2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application

VM Performance Optimization Virtual Machine Migration Method Based on Ant Colony Optimization in Cloud

Hui Zhao[1,2], Shangshu Li[1], Jing Wang[1*], Jianhua Li[1]
[1]*Xidian University, China*
[2]*Key Laboratory of Smart Human-Computer Interaction and Wearable Technology of Shaanxi Province, China*
{hzhao,wangjing}@mail.xidian.edu.cn

***Abstract*—Virtual machine migration (VMM) is one of the most commonly used technologies in cloud platforms. However, the existing VMM methods did not try to optimize VM performance for cloud users when migrating VMs, so as to affect the users' experiences. In this paper, a VM Performance Optimization Virtual Machine Migration method (POVMM) is proposed, which can bring benefits to both cloud users and cloud service providers. It first takes the workload of PM into account to establish an improved workload-based VM performance model, and then uses the trained model to predict VM performance after migration. Then it formulates VMM as a multi-objective optimization problem, whose optimization objectives include maximizing VM performance, minimizing migration costs of all migrated VMs and reducing the number of working PMs in cloud. Lastly, an ACO-based (ant colony optimization) algorithm, POVMM, is proposed to obtain the approximate optimal solution of the VMM problem. The simulation experiment is completed on the cloud simulation software, CloudSim. Through comparing with the other VMM algorithms, the POVMM algorithm has better results, which proves the effectiveness of the POVMM algorithm.**

***Index Terms*—Virtual machine migration, VM performance model, ant colony optimization, cloud computing**

I. Introduction

Virtual machine migration (VMM) technology is an effective approach to minimize energy and cost [1]–[7], optimize the network traffic [8], [9], make load balance among PMs [10]–[12], ensure service security [13], [14], and optimize multi-objectives [4], [8], [15]–[17] for cloud service providers. However, the existing VMM techniques did not consider how to optimize VM performance for cloud users when migrating VMs, which may affect the user's experience. As a result, optimize VM performance for cloud users remains an issue to be considered in migrating VMs in cloud.

In our previous works, we have proposed performance-aware static VM placement methods [18], [19]. However, the previous VM performance model was trained depending on the number of used vCPU kernels of PM. It did not consider the workload of the physical machine, so its predicted VM performance may not be accurate in some cases. Generally, the VM performance may be different with different PM workloads. In addition, they are just static placement of VMs, not dynamic migration of VMs. Therefore, they did not consider the migration cost optimization either.

In this paper, a VM Performance Optimization Virtual Machine Migration method (POVMM) is proposed, which can bring benefits to both cloud users and cloud service providers, i.e., it tries to optimize VM performance for cloud users, minimize the migration cost and reduce the number of working PMs for cloud service providers.

Firstly, we all known that heavy workload of PM may drop the performance of VMs which are running on the PM. We considered the above factor that PM workload will affect VM performance, and then re-trained the improved workload-based VM performance model in this paper. The new model is more consistent with the VM performance trend in the cloud platform, so it can estimate the VM performance more correctly.

Secondly, we consider the following three aspects when building the VMM model. First of all, green computing is one of the research hotspots in cloud computing. Timely shutting down unnecessary servers can effectively reduce energy consumption. Therefore, reducing the number of working PMs becomes one optimization goal. In addition, to reduce the impact on cloud users, minimize downtime and migration traffic should be considered when migrating VMs. Therefore, minimizing migration costs is another optimization goal. Finally, VM performance is the most direct concern of cloud users, so minimizing virtual machine performance degradation is also one of the goals. By determining the above optimization objectives, we then formulate the VM migration as a multi-objective optimization problem.

