Received October 29, 2019, accepted November 17, 2019, date of publication November 21, 2019, date of current version December 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2954903

Server Consolidation Energy-Saving Algorithm Based on Resource Reservation and Resource Allocation Strategy

TAO SONG[1], YUELIN WANG[1], GUILING LI[2], AND SHANCHEN PANG[1]
[1]College of Computer Science and Technology, China University of Petroleum, Qingdao 266580, China
[2]Academy of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding author: Guiling Li (qd-liguiling@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61572522, Grant 61572523, and Grant 61672033, and in part by the Key Research and Development Program of Shandong Province under Grant 2017GGX10147.

ABSTRACT Energy consumption has overtaken equipment costs given the growth of computing power, thereby becoming the dominant cost to data centers. Constrained by ensuring that peak resource requirements of VMs are met, the effective use of resource fragments is an effective means of energy conservation. We apply resource reservation to the resource-utilization-aware energy efficient server consolidation algorithm (RUAEE), and propose a server consolidation energy-saving algorithm (SCES). We adjust the allocations of VMs by computing the resource reservation and resource allocation ratio dynamically. This technique maintains the host resource utilization to within a reasonable range, reduces resource fragmentation and reserves resources to satisfy the peak resource requests simultaneously. Compared with the RUAEE algorithm, our algorithm can reduce the risk of system overload and improve the stability of the system. The experimental results based on the actual workload of the Google cluster show that the algorithm is effective at reducing resource waste, improving system stability, and achieving energy savings.

INDEX TERMS Cloud computing, resource fragmentation, resource allocation, online migration, resource reservation.

I. INTRODUCTION

The "pay-as-you-go" model provided by cloud computing provides users with convenient computing services. Many types of services provided by cloud providers, such as IaaS, PaaS, and SaaS, provide high scalability and flexibility. Users can use cloud services without being restricted by hardware and software systems. The cloud is a convenient service mode whereby users can choose to pay based on a calculated amount or by time.

Power consumption is a large portion of cloud computing operating costs [1]. Researchers speculate that US data center energy consumption will exceed 100 billion kWh by 2020 [2]. If the current trend continues, the energy cost of data centers will exceed the cost of the equipment [3]. Thus, saving energy and reducing the waste of resources are gaining attention in cloud computing.

Green computing is primarily geared toward data centers and concerns reducing energy consumption. Green computing proposes an energy-saving strategy to optimize data centers from software and hardware perspectives to reduce energy consumption [4]. The number of hosts in the data center can be effectively reduced by adjusting the virtual machine (VM) placement policy. However excessively concentrated VMs can cause system overload, reduce system stability and violate the service level agreement (SLA). Excessive VM migration can also incur additional overhead and reduce system availability. Therefore, reducing energy consumption cannot be at the expense of system stability or availability.

Another notable problem is resource fragmentation caused by server consolidation. Different VMs have different memory and CPU requirements. If we do not fully consider the balance of resources when placing a VM, resource fragmentation will occur, where one resource becomes depleted while another has a large surplus [5], [6]. As shown in Fig.1 (a), CPU utilization remains at 50%, and memory utilization remains at 15%. Fig.1 (b) shows that, CPU utilization remains at 10%, memory utilization remains at 40%. Both figures show that one utilization is much higher than another.

The associate editor coordinating the review of this manuscript and approving it for publication was Honghao Gao.

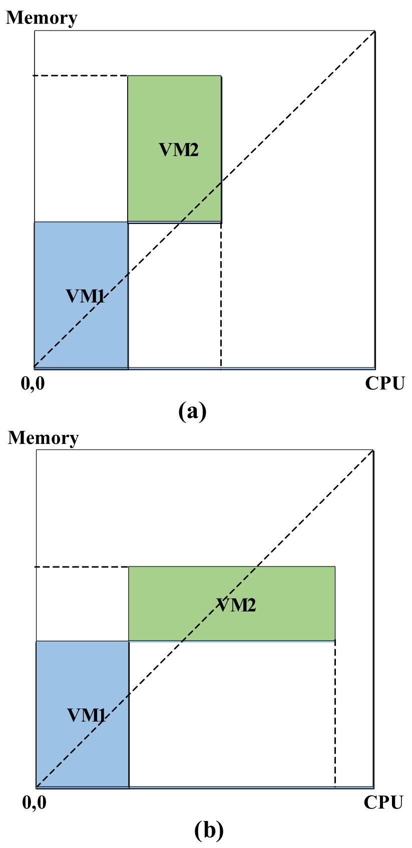


FIGURE 1. **Resource fragmentation.**

The resource fragmentation will not be alleviated until the VM is migrated or deleted.

In addition, excessively concentrated VMs can cause system overload when faced with workload bursts of the VMs [7]. System availability and stability will be reduced. Therefore, some resources should be reserved to address peak resource demands. However, excessive resource reservation will also waste resources and increase operating costs. It is a huge challenge to design a reasonable resource reservation policy to meet resource requirements of the peak tasks and avoid the waste of resources.

In this paper, we propose a server consolidation energy-saving algorithm based on resource reservation to reduce resource fragmentation. We reserve a certain amount of resources for the host to ensure that peak resource demands are satisfied, while avoiding excessive reservation. The main contributions of this paper can be summarized as follows.

- Resource reservation by the host is performed during the VM allocation process to prevent server overload caused by workload bursts.
- The execution mode of the resource-utilization-aware energy efficient server consolidation algorithm (RUAEE) is redesigned, and the VM is allocated given the resource reservation as the upper limit to reduce the risk of overload.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 gives the algorithm model. Section 4 gives the algorithm flow. The experiments and results are discussed in Section 5, and the Section 6 summarizes our work.

