/cloudsim_ctvmc

host rather than their total resources requirement [5] [10] [18] [37] [39].

In CTVMC, we consider the number of VMs in each PM, and then use the number of VMs and the changes of PM resource utilization to obtain the trend of the PM resource utilization. In the Algorithm 1, Algorithm 2, and Algorithm 4 of the subsequent sections, the PM resource utilization trend is used to choose the migrated VM and the target PM. In contrast to the existing methods, this method can make these three algorithms more concise, and reduce computational complexity of these algorithms.

1) Definition of Similarity

Before we introduce our policy, we need to propose a metric, called *Similarity*, which can be used to find a host with a combined trend that is complementary to VM. According to the idea of shaving peaks and filling valleys, we make all peaks of VM resource utilization correspond to all troughs of host resource utilization as much as possible. By combining the VM and the host in this way, a relatively smooth host resource utilization fluctuation curve can be obtained. If the *Similarity* calculated by the combined trend is small, it means that the host will be more stable in the past, present, and future after the VM is put into it. The combined trend vector for V_j is defined in Eq. (10).

$$\vec{V_j^T} = [P_{V_j}, F_{V_j}] \tag{10}$$

The combined trend vector for H_i is defined in Eq. (11).

$$\vec{H_i^T} = [P_{H_j}/vmList.size, F_{H_j}/vmList.size] \tag{11}$$

where the *vmList.size* is the number of VMs running in H_i. Because each host runs multiple VMs at the same time, the combined trend of the host is usually much larger than that of the VM. We abstract the combined trend of the host into a unique combined trend VM through normalization to facilitate matching with candidate VMs.

The *Similarity* value S_{ij} is the cosine similarity of H_i^T and V_i^T which is defined in Eq. (12)

$$S_{ij} = \frac{\vec{H_i^T} \cdot \vec{V_j^T}}{||\vec{H_i^T}|| \cdot ||\vec{V_j^T}||} \tag{12}$$

2) Host Overload Detection Inside Placement

In the VM placement policy, an important step is to determine whether the host will be overloaded after the VM is placed. As mentioned in the introduction, only using the current and future resource utilization of the host to make a decision can reduce the risk of host overloading, but incorrect predictions will also lead to unnecessary VM migrations. Additionally, according to our analysis based on temporal locality, if a host is overloaded in the past, it is highly likely to be overloaded again in the future. Therefore, the impact of historical resource utilization on the host should also be fully analyzed to reduce the number of VM migrations and avoid SLAV as much as possible. To ensure continuous and stable operation, an ideal host should not be overloaded in the past, present, or future after placing a VM.

The pseudo-code of the *Host Overload Detection Inside Placement* is shown in Algorithm 2. The input of Algorithm 2 is the candidate host, which we check whether it has enough resources to place the VM (line 2). We calculated the *windowSize* in line 3 by using the Algorithm 3 introduced in the next section. In lines 6-11, we judge whether the host has been overloaded in the past. In lines 12-14, we judge whether the host is experiencing overloaded right now. In lines 15-17, we judge whether the host will be overloaded in the future by using the ARIMA prediction. Only if the host is not overloaded in the past, present, and future, do we consider the host to be in a stable state in line 15. It is worth noting that in line 18, we need to delete the temporary VM from the host. Because the actual VM reallocation is implemented by the VMM, here we only judge whether the host is suitable for the VM by pre-allocation.

Algorithm 2 isHostOverloadedInsidePlacement

Input: V_j: the selected VM
H_i: the candidate host
T: the threshold of the candidate host
Output: isHostOverloaded

```
1: isHostOverloaded = False
2: H_i.add(V_j) /* add a temp VM in vmList */
3: windowSize ← adaptiveWindowSizeSelection(host) /* Adaptive window size calculated in Algorithm. 3*/;
4: U_{H_i} ← H_i.getUtilizationHistory() /* resource utilization list of H_i */
5: predict H_i^U(t+1) /* using ARIMA prediction*/;
6: for j = 1 to windowSize do
7:    if H_i^U(t-j) > T then
8:       isHostOverloaded = True
9:       break
10:   end if
11: end for
12: if H_i^U(t) > T then
13:   isHostOverloaded = True
14: end if
15: if H_i^U(t+1) > T then
16:   isHostOverloaded = True
17: end if
18: H_i.remove(V_j) /* remove the temp VM*/
19: return isHostOverloaded
```

3) Adaptive Window Size Selection Algorithm

In our algorithm, we consider that the host cannot be overloaded in the past. So the window size for the historical utilization is the key factor for the Algorithm 2. Most studies that use historical resource utilization do not specify the window size. In the widely used Cloudsim simulator, the window size is set to 30 by default. But the workloads in the hosts are highly dynamic, it is inappropriate to directly use the fixed value of 30. By analyzing the resource utilization of the hosts running with the PlanetLab Trace, we present an extreme situation in Figure 4 that may lead to wrong judgment. Let the dashed line represent the current threshold of the host and the red line represent a sample host where the extreme case occurred. We can find that sample host 1 has been running

steadily at the past 29 time slots but overloaded in the 30th timeslot.

The workloads in the hosts are highly dynamic. Based on the principle of temporal characteristics, we only focus on resource utilizations of a host which have briefly exploded in the recent period, and ignore the outdated bursts of resource utilizations which no longer affect the running status of the host. However, the existing distance metrics cannot capture the recent fluctuations in the nearest resource utilization. Therefore, we propose a novel distance standard strategy, named *Adaptive Window Size Selection algorithm* (AWSS), which can find the optimal historical data interval.