Thirdly, as we all know, VMM is an NP-hard problem [20]. Some intelligent algorithms, such as ACO (ant colony optimization), have been used to solve the NP-hard VMM problem. In this paper, a VM performance optimization VMM algorithm (POVMM) based on ACO

979-8-3503-1993-4/22/$31.00 ©2022 IEEE
DOI 10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00183

TABLE I
SOME IMPORTANT SYMBOLS USED IN THIS PAPER

Symbols	Meaning
M_j	The PM j, $1 \leq j \leq M$
V_i	The VM i, $1 \leq i \leq N$
mp_c^j	The performance of VMs placed on M_j
$mp_c^j(n)$	The previous VM performance model
$mp_c^j(u)$	The workload-based VM performance model
u_j	The current CPU utilization of M_j
u_j'	The new CPU utilization of M_j
vu_i	The CPU utilization of V_i
vp_c^i	The performance of V_i
mg_i	The migration cost of V_i
y_j	The state of M_j
$X = \{x_{ij}\}$	The current VM placement
$X' = \{x_{ij}'\}$	The new VM placement

is proposed to obtain the approximate optimal solution of the VMM problem. Of course, we also verified POVMM method and other VMM methods through some experiments. The experimental results show that the POVMM method is effective.

The rest of this article is organized as follows. We describe the workload-based virtual machine performance model in Section II. Section III gives the VM migration problem formulation. Section IV proposes POVMM algorithm, which is also confirmed experimentally in Section V. Finally, this paper is concluded in Section VI.

II. PM WORKLOAD-BASED VM PERFORMANCE MODEL

In this section, we will present how to build the workload-based VM performance model.

A. The previous VM performance model

The previous VM performance model was trained depending on the total number of used vCPU kernels in PM [18], [19].

Given a PM M_j is equipped with a limited amount of resources. We use M_c^j, M_m^j, and M_n^j to denote its processors (CPUs), memory, and bandwidth. Given a VM V_i running on PM M_j. v_c^i, v_m^i and v_n^i are its required processors (CPUs), memory, and bandwidth resource. Although there are different resources of VMs and PMs, for simplicity, we only focuses on the CPU performance of VMs in this paper. mp_c^j is denoted as the VM performance of VMs running on M_j. According to [18], [19], the previous VM performance model is formulated as formula (1).

$$mp_c^j(n) = \begin{cases} 0.9982 - 0.0082 \cdot n & n \leq M_c^j - M_{c,r}^j \\ 6.93/n + 0.1507 & otherwise \end{cases} \tag{1}$$

where $n = \sum_i v_c^i$, $M_{c,r}^j$ is CPU used by PM itself.

B. The PM workload-based VM performance modeling

As previously analyzed, the state of the VMs and the workload of the PMs were not considered when training the VM performance model, so the VM performance may not be predicted accurately. Therefore, we should take the VM working states and PM workload into consideration to build the workload-based VM performance model. We run some applications (Hyper PI [21], transcode videos and compress files) in VMs running on a PM with 16 kernels CPU and 32GB memory under different PM workloads respectively. The running times are recorded as the VM performances under various PM workloads. We then calculate its relative performances, which are drawn in Fig. 1. We can observe the VM performance drops fast when the workload of PM is heavy (more than 80%).

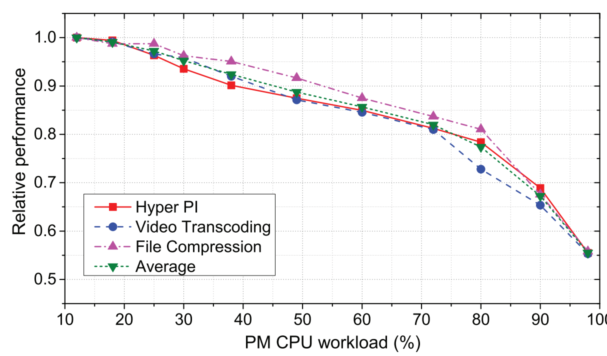


Fig. 1. The relative performance under various PM workloads.

Finally, the workload-based VM performance model can be denoted as:

$$mp_c^j(u) = \begin{cases} 1.0489 - 0.3288 \cdot u & 0 \leq u \leq 80\% \\ 1.7459 - 1.2074 \cdot u & 80\% < u \leq 100\% \end{cases} \tag{2}$$

where u is the CPU utilization (workload) of the PM.

III. VMM PROBLEM FORMULATION

In this section, we formulate the VM migration as a multi-objection optimization problem.