II. RELATED WORK

In recent years, many researchers have conducted research on energy conservation. Many studies are improving energy efficiency while reducing energy usage [8]. Researchers have proposed solutions at both the hardware and software levels. At the hardware level, some researchers have used dynamic voltage regulation to reduce energy consumption at data centers [9]. At the software level, Han *et al.* [10] adopted a VM management approach. In that paper, the author used a VM management method that approximates a large-scale Markov decision process to obtain an optimal solution to the problem.

Reducing resource fragmentation is also an energy-saving solution. Han *et al.* [11] used a resource utilization-aware energy efficiency service integration algorithm (RUAEE) in their paper. They considered resource fragmentation when assigning VMs, and controlled host resource utilization within a range, so that the combination of VMs could fully utilize host resources and reduce the generation of resource fragments. Similarly, Li *et al.* [5] also discussed the problem of resource fragmentation. They built a multi-dimensional spatial partitioning model to describe resource usage in the host. Based on the model, a VM placement algorithm EAGLE was proposed to balance the multi-dimensional resource usage status and effectively reduce energy consumption.

The online migration of VMs is an important technology for server consolidation. However, migration can have many negative effects, such as service disruptions, network congestion, and additional migration costs [12]. However, it is also important to increase resource utilization and reduce energy consumption by migrating VMs.

Optimization of a single target often results in performance loss in other aspects of the system. Malekloo *et al.* [13] used a multi-objective ant colony algorithm in their paper. They considered energy consumption, resource waste and communication cost in the algorithm. The objective function was designed by the pheromone volatilization mechanism unique to the ant colony algorithm. They achieved a balance between system performance and SLA compliance. Compared with typical algorithms that only consider energy consumption, the number of SLA violations was reduced, and the stability of the system was improved.

Workload bursts in VMs are widespread in the data centers [14]–[17]. Workload bursts will cause not only SLA violations, but also insufficient VM resources while also reducing task execution efficiency. Some previous work have studied the modeling and dynamic configuration of workload bursts in cloud computing [18]–[20]. Huang and Tsang [21] proposed to reserving certain resources in each host to address the fluctuation of work responsibility, but did not specify how much resources should be reserved. Jiang and Chen [22] proposed an online resource allocation algorithm. That algorithm took the topology of the data center as input, and then obtained the migration cost of the VMs, and resources were reserved for the host to prevent SLA violations. In this way, the energy consumption and migration times of the cloud data center and the migration path length were effectively reduced. Zhang *et al.* [23] used a two-state Markov chain-based switching method to predict the workload and reserved resources in a predictive manner to reduce the risk of server overload. Hussain and Aleem [24] used a predictive-based approach to predict workload growth and migrate VMs to meet peak resource requirements.

III. PROBLEM DESCRIPTION

A. RESOURCE UTILIZATION MODEL

The one-dimensional resource model can only describe the usage status of a single resource [25]–[28], [28]–[36]; it cannot accurately determine the resources required for VM allocation. To describe the usage of host resources in more detail, this paper takes two-dimensional resources to describe the usage of host resources [37]–[45]. As shown in Fig. 2, the abscissa indicates the usage of the CPU, and the ordinate indicates the usage of the memory.

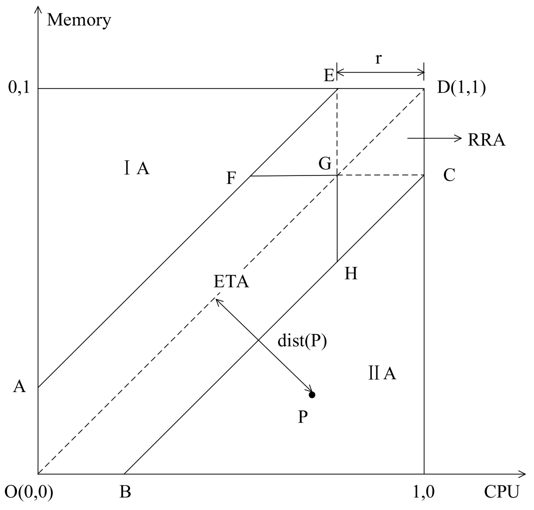


FIGURE 2. Resource utilization description model.

Resource reservation area (RRA): We define the area CDEFGH as the resource reservation area. The resources in this area are reserved to address any sudden growth of workloads and avoid SLA violations. CD and ED are the reserved memory and CPU resource sizes.

Equilibrium tolerance area (ETA) [11]: We define the area AOBHGF as an equilibrium tolerance area. The resources in this area have a reasonable utilization. The algorithm designed in this paper aims to dispatch hosts outside this area through VMs. The method of dispatching to the ETA makes the resource utilization more reasonable, while avoiding excessive use of resources and causing SLA violations.

Areas I and II (IA and IIA) [11]: The resource utilization in these regions is unbalanced, and resource waste is prone to occur. The memory utilization rate of the host in the area IA is too high, and the CPU utilization rate in the area IIA is too high. The migration algorithm performs adjustments to reduce resource fragmentation.

Suppose that point P represents the resource usage of a host, and the function f of the host resource usage can be expressed as follows:

$$f(x, y) = \begin{cases} RRA, & (1-x) \le r\&(1-y) \le r \\ ETA, & dist(p) \le \frac{\sqrt{2}}{2}r \\ & \&(1-x) > r\&(1-y) > r \\ IA \text{ or } IIA, & otherwise \end{cases} \quad (1)$$

where dist(p) represents the distance from point P to line x=y, x represents the CPU utilization, and y represents the memory utilization. The resource usage of the host is determined by the function f used to facilitate VM scheduling.