In Algorithm. 2, the distance from the position of the peak point to the initial position is used as the window size of the resource utilization data we select. Then the Adaptive Window Size Selection algorithm will traverse the historical resource utilization of the host within this window size, especially the resource utilization at the peak point. As long as all resource utilization does not exceed the threshold, the host is determined not to be overloaded in the past. As shown in Figure 4, the blue line also displays the resource utilization of a typical host running with the PlanetLab Trace in the system. The host resource utilization reaches its peak in the sixth time slot, which is lower than the threshold. We can assume that this host will not be overloaded in the future.

The pseudo-code of the *Adaptive Window Size Selection* is shown in Algorithm 3. In lines 6-12, we constantly update the value of the *maxAvgSum*, which is calculated by accumulating the current resource utilization and the maximum average value is selected. This is an efficient way to process the resource utilization bursts in the near past. Because even after taking the average value, the position of the peak resource utilization can still be retained due to the small step. However, for resource utilization bursts in workloads that have occurred for a long time, the *maxAvgSum* calculated by the Adaptive Window Size Selection algorithm becomes smaller as the step size becomes larger, thus successfully weakening the impact of expired bursts. In line 13, we return the best window size, which represents the subscript of the recent peak resource utilization after measurement.

In predictive analysis, concept drift refers to the phenomenon where the statistical properties of the target variable change in an unpredictable manner over time. This will lead to a decrease in the prediction accuracy of the model. The adaptive window size selection algorithm attempts to find the peak position in the host's historical resource utilization data, where the time of resource utilization burst is relatively close to the current time. Since the adaptive window size selection algorithm chooses the nearest resource utilization burst of the host, it can greatly avoid the occurrence of concept drift.

4) The Process Of VM Placement Policy

As shown in the Algorithm 4, the running process of VM placement policy is the combination of the above metrics and policies. After selecting multiple VMs to migrate from the hosts through Algorithm 1, we need to reallocate them to a suitable host without SLAV. In lines 4-7, we will detect whether the host has enough resources to place the VM, then call the Algorithm 2 to check whether the host will be overloaded in the past, current, or future. In lines 8-15, we choose the host which has the least S_{ij} as the target host for VM by using the combined trend of host and VM. By making the combined trend of the VM and the host complementary, we can stabilize the workload of the host after the VM placement, thereby reducing the risk of SLAV.

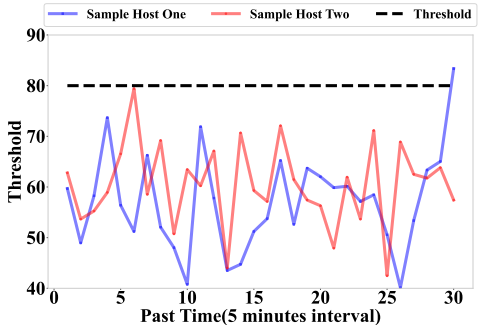


Fig. 4: Historical resource utilization of sample hosts

Algorithm 3 adaptiveWindowSizeSelection

Input: H_i: the candidate host
Output: windowSize: the window size for historical resource utilization

```
1:  windowSize = 0
2:  sum = 0 /* the sum of resource utilization */
3:  maxAvgSum = 0 /* dynamic average value of resource utilization */
4:  U_{H_i} ← H_i.getUtilizationHistory()
5:  maxSize = U_{H_i}.size /* the size of the whole resource utilization list */
6:  for j = 1 to maxSize do
7:      sum += H_i^U(t − j)
8:      if (sum / i) > maxAvgSum then
9:          maxAvgSum = sum / i
10:         windowSize = i
11:     end if
12: end for
13: return windowSize
```

5) Complexity analysis

The CTVMC is composed of four principal algorithms including getMigratedVmList (Algorithm 1), isHostOverloadedInsidePlacement (Algorithm 2), adaptiveWindowSizeSelection (Algorithm 3), and getTargetHost (Algorithm 4).

In Algorithm 1, the main operation is to traverse every overloaded host. Each VM in every host is iterated and select the VM with the largest combined trend is selected to migrate until the host is no longer overloaded (Lines 3-22). So, it can be deduced that the time complexity of Algorithm 1 can be equal to O($N * M$), where N is the number of host, and M is the largest number of VMs in the host list.

The Algorithm 2 detects the overloaded host. The main operation of Algorithm 2 is to judge whether the host has been overloaded in the past (Lines 6-11). Additionally, the prediction algorithm ARIMA is called by Algorithm 2, and the time complexity of ARIMA is O($p*N+q*N+d*N$) where p, q, and d are all small constants, and N is $windowSize$

TABLE VI: Power consumption of PMs

PM Type	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant G5	93.7	97	101	105	110	116	121	125	129	133	135
Express5800/GT110f-S	15.9	20.3	22.4	24.4	27.2	29.8	33.0	36.8	39.5	42.6	45.1
IBM System x3450	130	143	155	166	177	190	202	215	228	241	252

Algorithm 4 getTargetHost: VM Placement Policy

Input: V_j: the seleceted VM
hostList: the candidate hostList that VM can be placed
Output: targetHost

```
1: targetHost ← NULL /*host initially selected for placement is empty*/
2: minSimilarity = MAX_VALUE /*initialized to the maximum value to facilitate comparison*/
3: for each H_i ∈ hostList do
4:   if H_i isSuitableForVm(V_j) then
5:     if isHostOverloadedInsidePlacement(H_i) then
6:       continue
7:     end if
8:     V_j^T ← combined trend vector of V_j
9:     H_i^T ← combined trend vector of H_i
10:    S_ij ← calculated in (12) using V_j^T and H_i^T
11:    if S_ij <= minSimilar then
12:      minSimilar ← S_ij
13:      targetHost ← H_i /*record the host with the smallest similarity*/
14:    end if
15:  end if
16: end for
17: return targetHost
```

which is calculated by Algorithm 3. So, the time complexity of Algorithm 2 is equal to O($windowSize$).