There are *M* PMs $M_j(j \in J = \{1, 2, ..., M\})$ and *N* VMs $V_i(i \in I = \{1, 2, ..., N\})$ in cloud, where $N >= M$. Of course, the resources of any VM cannot exceed the resources of a PM can offer. If V_i is running on M_j, its workload is denoted as vu_i. As a result, the workload of M_j is:

$$u_j = \frac{\sum_i (v_c^i \cdot vu_i) + M_{c,r}^j}{M_c^j} \tag{3}$$

As mentioned above, mp_c^j is the relative performance of VMs which are placed on M_j. If V_i is running on

M_j, then we assume $vp_c^i = mp_c^j$, i.e., the relative VM performance is the same as that of its host PM. So the performance of V_i could be calculated by Eq. (3) with VM performance model of Eq. (2).

$X = \{x_{ij}\}$ denotes the current VM placement. If V_i is placed on M_j, then $x_{ij} = 1$; otherwise, $x_{ij} = 0$. After VM migration, we set $X' = \{x'_{ij}\}$ be the new VM placement result. If $x_{ij} \neq x'_{ij}$, which means V_i is moved from M_j to $M_{j'}$. This will bring some VM migration cost (network traffic). In this paper, the migration cost of migrating V_i can be calculated based on pre-copy VMM technique [22].

$$mg_i = (\sum_{j=1}^{M}(|x'_{ij} - x_{ij}|)) \cdot (NT_i^{mig}) \tag{4}$$

where NT_i^{mig} is the total network traffic for migrating V_i, which is calculated according to [22].

$$NT_i^{mig} = \frac{v_m^i \cdot (1 - \lambda^{r+1})}{(1 - \lambda)} \tag{5}$$

Here, $\lambda = \frac{D}{R}$. R and D are the memory transmission rate and dirtying rate during VM migration. In general, D is less than R, i.e., $D < R$. $r = \lceil \log_{\lambda} \delta \rceil$ is the number of iterations of the pre-copying algorithm. $\delta (0 < \delta < 1)$ is the ratio of the set threshold to the memory.

After VM migration, the workload of M_j will be u'_j.

$$u'_j = \frac{y_j \cdot (M_{c,r}^j + \sum x'_{ij} \cdot v_c^i \cdot vu_i)}{M_c^j} \tag{6}$$

where y_j is the state of PM M_j. If M_j is still working, then $y_j = 1$; otherwise, $y_j = 0$ and $u'_j = 0$. After calculating u'_j of M_j, we can also calculate its VM performance with u'_j and the VM performance model of Eq. (2).

After VM migration, we use F, Q and H to denote the sum of all VM performances, the total migration cost (network traffic) of migrated VMs and the number of working PMs separately. They are calculated by:

$$F = \sum_{i=1}^{N}(vp_c^i) \tag{7}$$

$$Q = \sum_{i=1}^{N}(mg_i) \tag{8}$$

$$H = \sum_{j=1}^{M}(y_j) \tag{9}$$

As analyzed above, the formulated VMM optimization problem has three objectives, that is, maximizing VM performance, minimizing migration costs of all migrated VMs and reducing the number of working PMs in cloud. As a result, we define the VM migration problem as follows.

$$\begin{aligned} max(F) &= max(\sum_{i=1}^{N}(vp_c^i)) \\ and\ min(Q) &= min(\sum_{i=1}^{N}(mg_i)) \\ and\ min(H) &= min(\sum_{j=1}^{M}(y_j)) \end{aligned} \tag{10}$$

Subject to:

$$y_j = \{0, 1\}, \forall j \in J \tag{11}$$

$$x'_{ij} = \{0, 1\}, \forall j \in J, \forall i \in I \tag{12}$$

$$\sum_{j=1}^{M} x'_{ij} = 1, \forall i \in I \tag{13}$$

$$0 <= \sum_{i=1}^{N} x'_{ij} <= N, \forall j \in J \tag{14}$$

$$\sum_{i=1}^{N} x'_{ij} \cdot v^i \leq M^j, \forall j \in J \tag{15}$$

The formula (10) gives the optimization objectives of the VMM optimization problem. The formulas (11-15) are the constraint conditions of the VMM optimization problem.