B. RESOURCE RESERVATION MODEL

Reserving resources for hosts helps prevent SLA violations due to workload bursts. In this paper, we apply Jiang's [22] VM resource reservation model. The burst state and non-burst state resource requirements of VM i are denoted as r_i^p and r_i^b, The difference between the two values r_i^p and r_i^b is denoted as Δ_i:

$$\Delta_i = r_i^p - r_i^b. \quad (2)$$

Δ_{max} denotes the maximum resource spike of the VMs in the host:

$$\Delta_{max} = \max_{0 \le j \le i} \Delta_j. \quad (3)$$

Δ_{avg} represent the average resource spike of the remaining VMs in the same host:

$$\Delta_{avg} = \sum\nolimits_{0<j\le i} \Delta_j / i. \quad (4)$$

The reserved resources of host j can be computed as follows:

$$R_j(\mathrm{t}) = \begin{cases} \Delta_{max}, & \text{only one VM} \\ \Delta_{max} + \eta\Delta_{avg}, & \text{otherwise} \end{cases} \quad (5)$$

where η is an experimental parameter. The obtained resource reservation value is used as the boundary value of the resource reservation area described in the previous section, which can effectively reduce SLA violations caused by the sudden growth of the workload while simultaneously ensuring the rationality of resource utilization and reducing resource fragmentation.

IV. ALGORITHM DESCRIPTION

The RUAEE algorithm periodically performs VM merge operations to reduce power consumption and uses a predictive-based approach to determine if the host is overloaded. The LR algorithm is used to estimate whether the CPU utilization of the host will exceed the host resource limit in the next cycle.

The MMT algorithm migrates the VM from the overloaded host. For the unbalanced host, the algorithm migrates the VM from the host to make the host utilization reach the ETA or HA area. For low-utilization hosts, the LR algorithm is used to predict the host CPU utilization of the next cycle. If it is lower than a preset threshold, the VM is migrated and the host is shut down. Finally, the VM placement module is the target host for migrating the VM identity.

This paper improves the RUAEE algorithm and redesigns the algorithm steps. The improved algorithm is divided into four parts: adjustment of the overloaded host, adjustment of the unbalanced host, adjustment of under-loaded host, and placement of the VM.

When the algorithm is executed, the resource reservation of the host is first calculated. The host is classified according to the resource usage. The VM in the overloaded host are reallocated, the unbalanced host are adjusted, the VM in the low-utilization host is migrated, and the host changes to a shutdown state. Finally, VM is migrated to the appropriate target host. These four modules work together to improve the resource utilization and reduce energy consumption in the data center.

The low-utilization host and the balanced host are used as candidate host sets, and the VMs in the overloaded host and unbalanced host are migrated to the candidate host set. After completing the adjustment of the overloaded and unbalanced hosts, we adjust for low-utilization hosts, migrate them to the appropriate host, and shut down the current host to bring the system to equilibrium, improve resource utilization, and reduce both resource fragmentation and data center energy consumption. The execution process is shown in Fig. 3.

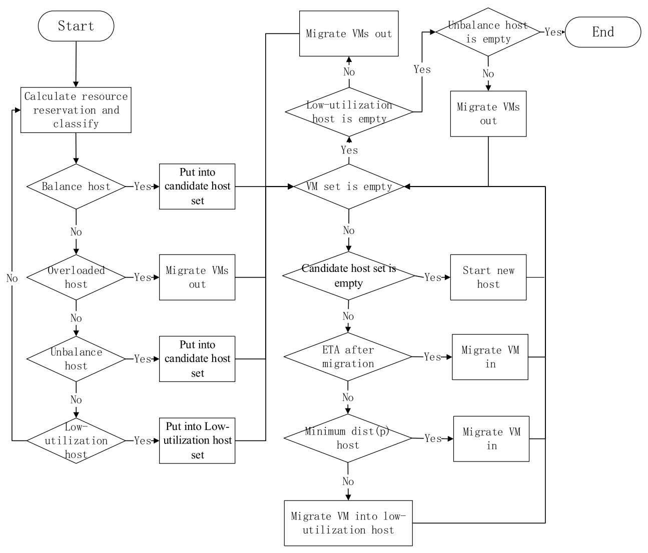


FIGURE 3. **Algorithm flowchart.**

A. ADJUSTMENT OF OVERLOADED HOST

Dynamic changes in VM resource requests from overloaded hosts can cause total resource demand to exceed the host capacity, which severely degrades the quality of service (QoS). To prevent an overload and minimize SLA violations, we should migrate the VM from the overloaded host to an appropriate host.

First, we calculate the resource reservation value $R_j(t)$ of each host. We let p be the current resource utilization of the host and calculate dist(p) of each host. We select the host in the RRA area and sort the VMs in descending order by their resource requirements. We add the VMs to the migration queue until the host falls into the ETA area, and invoke the VM placement module to migrate the VMs. The pseudo-code of the algorithm is given below.

Algorithm 1 Adjustment of Overloaded Host

```
1: Input: PM set Output: migrateVmList
2: for each host in PM set do
3:   if $f$(host) = RRA then
4:     migrateVmList = selectVm(host)
5:   end if
6: end for
7: return migrateVmList
```

B. ADJUSTMENT OF UNBALANCED HOST

To make the resource utilization more balanced and prevent resource fragmentation, we attempt to move the hosts in the IA and IIA area to the ETA area by migrating VMs. We calculate the resource reservation of each host and place the hosts from the IA and IIA areas into the host set to be adjusted. For the host set to be adjusted, the resource utilization falls into the ETA area when adding or removing the VM. The pseudo-code of the algorithm is given below.