The main operation of Algorithm 3 is to select appropriate historical resource utilization to eliminate the expired bursts in workloads (Lines 6-12). Then, the time complexity of Algorithm 3 can be equal to O($maxSize$), where $maxSize$ is the size of the whole resource utilizations list of host H_i which can be collected by local manager.

The Algorithm 4 reallocated the VMs, which are selected by Algorithm 1, to a suitable host. Since the Algorithm 4 should traverse all host and call the Algorithm 2, the time complexity of Algorithm 4 is O($N * windowSize$).

V. Performance Evaluation

In this seciton, we compared CTVMC with the other well-known VM consolidation strategies as follows: FFD [23], PABFD [22], PAVMP [11], UP-VMC [24], and PEAS [9], which have been introduced in the Related Works. The metrics of energy consumption, number of VM migrations, SLAV, and ESV are fully analyzed in this section.

A. Experimental Setup

1) Simulation environment

We selected CloudSim [43] as the simulator. As the mainstream cloud computing simulation framework, CloudSim and its derivatives has been adopted in many studies [37], [39], [44]. The CloudSim is adapted in this paper to simulate cloud data centers with heterogeneous PMs. These PMs can be divided into four types, and their configurations are shown in Table VII.

We also have implemented eight types of VMs, all of which are selected from Amazon EC2 VM instances[2]. The configurations of these VMs are shown in Table VIII. The numbers of PMs and VMs in the PlanetLab trace are fixed at 800 and 1600, respectively. In order to verify the scalability of CTVMC strategy, we construct three kinds of cloud data centers according to in Google Cluster trace which contain different numbers of heterogeneous PMs and VMs including: 800PMs+1600VMs, 2000PMs+4000VMs, and 4000PMs+8000VMs.

TABLE VII: Heterogeneous PM Models

Model	CPU(MIPS)	Core	RAM(GB)
HP ProLiant ML110 G4	1860	2	4
HP ProLiant ML110 G5	2660	2	4
Express5800/GT110f-S	2500	4	8
IBM System x3450	2800	8	16

TABLE VIII: Heterogeneous VM Models

Model	CPU(MIPS)	Core	RAM(GB)
High-CPU Medium Instances	2500	1	0.87
Extra Large Instances	2000	1	3.74
Small Instances	1000	1	1.74
Micro Instances	500	1	0.613
g1.s Instance	2300	1	1.00
g1.m Instance	2300	1	2.00
g2.m Instance	2100	1	1.00
g2.l Instance	2100	1	1.92

2) Workload trace

To enhance the persuasiveness of our simulation experiments, the resource utilization of the VMs follows the traces from two publicly available real-world datasets, including PlanetLab [20] and Google Cluster Trace [25].

- **PlanetLab Trace**: The CoMon project generated this workload data by monitoring the infrastructure of PlanetLab located at more than 500 places around the world. Measured on 10 different days in March and April 2011, the workload data includes CPU utilization of VMs recorded at 5-minute intervals. Each VM contains 288 CPU utilization records in one day. The characteristics

[2]https://aws.amazon.com/ec2/instance-types/

of the workloads are presented in Table IX, where W1 to W10 represent the first day to the tenth day.

- **Google Cluster Trace**: Similar to PlanetLab Trace, Google generated Google Cluster Trace by monitoring the VMs in their cloud data centers for up to 29 days in May 2011. Google Cluster Trace is a large dataset with CPU and memory resource utilization of VMs collected every 5 minutes. To create the CPU utilization records for VMs, we randomly selected a total of 1600 subtasks from over 650 thousand jobs and extracted task utilization values of CPU over one day. The characteristics of the Google Cluster Trace is presented in Table X.

Since these two traces can reflect the characteristics of data flow in real-world cluster data centers, they have been widely used by the VM consolidation strategies [5], [10], [18], [37], [39].

TABLE IX: The Characteristics of PlanetLab Trace

Workload	No. of VMs	Mean(%)	St.dev.(%)	Median(%)
W1	1052	12.31	17.09	6
W2	898	11.44	16.83	5
W3	1061	10.70	15.57	4
W4	1516	9.26	12.78	5
W5	1078	10.56	14.14	6
W6	1463	12.39	16.55	6
W7	1358	11.12	15.10	6
W8	1233	11.56	15.07	6
W9	1054	11.54	15.15	6
W10	1033	10.43	15.21	4

TABLE X: The Characteristics of Google Cluster Trace

Workload	No. of PMs	No. of VMs	Mean(%)	St.dev.(%)	Median(%)
G1	800	1600	5.57	6.45	3.00
G2	2000	4000	11.22	10.02	8.00
G3	4000	8000	10.76	9.59	8.00

B. Performance Metrics

From the perspective of cloud service providers, we have established several metrics to evaluate the performance of our CTVMC strategy. The definitions of these metrics are defined as follows.

- **Energy consumption** Many components in cloud data center consume energy, such as CPU, memory and cooling systems, etc. With the expansion of cloud data center infrastructure, the surge in energy consumption is the most concerning issue for cloud service providers. The energy consumption of the host is linearly correlated with the CPU utilization of the host [16]. That means, the CPU utilization is the most important factor in both energy consumption and SLA compliance [10] [18] [39] [45]. So we use $P_{H_i}^{power}$ to express the amount of energy (in watts) that H_i consumes at current time t:

$$P_{H_i}^{power}(t) = P_{H_i}^{idle}(t) + (P_{H_i}^{max}(t) - P_{H_i}^{idle}(t)) \cdot H_i^R(t) \tag{13}$$

where $P_{H_i}^{idle}(t)$ represents the power consumption of H_i at idle and $P_{H_i}^{max}(t)$ represents the peak power consumption of H_i. The total energy consumption E during cloud data center operation can be calculated in Eq. (14)

$$E = \sum_{i=1}^{N} \int_0^t P_{H_i}^{power}(t)dt \tag{14}$$

We use SPECpower benchmark[3] based on real data to measure the energy consumption of the above four PMs. The configuration and power consumption characteristics are presented in Table VI.