IV. POVMM Algorithm

In this paper, we proposed an ACO-based algorithm (POVMM) to solve VMM problem. The problem (10) could be converted to (denoted as Π):

$$\begin{aligned} \Pi : min(W) &= min(Q/F) = min(\sum_{i=1}^{N}(\frac{mg_i}{vp_c^i})) \\ and\ min(H) &= min(\sum_{j=1}^{M}(y_j)) \end{aligned} \tag{16}$$

The pseudo-code for the ACO-based algorithm is list in Algorithm 1. Its main algorithmic procedure is described as follows.

Firstly, we initialize all parameters in the initialization phase. The trails of W and H are set to τ_w^0 and τ_h^0, and they are:

$$\tau_{ij,w}^0 = \frac{1}{W(S_0)} \tag{17}$$

$$\tau_{ij,h}^0 = \frac{1}{H(S_0)} \tag{18}$$

where S_0 is an initial solution which could be obtained by first fit decreasing algorithm.

Secondly, when the iteration starts, each ant selects a specific host PM for the unplaced virtual machine. The selected host PM need to meet the following condition:

$$j = \begin{cases} \max\limits_{u \in \Omega_k(i)} \{(\tau_{iu})^{\alpha} \cdot (\eta_{iu})^{\beta}\}, & q \leq q_0 \\ l, & \text{otherwise} \end{cases} \tag{19}$$

where $\Omega_k(i)$ represents the collection of PMs that can run the virtual machine V_i; q_0 is a fixed parameter, and $0 \leq q_0 \leq 1$.

$$\tau_{iu} = \sigma \cdot \tau_{iu,w} + (1 - \sigma) \cdot \tau_{iu,h} \tag{20}$$

$$\eta_{iu} = \frac{1}{W(S) \cdot F(S)} \tag{21}$$

where σ is a random value between 0 and 1; α and β are parameters to control the influence of τ_{iu} and η_{iu}.

Thirdly, ant will update the pheromone trail level using the following local updating rules when it have constructed a replacement of V_i on M_j.

$$\tau_{ij,w}(t) = (1 - \rho_l) \cdot \tau_{ij,w}(t-1) + \rho_l \cdot \tau^0_{ij,w} \tag{22}$$

$$\tau_{ij,h}(t) = (1 - \rho_l) \cdot \tau_{ij,h}(t-1) + \rho_l \cdot \tau^0_{ij,h} \tag{23}$$

where $\rho_l (0 < \rho_l < 1)$ is the local pheromone evaporating parameter.

Fourth, when obtaining all ants' solutions, the non-nominated solutions will be removed, and the remaining solutions will form a solution set, the Pareto set. Then the global pheromone should be updated in the Pareto set.

$$\tau_{ij,w}(t) = (1 - \rho_g) \cdot \tau_{ij,w}(t-1) + \frac{\rho_g \cdot \gamma}{W(S)} \tag{24}$$

$$\tau_{ij,h}(t) = (1 - \rho_g) \cdot \tau_{ij,h}(t-1) + \frac{\rho_g \cdot \gamma}{H(S)} \tag{25}$$

$$\gamma = \frac{N_{ant}}{t - N_{iter,s} + 1} \tag{26}$$

where N_{ant} is the number of ants. $N_{iter,s}$ is the iterations that the current solution s has resided in the external set.

Lastly, add the number of iteration by 1, and repeat the above procedure until the maximum iterations is exceeded. Choose one of the Pareto set as the solution X', and compare the X with X' to calculate F, Q, and H of the VMM solution.

V. Evaluation

A. Simulation settings

We use CloudSim framework [23], a well-known cloud simulation framework, to evaluate POVMM. In our simulations, we simulate a cloud platform of 200 homogeneous physics machines with 16 kernels CPU, 32GB memory and 1TB hard disk. The bandwidth among PMs is set to 100 Mbps. At the same time, we also simulate 400-1200 VMs with different configurations.