Algorithm 2 Adjustment of Unbalanced Host

```
1: Input: PM set, migrateVmList Output: migrateVmList
2: for each host in PM set do
3:   if $f$(host) = $IA$ or $IIA$ then
4:     destinationHostList.add(host)
5:   end if
6: end for
7: if migrateVmList = null then
8:   for each host in PM set do
9:     if $f$(host) = $IA$ or $IIA$ then
10:      migrateVmList = selectVm(host)
11:    end if
12:  end for
13: end if
14: return migrateVmList
```

C. ADJUSTMENT OF LOW-UTILIZATION HOST

This module is designed to select hosts with lower utilization and turn them off after the VMs are migrated to other hosts to save power. The host is marked as a low-utilization host when the host resource usage is lower than a certain threshold, and the low-utilization host is first used as a candidate host with low priority. If the candidate host fails to meet the conditions, the VM is moved to the low-utilization host. When VM set is empty, if there is still a low-utilization host, the low-utilization host set is sorted in descending order of resource utilization, and the VMs are migrated out in order until there is no suitable host in the candidate host set or the low-utilization host set, which is empty. Considering the heterogeneity of the server and after many experiments, we chose 10% as the low utilization threshold. The pseudo-code of the algorithm is given below.

Algorithm 3 Adjustment of Low-Utilization Host

```
1: Input: low-utilizedHostList Output: migrateVmList
2: sort low-utilizedHostList in decreasing order by utilization
3: for each host in low-utilizedHostList do
4:   destinationLowHostList.add(host)
5: end for
6: if migrateVmList = null then
7:   if low-utilizedHostList ≠ null then
8:     for each host in low-utilizedHostList do
9:       migrateVmList = selectAllVm(host)
10:    end for
11:  end if
12: end if
13: return migrateVmList
```

D. VM PLACEMENT STRATEGY

Excessive VM migration can degrade system performance and lead to SLA violations. Therefore it is necessary to minimize the number of VM migrations.

When the migration module finds that there are VMs that need to be migrated, it is preferable to balance the host for migration and make the host utilization enter the ETA area. We select the host with the smallest dist(p) after the VM is placed from the candidate host set. Finally, if the algorithm still cannot find a suitable host, then a high utilization migration is chosen from low-utilization hosts. The pseudo-code of the algorithm is given below.

Algorithm 4 VM Placement Strategy

```
1: Input: migrateVmList, PM set Output: migrationList
2: for each host in PM set do
3:   if $f$(host) = ETA then
4:     destinationHostList.add(host)
5:   end if
6: end for
7: for each vm in migrateVmList do
8: flag ← 0
8:   for each host in destinationHostList do
9:     if $f$(allocatedHost(vm. host)) = ETA then
10:      migrationList(vm, host)
11:      migrateVmList.remove(vm)
12:      flag ← 1
13:      break
14:    end if
15:  end for
16:  if flag = 0 then
17:    for each host in destinationHostList do
18:      calculate smallest dist(allocatedHost(vm. host))
19:      if $f$(allocatedHost(vm. host)) ≠ RRA then
20:        migrationList(vm, host)
21:        migrateVmList.remove(vm)
22:        flag ← 1
23:      end if
24:    end for
25:  end if
26:  if flag = 0 then
27:    for each host in destinationLowHostList do
28:      migrationList(vm, host)
29:      migrateVmList.remove(vm)
30:      destinationLowHostList.remove(host)
31:    end for
32:  end if
33: return migrationList
```

V. EXPERIMENTS AND RESULTS

This chapter will introduce the simulation results of the algorithm and analyze it. Since it is extremely difficult to implement algorithms on large-scale cloud computing infrastructure, this experiment uses simulation experiments to evaluate the performance of the algorithm.

The experiment used Amazon EC2 instances as VMs. The data set uses the Google Cloud Task Dataset (GoCj) [25], which is the real dataset collected from Google servers. The experimental conclusion is based on the actual workload of the real server running in the simulation environment to ensure the credibility of the conclusions.

A. EXPERIMENTAL SETUP

We set up 800 heterogeneous hosts in the data center including four types of hosts: IBM server $\times$ 3250 (4 cores $\times$ 2933 MHz, 8 GB), IBM server $\times$ 3250 (4 cores$\times$3067 MHz, 8 GB,IBM server $\times$ 3550(6 cores $\times$ 2933 MHz, 12 GB), IBM server $\times$ 3550(6 cores $\times$ 3067 MHz, 12 GB). Table 1 shows the servers energy consumption data.

We set up five types of Amazon EC2 instances: micro instances (500 MIPS, 613 MB), small instances (1000 MIPS, 1.7 GB), medium instances (2000 MIPS, 3.75 GB), high CPU medium instance (2500 MIPS, 0.85 GB), large instances (3000MIPS, 5.5GB). The cloud task uses 1000 tasks in GoCj. GoCj is the Google Cloud Task Dataset, which is extracted from Google server logs.