- **SLA Violations**
The cloud service providers usually propose SLA constraints in cloud data center to maintain QoS for users. However, SLAV inevitably occurs when host overloading causes performance degradation or the VM is migrated to another host. Therefore, SLAV is a combined metric composed of SLATAH and PDM:

$$SLAV = SLATAH \cdot PDM, \tag{15}$$

where SLATAH represents the proportion of time that the host experiences resource utilization exceeding 100%, and PDM represents the performance degradation caused by the VM migration. SLATAH and PDM can be calculated by the Eq. (16) and Eq. (17), respectively:

$$SLATAH = \frac{1}{N} \sum_{i=1}^{N} \frac{T_{s_i}}{T_{a_i}} \tag{16}$$

$$PDM = \frac{1}{M} \sum_{j=1}^{M} \frac{C_{d_j}}{C_{r_j}} \tag{17}$$

where T_{s_i} represents the total time that the resource utilization of the host reaches 100%, T_{a_i} represents the total time that the host is active, C_{d_j} represents performance degradation due to VM migration, and C_{r_j} represents the total CPU utilization requested by a VM during its lifetime. The average performance degradation during a VM migration can be estimated as approximately 10% of the VM's current CPU utilization. It is worth noting that selecting a VM with more memory for migration under the condition of limited bandwidth will prolong the migration time, resulting in higher performance degradation.

- **Number of VM migrations**
Although VM live migration technology can reallocate VMs and improve the resource utilization of hosts, it also leads to additional energy consumption and affects the performance of applications running on the VM. Therefore, the number of VM migrations should be reduced to avoid SLAV.
- **Energy consumption and SLAV (ESV)**
The objective of our CTVMC strategy is to reduce energy consumption and SLAV, simultaneously. However, these metrics are negatively correlated as energy consumption usually be decreased by the cost of increasing SLAV. In

[3] https://www.spec.org/power_ssj2008/results

order to obtain the trade-off between energy consumption and SLAV, we propose another combined metric, called ESV, which is calculated by the product of total energy consumption and SLAV, as shown in Eq. (5(d)).

$$ESV = E \cdot SLAV \tag{18}$$

C. Simulation Result

To verify the efficiency of the proposed CTVMC strategy, we will compare it with other well-known strategies. To ensure the fairness of the evaluation, we set the overload threshold of the host to a static value of 80% for all strategies. As for host underload detection, we chose a simple policy preset by CloudSim, which selects the host with the lowest resource utilization as the underload host.

Two classical VM selection strategies [22], MMT and MC, are proposed. In MMT, the migrated VM with the shortest migration time is selected, which is mainly related to the memory size of the VM and the bandwidth of the host. In MC, the VM with the highest correlation with the CPU utilization of other VMs is selected for migration. Then, we introduce four VM placement policies, including FFD, PABFD, PAVMP, UP-VMC, and PEAS. FFD is a simple policy, which tries to allocate a VM to the first host from the host list that can provide enough resources.

Since our strategy mainly focuses on VM selection and VM placement, we need to combine VM placement strategies and VM selection strategies in different studies to form a complete VM consolidation strategy for comparison. We chose the five best-combined strategies as our comparison algorithms, namely MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC, and PEAS.

Through the metrics of energy consumption, number of migrations, SLAV and ESV, and ten days of PlanetLab Trace, we can form a total of 4 * 10 = 40 groups of comparison experiments, as shown in Table XI. The detailed experimental results of the Google Cluster Trace can also be shown in Table XII. In general, the third cloumn of Table XI and XII shows that the CTVMC, PEAS, and PABFD are comparable and achieve high performance in terms of energy consumption. In terms of number of migrations, SLAV, and ESV metrics, while our CTVMC strategy remains robust, other strategies exhibit more significant performance fluctuations. These strategies sacrifices SLAV and ESV performance to save energy. To perform intuitive analysis, we visualize the results by drawing charts of the different metrics, as shown in Table XI, TableXII, Figure 5, and Figure 6.

1) Evaluating Energy Consumption

According to the PlanetLab Trace, we compare the energy consumption of the CTVMC with the combined strategies of MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC and PEAS under different workloads as shown in Figure 5(a). The enerty consumptions consumptions of CTVMC, PEAS, and MMT+PABFD are comparable which are much lower than that of other three strategies. However, it can be found that the energy consumption of CTVMC is about about 5.1% and 10.68% lower than that of PEAS and MMT+PABFD.