Algorithm 1 The ACO-based POVMM algorithm

Input: VMs $V = \{V_i\}$, PMs $M = \{M_j\}$, X
Output: X', F, Q, H
Initialize parameters: N_{ant}, N^{max}_{iter}, ρ_l, ρ_g, q_0, Pareto set S_{opt} and set $N_{iter} = 0$;
Get original solution S_0 generated by the first fit decreasing algorithm;
Initialize all pheromone values τ^0_w and τ^0_h;
while $N_{iter} < N^{max}_{iter}$ **do**
  for each ant k **do**
    Get ant k's current solution and insert it into S_{opt};
    Update the local pheromone with Eqs. (22) and (23);
  end for
  Remove non-dominated solutions in S_{opt};
  for each solution in S_{opt} **do**
    Update the global pheromone with Eqs. (24) and (25);
  end for
  $N_{iter}++$;
end while
Get an optimal solution X' from S_{opt};
Calculate F, Q, and H by comparing X' with X;
return X', F, Q, H

As for the ACO-based algorithm, we select 10 ants, i.e., $N_{ant} = 10$. The maximum number of iterations is 200, $N^{max}_{iter} = 200$. Meanwhile, we set $\alpha = 1$, $\beta \in \{1, 2\}$, $q_0 \in \{0.1, 0.4, 0.6, 0.9\}$, $\rho_l, \rho_g \in \{0.1, 0.3, 0.5, 0.8\}$. We first performed the experiments with different combinations of parameters separately. According to experiments results, the final ACO parameters are set as $\alpha = \beta = 1$, $q_0 = 0.9$, $\rho_l = \rho_g = 0.1$.

We compare POVMM with CMBFD-OM [4] and QoS-MMP [17] in term of three metrics, which are the average VM performance reduction, the total migration cost of migrated VMs and the number of working PMs.

B. Simulation results analysis

1) Results on the average VM performance reduction: This experiment is used to test the average VM performance reduction of different VMM algorithms by ranging the number of VMs from 400 to 1000. The simulation results are drawn in Fig. 2.

We can see that POVMM has the lowest average VM performance reduction compared with CMBFD-OM and QoS-MMP, which means POVMM can provide cloud users with better-performing VMs. This is because POVMM migrates VMs considering the performance degradation of VMs and it migrates VMs based on PM workload.

However, CMBFD-OM and QoS-MMP did not try to minimize the VM performance reduction during VM migration, therefore the VM performance decline significantly through the two VMM algorithms. Furthermore, the CPU utilization of CMBFD-OM is higher than that of QoS-MMP, which results in that it has more VM performance reduction than QoS-MMP.

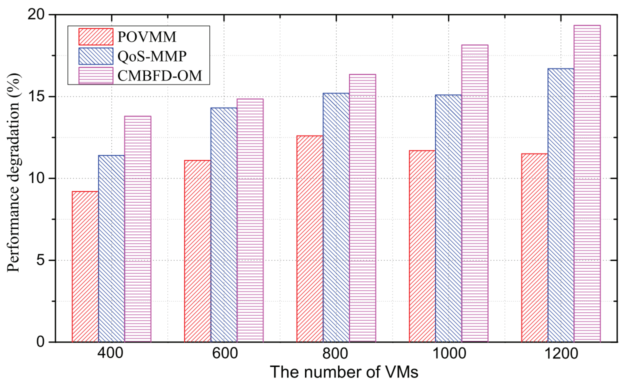


Fig. 2. Comparison results of different VMM methods on the average VM performance reduction.

2) Results on the total migration cost of migrated VMs: The second experiment is used to test the total migration cost of different VMM algorithms. We also set the number of VMs from 400 to 1000, increasing by 200 each time. Figure 3 draws the different results.

From the Fig. 3, we can know that compared to the other algorithms, POVMM has the smallest VM migration cost no matter how many VMs there are. In all cases, POVMM generates the least additional network traffic for migrating VMs. That is because POVMM preferentially choose and move VMs with smaller memory when there are VMs need to be migrated.

3) Results on the number of working PMs: In the last experiment, we compare POVMM with CMBFD-OM and QoS-MMP in terms of the number of working PMs. We set the number of VMs as 400 and 1000, and then we migrate VMs 10 times in each case. In each experiment, we count and record the number of working PMs for the different algorithms.

We use the boxplots to show the results of different VMM algorithms when the number of VMs is 400 and 1200 respectively. The results of statistical distribution of the number of working PMs are shown in Fig. 4(a) and 4(b).

As shown in the above figures, the number of working PMs of POVMM is slightly more than that of CMBFD-OM. POVMM tries to minimize VM performance degradation, resulting in a little more working PMs being needed. However, POVMM could optimize VM performance for cloud users, therefore, we consider it acceptable to need a little more additional working PMs.