B. RESULTS AND ANALYSIS

This section will discuss the implementation of the algorithm in the simulation environment compared with first fit (FF), EAGLE and RUAE. FF is a traditional greedy algorithm that

TABLE 1. **Server energy consumption and resource utilization.**

Host	*0%*	*10%*	*20%*	*30%*	*40%*	*50%*	*60%*	*70%*	*80%*	*90%*	*100%*
x3250(2933)	41.6	46.7	52.3	57.9	65.4	73	80.7	89.5	99.6	105	113
x3250(3067)	42.3	46.7	59.7	55.4	61.8	69.3	76.1	87	96.1	106	113
x3550(2933)	66	107	120	131	143	156	173	191	211	229	247
x3550(3067)	58.4	98	109	118	128	140	153	170	189	205	222

applies the least expensive selection every time it is executed however it does not necessarily achieve the global optimal choice. EAGLE [5] was an early algorithm in resource fragmentation research. EAGLE uses two-dimensional modeling of resources to integrate VMs. RUAEE [11] uses predictive methods to reduce SLA violations while considering resource fragmentation issues.

Resource fragmentation within the server is an important factor affecting server usage. Excessive resource fragmentation wastes server resources and incurs additional power consumption. The algorithm proposed in this paper models the resource fragmentation problem and fully considers resource fragmentation when allocating VMs; thus, it can effectively reduce the additional energy consumption caused by resource fragmentation. The result is shown in Fig. 4.

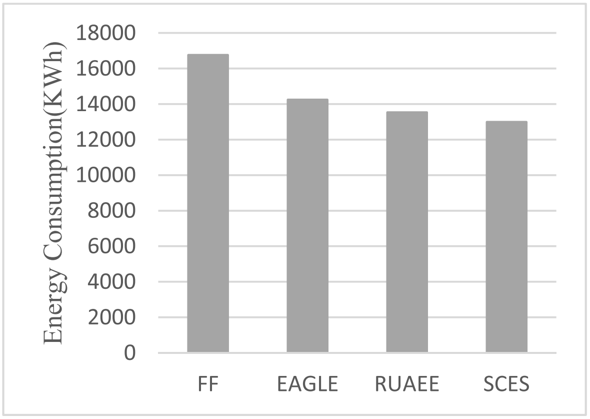
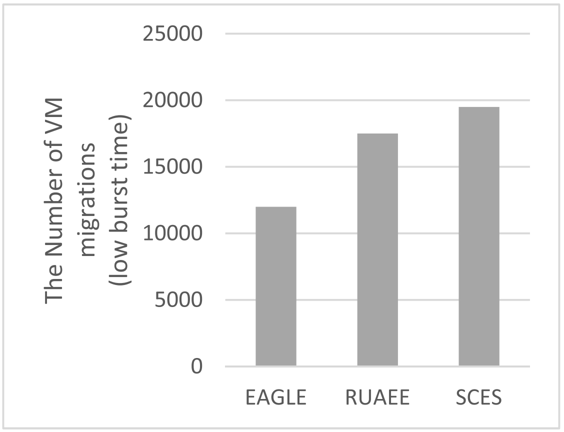


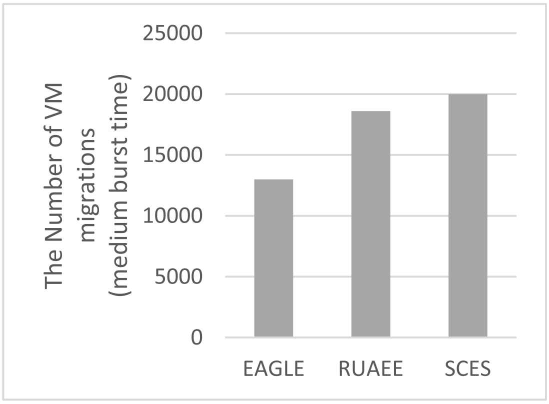
FIGURE 4. **Energy consumption.**

Fig. 4 shows the energy consumption of FF, EAGLE, RUAEE, and SECE in the data center. FF does not consider the problem of resource utilization. When assigning a VM, the VM is assigned to the first host that can satisfy the resource request, resulting in an imbalance of resource usage, which requires more hosts for placing of the VM and consumes more energy. EAGLE improves the allocation of resources and integrates resource fragments to reduce the number of active hosts and thus reduce energy consumption. Compared with EAGLE, RUAEE controls the resource allocation model in a smaller area when the resource allocation model is established, and the resource utilization rate is higher, thereby reducing the number of active hosts as well as the energy consumption. Compared with RUAEE, SCES uses reserved resources to reduce the number of host overloads and further reduce energy consumption.

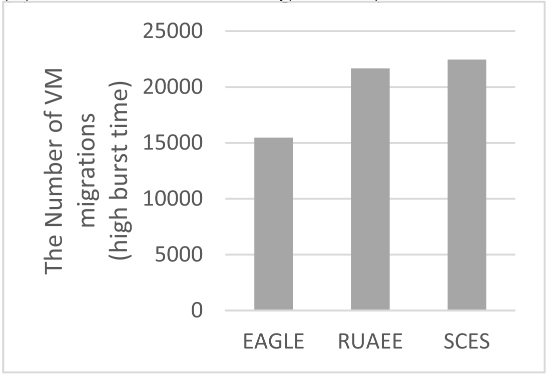
Fig. 5 shows the number of VM migrations for the three algorithms under different workload burst frequencies. Since FF is a static algorithm that does not consider the migration



(a) The number of VM migrations (low burst time)