TABLE XI: Simulation results of PlanetLab Data

Workload	VM consolidation strategies	Energy consumption(kWh)	Migration	SLAV($*10^{-6}$)	ESV($*10^{-4}$)
W1	MC+PAVMP	56.33	17220	49.2	27.71
	PEAS	25.41	27690	7	1.78
	MMT+FFD	67.11	12119	7.8	5.23
	MMT+PABFD	28.66	21875	32	9.17
	CTVMC	26.05	3276	1.6	0.42
	MMT+UP-VMC	67.83	12125	8.3	5.63
W2	MC+PAVMP	43.53	12659	46.3	20.15
	PEAS	18.66	20826	7.3	1.36
	MMT+FFD	51.88	8773	8.8	4.57
	MMT+PABFD	21.3	15998	30.8	6.56
	CTVMC	19.05	2387	1.3	0.25
	MMT+UP-VMC	49.52	9336	7.8	3.86
W3	MC+PAVMP	44.78	15013	50.6	22.66
	PEAS	21.18	18426	4.6	0.97
	MMT+FFD	52.95	9694	8.2	4.34
	MMT+PABFD	22.8	18419	35.3	8.05
	CTVMC	20.59	3173	2.3	0.47
	MMT+UP-VMC	51.79	11472	11	5.7
W4	MC+PAVMP	59.69	20660	47.3	28.23
	PEAS	25.55	24689	3.4	0.87
	MMT+FFD	65.83	14042	6.5	4.28
	MMT+PABFD	29.39	31402	48.8	14.34
	CTVMC	25.67	3815	2	0.51
	MMT+UP-VMC	64.23	15006	6.8	4.37
W5	MC+PAVMP	50.17	15588	42.2	21.17
	PEAS	20.93	21384	5.2	1.01
	MMT+FFD	56.09	10805	7.1	3.98
	MMT+PABFD	24.39	21276	38	9.27
	CTVMC	21.34	2975	1.8	0.38
	MMT+UP-VMC	55.57	11308	10.5	5.83
W6	MC+PAVMP	75.55	24359	53.7	40.57
	PEAS	43.74	48722	15.2	6.65
	MMT+FFD	97.43	17107	7.7	7.5
	MMT+PABFD	40.23	33049	37.8	15.21
	CTVMC	35.74	3560	1.6	0.57
	MMT+UP-VMC	95.08	16928	11.2	10.65
W7	MC+PAVMP	63.62	19761	52.2	33.21
	PEAS	27.86	34550	9.5	2.65
	MMT+FFD	74.31	13993	6.8	5.05
	MMT+PABFD	32.15	26185	36.5	11.73
	CTVMC	29.12	4136	2.1	0.61
	MMT+UP-VMC	73.7	14425	9.7	7.15
W8	MC+PAVMP	61.27	19553	51.5	31.55
	PEAS	28.98	34305	11.2	3.25
	MMT+FFD	74.63	13391	7.6	5.67
	MMT+PABFD	30.93	27149	39.7	12.23
	CTVMC	27.74	3808	1.9	0.53
	MMT+UP-VMC	71.63	13902	8.9	6.37
W9	MC+PAVMP	57.37	17433	52.9	30.35
	PEAS	29.51	25971	8.2	2.42
	MMT+FFD	64.99	11445	7.9	5.13
	MMT+PABFD	26.63	20335	31.6	8.42
	CTVMC	24	3003	1.4	0.34
	MMT+UP-VMC	61.56	12544	8.5	5.23
W10	MC+PAVMP	46.27	16424	67.3	31.14
	PEAS	20.68	34507	22	4.55
	MMT+FFD	53.29	11332	11.1	5.92
	MMT+PABFD	22.15	21919	46.2	10.23
	CTVMC	19.56	3745	3.2	0.63
	MMT+UP-VMC	49.78	12608	14.4	7.17

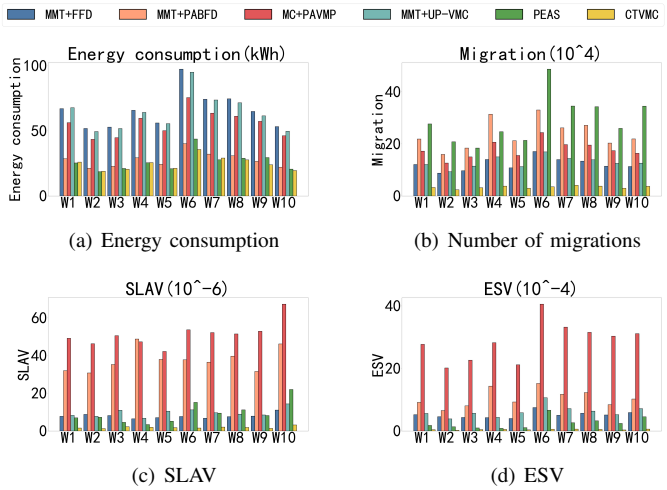
TABLE XII: Simulation results of Google Cluster Trace

Workload	VM consolidation strategies	Energy consumption(kWh)	Migration	SLAV($*10^{-6}$)	ESV($*10^{-4}$)
G1	MC+PAVMP	60.09	30215	17.1	10.28
	PEAS	48.14	51746	6	2.89
	MMT+FFD	57.85	27587	7.9	4.57
	MMT+PABFD	35.38	62937	66.8	23.63
	CTVMC	38.02	6769	2.4	0.91
	MMT+UP-VMC	55.97	33565	11.7	6.55
G2	MC+PAVMP	246.71	99700	26.7	65.87
	PEAS	217.72	188773	6.9	15.02
	MMT+FFD	294.09	113321	15.8	46.47
	MMT+PABFD	134.38	171342	48.2	64.77
	CTVMC	200.54	26253	4.6	9.22
	MMT+UP-VMC	282.99	82700	11.2	31.69
G3	MC+PAVMP	462.03	188361	26.7	123.36
	PEAS	451.34	285645	6.1	27.53
	MMT+FFD	554.54	241584	21.6	119.78
	MMT+PABFD	294.64	295335	44.4	130.82
	CTVMC	393.19	66971	4.9	19.27
	MMT+UP-VMC	539.28	173571	15	80.9

Compared to MMT+FFD, MC+PAVMP, and MMT+UP-VMC, the CTVMC outperforms these strategies by 62.21%, 55.45%, and 61.16%, respectively.

The second set of experiments was performed by using the Google Cluster Trace. In order to minimize energy consumption, the PABFD ignores the performance degradation of virtual machines caused by virtual machine migration and host overload. From Figure 6(a), we can see that the energy consumption of CTVMC is about 30.31%, 17.83%, 28.07%, and 11.91% lower than that of MMT+FFD, MC+PAVMP, MMT+UP-VMC, and PEAS, respectively.