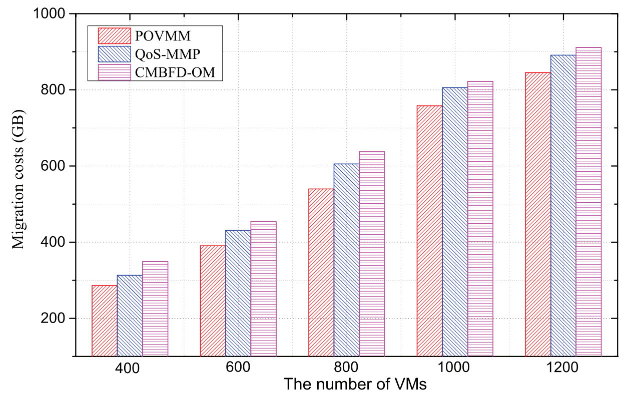
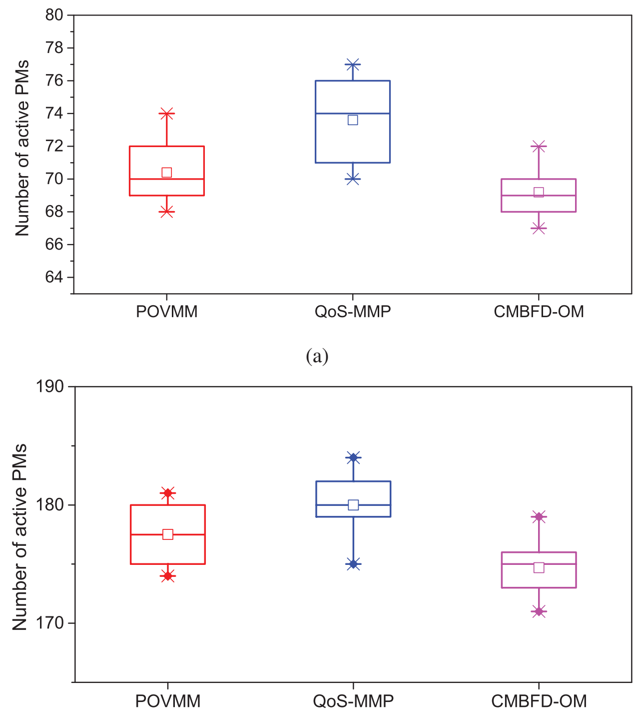


Fig. 3. Comparison results of different VMM methods on the total VM migration cost.



(b)

Fig. 4. Boxplots based on 10 independent runs when there are 400 VMs and 1200 VMs. (a) 400 VMs; and (b) 1200 VMs.

VI. Conclusions

This paper first analyzes the problem of VM performance reduction when migrating VMs in cloud computing. Based on the above analysis, a VM Performance Optimization Virtual Machine Migration method (POVMM) is proposed in this paper. It trains an improved workload-based VM performance model to predict VM performance after migrating VMs. Then it takes three objectives to formulate VM migration as a multi-objective optimization problem, and proposed an ACO-based algorithm to solve the NP-hard VMM problem. Finally, some experiment are conducted to investigate POVMM and other VMM methods. The results also prove the effectiveness of the POVMM algorithm.

As a future research direction, we plan to formulate our VMM strategy with other common optimization objectives, such as minimizing energy consumption of cloud, load balancing among PMs, etc.

ACKNOWLEDGMENT

The research was supported by the Fundamental Research Funds for the Central Universities (JB210312, JB210309), the National Science Foundation of China (61972302) and the Key Research and Development Program of Shaanxi (2021GY-014).

REFERENCES

[1] Q. Wu, F. Ishikawa, Q. Zhu, et al. Energy and Migration Cost-Aware Dynamic Virtual Machine Consolidation in Heterogeneous Cloud Datacenters[J]. IEEE Transactions on Services Computing, 2019, 12(4): 550-563.

[2] R. I. Meneguette, A. Boukerche. An Efficient Green-Aware Architecture for Virtual Machine Migration in Sustainable Vehicular Clouds[J]. IEEE Transactions on Sustainable Computing, 2020, 5(1): 25-36.