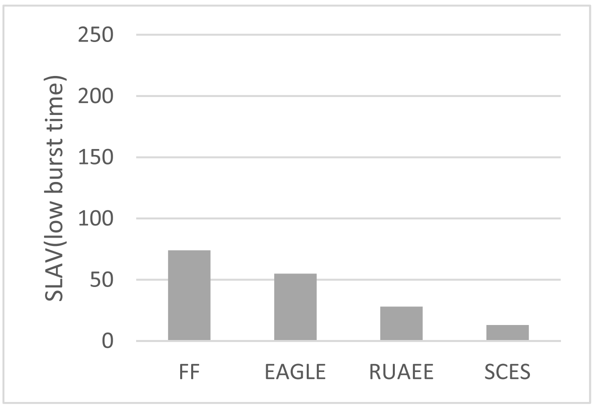
(b) The number of VM migrations (medium burst time)



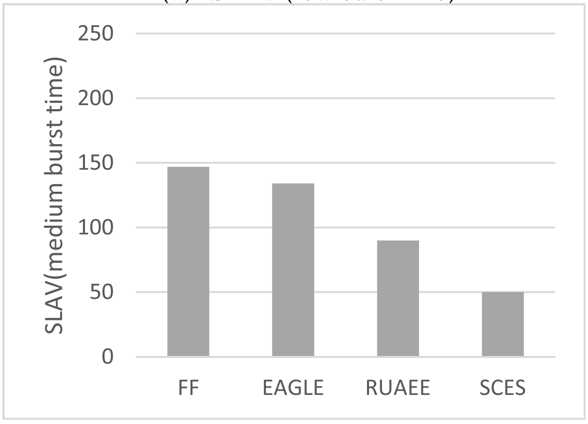
(c) The number of VM migrations (high burst time)

FIGURE 5. **Number of VM migrations.**

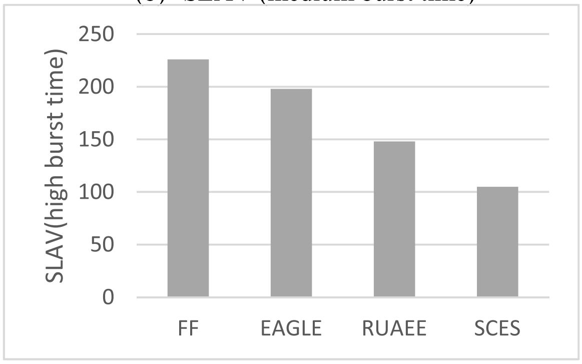
of VMs, it is not considered in this section. The acceptable resource area of EAGLE is the largest; thus, the number of VM migrations is the lowest. Compared with RUAEE, SCES has greater VM migration times since the VM needs to be moved out and redistributed when resource reservation



(a) SLAV (low burst time)



(b) SLAV (medium burst time)



(c) SLAV (high burst time)

FIGURE 6. SLAV.

is performed. RUAEE is based on the predictive migration of VMs to address workload bursts. Therefore, as the workload burst frequency increases, the gap with the SCES will become increasingly small. More accurate partitioning can better reduce resource fragmentation; however, this also means more VM migrations. We have minimized the number of VM migrations when designing algorithms, and the number of VM migrations continues increasing. This is inevitable, but the cost is acceptable compared to the increase in stability and the reduction in energy consumption.

Reference [13] noted that SLAV is used to indicate the number of times that the host reaches 100% utilization, thereby describing the host's ability to respond to workload bursts.

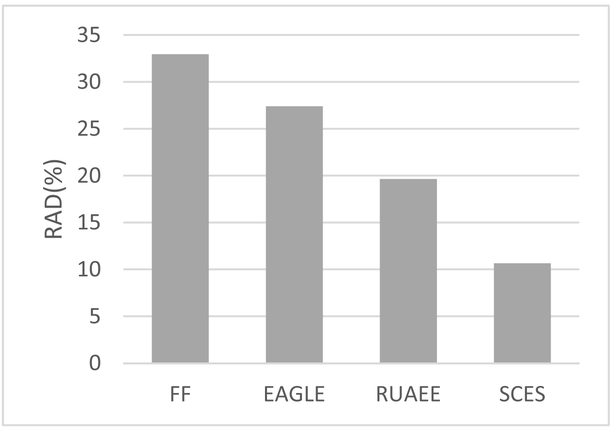


FIGURE 7. RAD.

Fig. 6 shows the number of SLA violations of four algorithms when experiencing workload bursts with different frequencies. FF and EAGLE do not consider the workload burst problem. The accuracy of the RUAEE based on the prediction method is also considered. SECS also degrades at higher burst frequencies because the algorithm cannot address different burst frequencies using the same reserved resources.

To measure the fragmentation of resources within the host, the following formula is introduced:

$$\mathrm{RAD} = \frac{\sum_{1}^{N} |U_{CPU}^{i} - U_{MEM}^{i}|}{N}. \tag{6}$$

where N represents the number of active hosts, U_{CPU}^{i} represents the CPU utilization, and U_{MEM}^{i} represents the memory utilization. Equation (6) can be used to characterize resource allocation in the host. The larger the RAD value is, the more unbalanced the resource allocation in the host, and the more resource fragments that exist. In contrast, the smaller the RAD value is, the more balanced the host resource allocation and fewer resource fragments exist. The experimental results are shown in Fig. 7.

Fig. 7 shows the case of host resource fragmentation, RAD represents the average of the difference in CPU and memory resource usage in the host. We would like that the ratios of CPU resources to memory resource allocation in the host to be as similar as possible. Both EAGLE and RUAEE optimize the resource fragmentation. EAGLE has a larger acceptable area; thus, the RAD is larger and has fewest VM migrations. RUAEE improves the utilization of the host as much as possible, and the acceptable domain is smaller; thus, a smaller RAD than the EAGLE algorithm is obtained. SCES does not pursue the highest resource utilization ratio with respect to RUAEE. Resources reserves make the host resource allocation more flexible, and the adjustment module for the unbalanced host also adjusts the host resources toward a more balanced target, thus obtaining the minimum RAD. The SECS algorithm can significantly reduce host resource fragmentation.