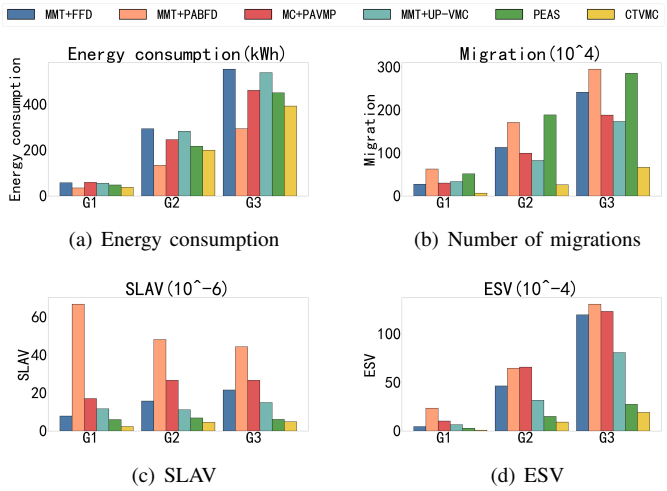
It can be seen that our strategy still has a significant advan-



(a) Energy consumption (b) Number of migrations

(c) SLAV (d) ESV

Fig. 5: Performance of different metrics in the PlanetLab Trace.



(a) Energy consumption (b) Number of migrations

(c) SLAV (d) ESV

Fig. 6: Performance of different metrics in the Google Cluster Trace.

tage in terms of energy efficiency. Because the combined trend is used to match the VM and the host, peak shaving and valley filling can be achieved to effectively improve the utilization of hosts in the past, present, and future. When all VMs are placed on the appropriate host, the energy consumption of the system is naturally reduced.

2) Evaluating Number of Migrations

In Figure 5(b), the CTVMC and other strategies are compared in terms of the number of migrations using the PlanetLab Trace. It is obvious that the number of migrations generated by our strategy is much less than that of others. Compared with MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC and PEAS, the number of VM migrations in CTVMC is lower than that of compared strategies by 72.39%, 85.74%, 81.15%, 81.04%, and 88.35%, respectively. As shown in Figure 6(b), The performances of using Google Cluster Trace in the VM migration metrics are shown in Figure 6(b). Compared to MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC, and PEAS, our CTVMC outperforms these strategies by 73.86%, 81.12%, 68.58%, 61.51%, and 81.00%, respectively.

We can see that our strategy is also lower than that of other strategies. One reason for such a great performance improvement is that CTVMC selects the VM that is considered to have the greatest impact on overloaded hosts according to the combined trend, which means unnecessary VM migrations can be avoided. Another reason is that CTVMC placed all the VMs in the most appropriate hosts. After VM placement, the workloads of the hosts become stable, which significantly reduces the number of VM migrations caused by host overloading.

3) Evaluating SLAV

Figure 5(c) shows the performance comparison of the SLAV metric. Compared to MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC, and PEAS, the SLAV of CTVMC outperforms these benchmark strategies by about 75.85%, 94.9%, 96.26%, 80.23%, and 79.49% according to the PlanetLab trace, respectively. For Google Cluster Trace, the overall performance characteristics of SLAV are consistent with PlanetLab Trace. As shown in Figure 6(c), the SLAV of CTVMC is lower than that of MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC, and PEAS, by about 73.73%, 92.53%, 83.12%, 68.60%, and 37.37%, respectively.

VM migration and host overloading are the two main factors causing SLAV, affecting the performance and availability of cloud services. CTVMC is a novel strategy that prevents the hosts from being overloaded in both the past, present, and future while migrating VMs. By doing so, the likelihood of VM migration and host overloading happening is greatly reduced. Therefore, SLA performance improvement is expected.

4) Evaluating ESV

Figure 5(d) shows the comparisons of the ESV using the PlanetLab Trace. Since the ESV combines energy consumption and SLAV, it can well reflect the purpose of CTVMC to jointly optimize energy consumption and SLAV. The results for all workloads are similar to the SLAV. Due to the great improvement in SLAV and the number of VM migrations, the CTVMC outperforms the MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC and PEAS by 90.88%, 95.52%, 98.36%, 92.37% and 81.54%, respectively. In Google Cluster trace, as shown in Figure 6(d), the SLAV of CTVMC is lower than that of MMT+FFD, MMT+PABFD, MC+PAVMP, MMT+UP-VMC, and PEAS, by about 82.79%, 86.59%, 85.26%, 95.32%, and 35.30%, respectively.

It can be seen that CTVMC involves a successful trade-off between energy consumption and SLAV. Notably, compared with other state-of-art strategies, the performance of CTVMC has only a small fluctuation in terms of these metrics. The reason may be that this combined trend strategy can comprehensively analyze the state of a host, resulting in more stable performance.

Overall, our strategy has a better performance than other strategies significantly in the above metrics. These results confirm the applicability and superiority of the CTVMC strategy.

VI. Conclusion

Improving the energy efficiency of the cloud data center while enhancing the SLA performance has always been the focus of cloud service providers. Existing strategies overlook the impact of sudden bursts in historical resource utilization on future hosts, thereby failing to keep hosts stable in the past, present, and future while increasing SLAV. In this paper, we propose a combined trend VM consolidation strategy, namely CTVMC. To eliminate the workload growth trend of the overloaded host, the CTVMC strategy selects the VM with the largest combined trend to migrate. Based on the temporal locality and prediction technique, CTVMC jointly uses the past, present and future resource utilization of VMs and hosts to filter candidate hosts in VM placement. By using the combined trend, we can find the most complementary host for each VM to achieve continuous stability of the host and eliminate the short-term bursts of workloads. We have evaluated CTVMC in comparison with several state-of-the-art VM selection algorithms and VM placement algorithms. The extensive experimental results based on the real-world workload provided by PlanetLab and Google Cluster demonstrate that our proposed strategy can effectively reduce energy consumption, SLAV, number of VM migrations, and ESV.