[3] D. Basu, X. Wang, Y. Hong, et al. Learn-as-you-go with Megh: Efficient Live Migration of Virtual Machines[J]. IEEE Transactions on Parallel and Distributed Systems, 2019, 30(8): 1786-1801.

[4] J.H. Wang, C.H. Huang, K. He, et al. An energy-awared resource allocation heuristics for VM scheduling in cloud[C]. Proceedings of 2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing. IEEE, 2013: 587-594.

[5] H. Wang, Y. Li, Y. Zhang, et al. Virtual Machine Migration Planning in Software-Defined Networks[J]. IEEE Transactions on Cloud Computing, 2019, 7(4): 1168-1182.

[6] V. Kherbache, E. Madelaine, F. Hermenier. Scheduling Live Migration of Virtual Machines[J]. IEEE Transactions on Cloud Computing, 2020, 8(1): 282-296.

[7] D. Fernando, J. Terner, K. Gopalan, et al. Live Migration Ate My VM: Recovering a Virtual Machine after Failure of Post-Copy Live Migration[C]. Proceedings of IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, Paris, France, 2019, pp. 343-351.

[8] J. Luo, X. Fan, L. Yin. Communication-Aware and Energy Saving Virtual Machine Allocation Algorithm in Data Center[C]. Proceedings of 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Zhangjiajie, China, 2019, pp. 819-826.

[9] V. Tran, J. Sun, B. Tang, et al. Traffic-Optimal Virtual Network Function Placement and Migration in Dynamic Cloud Data Centers[C]. Proceedings of 2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2022, pp. 919-929.

[10] H. Shen, L. Chen. A Resource Usage Intensity Aware Load Balancing Method for Virtual Machine Migration in Cloud Datacenters[J]. IEEE Transactions on Cloud Computing, 2020, 8(1): 17-31.

[11] B. Panda, S. C. Moharana, H. Das, et al. Energy Aware Virtual Machine Consolidation for Load Balancing in Virtualized Environment[C]. Proceedings of 2019 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2019, pp. 180-185.

[12] M.I.A. Bukhari, P. Luksch, A. Malekpour. Reducing Virtual Machine Migration Delay Using SCTP[C]. Proceedings of 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2018, pp. 1340-1344.

[13] H. M. Anitha, P. Jayarekha. Secure virtual machine migration in virtualized environment[C]. Proceedings of 2018 2nd International Conference on Inventive Systems and Control (ICISC), Coimbatore, 2018, pp. 938-943.

[14] X. Gao, J. Xiao, H. Wang, et al. Understanding the Security Implication of Aborting Virtual Machine Live Migration[J]. IEEE Transactions on Cloud Computing, 2022, 10(2): 1275-1286.

[15] N.T. Hieu, M.D. Francesco, A. Yla-Jaaski. Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers[J]. IEEE Transactions on Services Computing, 2020,13(1): 186-199.

[16] TianZhang He, Adel N. Toosi, Rajkumar Buyya. SLA-aware multiple migration planning and scheduling in SDN-NFV-enabled clouds[J]. Journal of Systems and Software, 2021, 176: 110943.

[17] Kamran, Nazir B. QoS-aware VM placement and migration for hybrid cloud infrastructure[J]. The Journal of Supercomputing, 2018, 74: 4623-4646.

[18] H. Zhao, J. Wang, F. Liu, et al. Power-aware and performance-guaranteed virtual machine placement in the cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(6): 1385-1400.

[19] H. Zhao, Q. Wang, Bo Wan, Shangshu Li. VM performance maximization and PM load balancing virtual machine placement in cloud[C]. Proceedings of 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), 2020: 857-864.

[20] W.Z. Zhang, Y.X. Chen, X. Gao, et al. Cluster-Aware Virtual Machine Collaborative Migration in Media Cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2017, 28(10): 2808-2822.

[21] Hyperpi, 2021. [Online]. Available: http://virgilioborges.com.br/hyperpi/.

[22] H. Liu, H. Jin, CZ Xu, et al. Performance and energy modeling for live migration of virtual machines[J]. Cluster Computing, 2013, 16(2): 249-264.

[23] CloudSim, 2021. [Online]. Available: http://www.cloudbus.org/cloudsim/.