Fig. 8 shows the resource distribution status of the host after the execution of the four algorithms. Each point

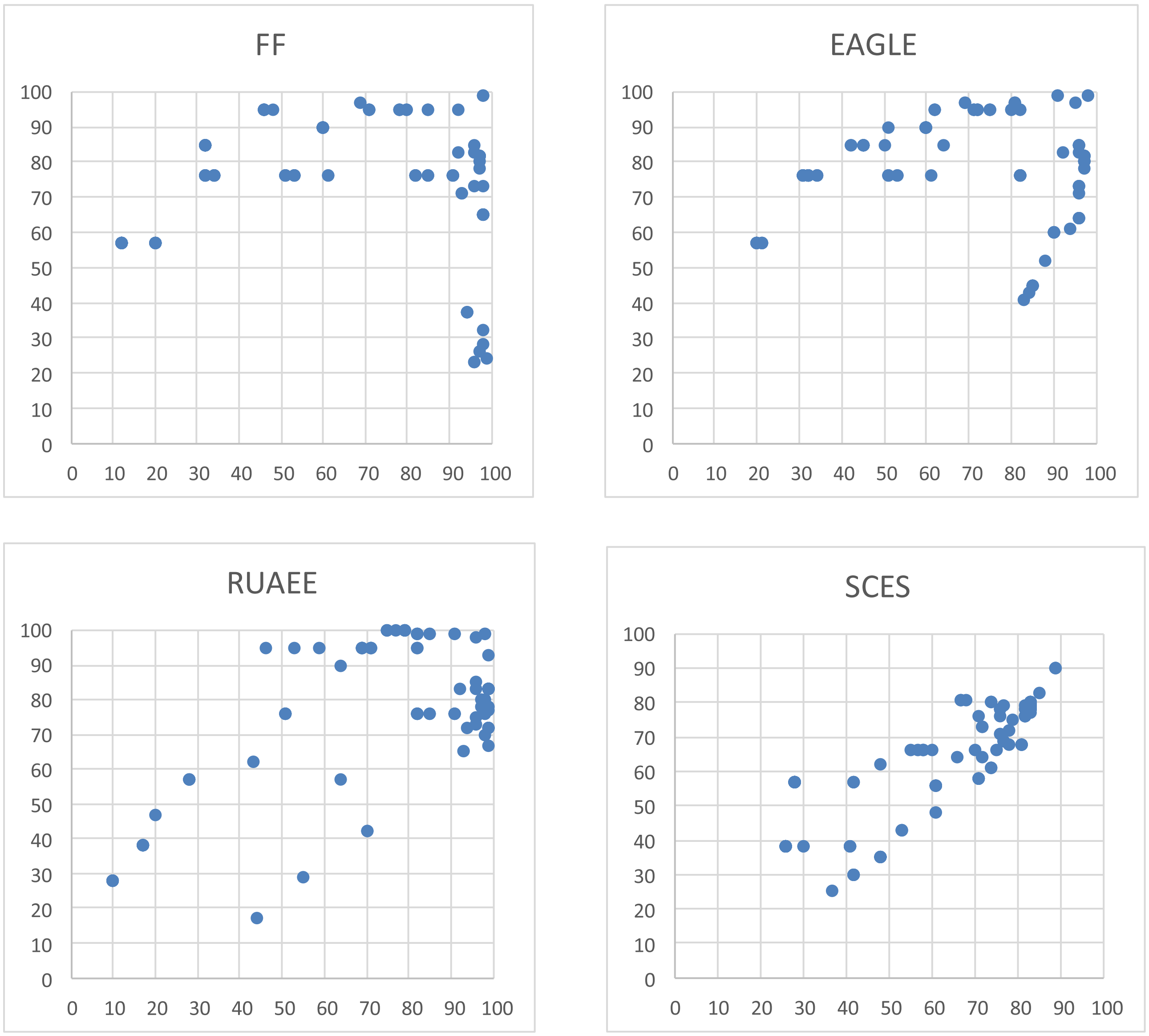


FIGURE 8. Host resource utilization distribution.

represents a host. The abscissa represents CPU utilization, and the ordinate represents memory utilization. FF does not consider the resource fragmentation problem; thus, the host distribution is very discrete. EAGLE uses a circular arc boundary when building the resource usage model; therefore the host distribution also has an arc shape. The host of the RUAEE is distributed around the high utilization rate of the line x=y. The host of the SCES algorithm is distributed near the line x=y, and reserves certain resources to address workload bursts. Compared with RUAEE, SECS can better address the fluctuation of resources demands since host resources have certain reservations and are not concentrated at approximately 100%.

VI. CONCLUSION

This paper proposes a server integration algorithm and resource allocation strategy based on resource reservation. Resource utilization by a host is usually characterized by the CPU utilization. However insufficient memory also affects the system performance. Therefore, this paper uses a 2D resource model to describe the utilization of host resources, and fully considers the resource fragmentation problem when assigning VMs toward ensuring host resource utilization balance. Workload bursts are another problem solved in this paper. This paper uses reserved resources to address workload bursts, and reserves some resources according to the current resource usage of the host. This reduces SLA violations effectively when experiencing workload bursts. Finally, we use the Google Cloud task dataset and Amazon EC2 instances as VMs to verify the effectiveness of the algorithm.