In order to simulate the real resource request and response in cluster data centers, two widely used data sets are employed to examine the experiments in this manuscript, including PlanetLab Trace, and Google Cluster Trace. Since only the utilizations of CPU and memory are provided in these two traces, we can only analyze these two resources in the stimulate experiments. If we find another trace which can provide multiple resource information in the future, we will utilize the information to further improve the accuracy of the algorithm while maintaining the simplicity of the CTVMC strategy. The VM and container are two kinds of popular virtual resource management technologies. Compared with VM, the container can provide lighter and more agile virtual resource management strategies. If container-based workloads also present temporal locality characteristics, the CTVMC strategy can also be applied in the container-based CDC environments. The further analysis for container-based trace and constructing the improved CTVMC strategy, which is suitable for the container environment, will be one of our future works. In recent years, the reinforcement learning algorithms are used in prediction. For further study, we also attempt to incorporate reinforcement learning to improve the accuracy of the prediction in the CTVMC strategy.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

References

[1] T. Dillon, C. Wu, and E. Chang, "Cloud computing: issues and challenges," in *Proceedings of the 2010 24th IEEE international conference on advanced information networking and applications*. IEEE, 2010, pp. 27–33.

[2] M. Koot and F. Wijnhoven, "Usage impact on data center electricity needs: A system dynamic forecasting model," *Applied Energy*, vol. 291, p. 116798, 2021.

[3] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[4] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1366–1379, 2012.

[5] S.-Y. Hsieh, C.-S. Liu, R. Buyya, and A. Y. Zomaya, "Utilization-prediction-aware virtual machine consolidation approach for energy-efficient cloud data centers," *Journal of Parallel and Distributed Computing*, vol. 139, pp. 99–109, 2020.

[6] H. Zhao, Q. Wang, J. Wang, B. Wan, and S. Li, "Vm performance maximization and pm load balancing virtual machine placement in cloud," in *Proceedings of the 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE, 2020, pp. 857–864.

[7] F. Xu, F. Liu, H. Jin, and A. V. Vasilakos, "Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions," *Proceedings of the IEEE*, vol. 102, no. 1, pp. 11–31, 2013.

[8] F. Zhang, G. Liu, B. Zhao, P. Kasprzak, X. Fu, and R. Yahyapour, "Cbase: Fast virtual machine storage data migration with a new data center structure," *Journal of Parallel and Distributed Computing*, vol. 124, pp. 14–26, 2019.

[9] W. Lin, W. Wu, and L. He, "An on-line virtual machine consolidation strategy for dual improvement in performance and energy conservation of server clusters in cloud data centers," *IEEE Transactions on Services Computing*, 2019.

[10] H. Zhou, Q. Li, K.-K. R. Choo, and H. Zhu, "Dadta: A novel adaptive strategy for energy and performance efficient virtual machine consolidation," *Journal of Parallel and Distributed Computing*, vol. 121, pp. 15–26, 2018.

[11] X. Fu and C. Zhou, "Predicted affinity based virtual machine placement in cloud computing environments," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 246–255, 2017.

[12] Y. Laili, F. Tao, F. Wang, L. Zhang, and T. Lin, "An iterative budget algorithm for dynamic virtual machine consolidation under cloud computing environment," *IEEE Transactions on Services Computing*, vol. 14, no. 1, pp. 30–43, 2018.

[13] S. Padhy and J. Chou, "Mirage: A consolidation aware migration avoidance genetic job scheduling algorithm for virtualized data centers," *Journal of Parallel and Distributed Computing*, vol. 154, pp. 106–118, 2021.

[14] A. Murtazaev and S. Oh, "Sercon: Server consolidation algorithm using live migration of virtual machines for green computing," *IETE Technical Review*, vol. 28, no. 3, pp. 212–231, 2011.

[15] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. IEEE, 2010, pp. 577–578.

[16] 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.

[17] N. T. Hieu, M. Di Francesco, and A. Ylä-Jääski, "Virtual machine consolidation with usage prediction for energy-efficient cloud data centers," in *Proceedings of the 2015 IEEE 8th International Conference on Cloud Computing*. IEEE, 2015, pp. 750–757.

[18] M. Ranjbari and J. A. Torkestani, "A learning automata-based algorithm for energy and sla efficient consolidation of virtual machines in cloud data centers," *Journal of Parallel and Distributed Computing*, vol. 113, pp. 55–62, 2018.

[19] N. T. Hieu, M. Di Francesco, and A. Ylä-Jääski, "Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 186–199, 2017.

[20] K. Park and V. S. Pai, "Comon: a mostly-scalable monitoring system for planetlab," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65–74, 2006.

[21] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini, "Temporal locality in today's content caching: Why it matters and how to model it," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 5, pp. 5–12, 2013.

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

[23] G. Keller, M. Tighe, H. Lutfiyya, and M. Bauer, "An analysis of first fit heuristics for the virtual machine relocation problem," in *Proceedings of the 2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*. IEEE, 2012, pp. 406–413.

[24] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N. T. Hieu, and H. Tenhunen, "Energy-aware vm consolidation in cloud data centers using utilization prediction model," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524–536, 2016.

[25] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format+ schema," *Google Inc., White Paper*, vol. 1, pp. 1–14, 2011.

[26] Z. Liu, A. Wierman, Y. Chen, B. Razon, and N. Chen, "Data center demand response: Avoiding the coincident peak via workload shifting and local generation," *Performance Evaluation*, vol. 70, no. 10, pp. 770–791, 2013.

[27] I. Takouna, E. Alzaghoul, and C. Meinel, "Robust virtual machine consolidation for efficient energy and performance in virtualized data centers," in *Proceedings of the 2014 IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)*. IEEE, 2014, pp. 470–477.

[28] J. Ahmadi, A. Toroghi Haghighat, A. M. Rahmani, and R. Ravanmehr, "Confidence interval-based overload avoidance algorithm for virtual machine placement," *Software: Practice and Experience*, vol. 52, no. 10, pp. 2288–2311, 2022.

[29] H. Zhao, N. Feng, J. Li, G. Zhang, J. Wang, Q. Wang, and B. Wan, "Vm performance-aware virtual machine migration method based on ant colony optimization in cloud environment," *Journal of Parallel and Distributed Computing*, vol. 176, pp. 17–27, 2023.

[30] Z. Han, H. Tan, G. Chen, R. Wang, Y. Chen, and F. C. Lau, "Dynamic virtual machine management via approximate markov decision process," in *Proceedings of the IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[31] H. Monshizadeh Naeen, E. Zeinali, and A. Toroghi Haghighat, "Adaptive markov-based approach for dynamic virtual machine consolidation in cloud data centers with quality-of-service constraints," *Software: Practice and Experience*, vol. 50, no. 2, pp. 161–183, 2020.

[32] R. Yadav, W. Zhang, K. Li, C. Liu, M. Shafiq, and N. K. Karn, "An adaptive heuristic for managing energy consumption and overloaded hosts in a cloud data center," *Wireless Networks*, vol. 26, pp. 1905–1919, 2020.

[33] R. Yadav and W. Zhang, "Mereg: Managing energy-sla tradeoff for green mobile cloud computing," *Wireless Communications and Mobile Computing*, vol. 2017, pp. 1–11, 2017.

[34] R. Yadav, W. Zhang, H. Chen, and T. Guo, "Mums: Energy-aware vm selection scheme for cloud data center," in *2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*. IEEE, 2017, pp. 132–136.

[35] W. Zhang, R. Yadav, Y.-C. Tian, S. K. S. Tyagi, I. A. Elgendy, and O. Kaiwartya, "Two-phase industrial manufacturing service management for energy efficiency of data centers," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 11, pp. 7525–7536, 2022.

[36] M. Singh, P. Kumar, and S. Tyagi, "Adaptive energy-aware algorithms to minimize power consumption and sla violation in cloud computing," *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science)*, vol. 14, no. 4, pp. 1008–1015, 2021.

[37] D. Basu, X. Wang, Y. Hong, H. Chen, and S. Bressan, "Learn-as-you-go with megh: Efficient live migration of virtual machines," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 8, pp. 1786–1801, 2019.

[38] Y. Zhan, S. Guo, P. Li, and J. Zhang, "A deep reinforcement learning based offloading game in edge computing," *IEEE Transactions on Computers*, vol. 69, no. 6, pp. 883–893, 2020.

[39] J. Zeng, D. Ding, K. Kang, H. Xie, and Q. Yin, "Adaptive drl-based virtual machine consolidation in energy-efficient cloud data center," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 2991–3002, 2022.

[40] A. Aghasi, K. Jamshidi, A. Bohlooli, and B. Javadi, "A decentralized adaptation of model-free q-learning for thermal-aware energy-efficient virtual machine placement in cloud data centers," *Computer Networks*, vol. 224, p. 109624, 2023.

[41] M. Chehelgerdi-Samani and F. Safi-Esfahani, "Pcvm.arima: predictive consolidation of virtual machines applying arima method," *The Journal of Supercomputing*, vol. 77, pp. 2172–2206, 2021.

[42] F. Bliemel, "Theil's forecast accuracy coefficient: A clarification," *Journal of Marketing Research*, vol. 10, no. 4, pp. 444–446, 1973.

[43] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and 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.

[44] T. He, A. N. Toosi, and R. Buyya, "Camig: Concurrency-aware live migration management of multiple virtual machines in sdn-enabled clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 10, pp. 2318–2331, 2021.

[45] D. Basu, X. Wang, Y. Hong, H. Chen, and S. Bressan, "Learn-as-you-go with megh: Efficient live migration of virtual machines," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 8, pp. 1786–1801, 2019.

Yuxuan Chen received his B.S. in Information Management and Information Systems from Nanjing Agricultural University, China in 2020, and subsequently received a Master's Degree in Computer Technology from Jinan University in 2023. His research interests include cloud computing, and resource management in data centers.

Zhen Zhang received the BS and MS degrees in computer science from Jilin University, China, in 1999 and 2003, and PhD degree in College of Computer Science and Engineering from South China University of Technology, China, in 2011. He is currently a professor at the Department of Computer Science of Jinan University. His research interests include graph theory, parallel and distributed processing and complex networks.

Yuhui Deng is a professor at the Computer Science Department of Jinan University. Before joining Jinan University, Dr. Yuhui Deng worked at EMC Corporation as a senior research scientist from 2008 to 2009. He worked as a research officer at Cranfield University in the United Kingdom from 2005 to 2008. He received his Ph.D. degree in computer science from Huazhong University of Science and Technology in 2004. His research interests cover information storage, cloud computing, green computing, computer architecture, performance evaluation, etc.

Geyong Min received the BSc degree in computer science from the Huazhong University of Science and Technology, China, in 1995, and the PhD degree in computing science from the University of Glasgow, U.K., in 2003. He is a professor of high performance computing and networking with the Department of Mathematics and Computer Science within the College of Engineering, Mathematics and Physical Sciences, University of Exeter, U.K. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high performance computing, ubiquitous computing, modelling, and performance engineering. He serves as an associate editor of the IEEE Transactions on Cloud Computing, IEEE Transactions on Sustainable Computing, IEEE Transactions on Computers, and etc.

Lin Cui is currently with the Department of Computer Science at Jinan University, Guangzhou, China. He received the Ph.D. degree from City University of Hong Kong in 2013. He has broad interests in networking systems, with focuses on the following topics: cloud data center resource management, data center networking, software defined networking (SDN), virtualization, programmable data plane and so on.