To further categorize the different intervals to improve the utilization of resources, we inevitably increase the number of VM migrations. In future work, we will consider adding more factors such as migration costs, communication costs and other more detailed energy consumption models. Moreover, it is necessary to further optimize the algorithm from the perspective of reducing VM migration, e.g. by adjusting the size of the tolerance area and adjusting the algorithm operation period.

REFERENCES

[1] M. Lin, R. Chen, J. Xiong, X. Li, and Z. Yao, "Efficient sequential data migration scheme considering dying data for HDD/SSD hybrid storage systems," *IEEE Access*, vol. 5, pp. 23366–23373, 2017.

[2] W. Josh and P. Delforge, "Data center efficiency assessment: Scaling up energy efficiency across the data center industry: Evaluating key drivers and barriers," NRDC Anthesis, Tech. Rep., 2014.

[3] R. Brown, "Report to congress on server and data center energy efficiency: Public law 109-431," Tech. Rep., 2008.

[4] R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges," 2010, *arXiv:1006.0308*. [Online]. Available: https://arxiv.org/abs/1006.0308

[5] X. Li, Z. Qian, S. Lu, and J. Wu, "Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center," *Math. Comput. Model.*, vol. 58, no. 5, pp. 1222–1235, 2013.

[6] W. Wei, X. Fan, H. Song, and X. Fan, and J. Yang, "Imperfect information dynamic stackelberg game based resource allocation using hidden Markov for cloud computing," *IEEE Trans. Serv. Comput.*, vol. 11, no. 1, pp. 78–89, Jan./Feb. 2016.

[7] W. Vogels, "Beyond server consolidation," *ACM Queue*, vol. 6, no. 1, pp. 20–26, 2008.

[8] K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, "Trends and challenges in cloud datacenters," *IEEE Cloud Comput.*, vol. 1, no. 1, pp. 10–20, May 2014.

[9] T. Horvath, T. Abdelzaher, K. Skadron, and X. Liu, "Dynamic voltage scaling in multitier Web servers with end-to-end delay control," *IEEE Trans. Comput.*, vol. 56, no. 4, pp. 444–458, Apr. 2007.

[10] Z. Han, H. Tan, G. Chen, R. Wang, Y. Chen, and F. C. Lau, "Dynamic virtual machine management via approximate Markov decision process," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr. 2016, pp. 1–9.

[11] G. Han, W. Que, G. Jia, and W. Zhang, "Resource-utilization-aware energy efficient server consolidation algorithm for green computing in IIOT," *J. Netw. Comput. Appl.*, vol. 103, pp. 205–214, Feb. 2018.

[12] M. F. Bari, M. F. Zhani, Q. Zhang, R. Ahmed, and R. Boutaba, "CQNCR: Optimal VM migration planning in cloud data centers," in *Proc. IEEE IFIP Netw. Conf.*, Jun. 2014, pp. 1–9.

[13] M.-H. Malekloo, N. Kara, and M. El Barachi, "An energy efficient and SLA compliant approach for resource allocation and consolidation in cloud computing environments," *Sustain. Comput., Inform. Syst.*, vol. 17, pp. 9–24, Mar. 2018.

[14] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: Measurements & analysis," in *Proc. 9th ACM SIGCOMM Conf. Internet Meas.*, 2009, pp. 202–208.

[15] N. Mi, G. Casale, L. Cherkasova, and E. Smirni, "Injecting realistic burstiness to a traditional client-server benchmark," in *Proc. 6th Int. Conf. Autonom. Comput.*, 2009, pp. 149–158.

[16] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: Incorporating time-varying network reservations in data centers," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 199–210, 2012.

[17] S. Zhang, Z. Qian, J. Wu, S. Lu, and L. Epstein, "Virtual network embedding with opportunistic resource sharing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 816–827, Mar. 2014.

[18] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing sla violations," in *Proc. 10th IFIP/IEEE Int. Symp. Integr. Netw. Manage.*, May 2007, pp. 119–128.

[19] G. Casale, N. Mi, and E. Smirni, "Model-driven system capacity planning under workload burstiness," *IEEE Trans. Comput.*, vol. 59, no. 1, pp. 66–80, Sep. 2009.

[20] G. Casale, N. Mi, L. Cherkasova, and E. Smirni, "Dealing with burstiness in multi-tier applications: Models and their parameterization," *IEEE Trans. Softw. Eng.*, vol. 38, no. 5, pp. 1040–1053, Sep. 2012.

[21] Z. Huang and D. H. Tsang, "SLA guaranteed virtual machine consolidation for computing clouds," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 1314–1319.

[22] H.-P. Jiang and W.-M. Chen, "Self-adaptive resource allocation for energy-aware virtual machine placement in dynamic computing cloud," *J. Netw. Comput. Appl.*, vol. 120, pp. 119–129, Oct. 2018.

[23] S. Zhang, Z. Qian, Z. Luo, J. Wu, and S. Lu, "Burstiness-aware resource reservation for server consolidation in computing clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 964–977, Apr. 2015.

[24] A. Hussain and M. Aleem, "GoCJ: Google cloud jobs dataset for distributed and cloud computing infrastructures," *Data*, vol. 3, no. 4, p. 38, 2018.

[25] W. Li, K. Liao, Q. He, and Y. Xia, "Performance-aware cost-effective resource provisioning for future grid IotT-cloud system," *J. Energy Eng.*, vol. 145, no. 5, 2019, Art. no. 04019016.

[26] J. Zhang, X. Wang, Y. Yuan, and L. Ni, "RcDT: Privacy preservation based on r-constrained dummy trajectory in mobile social networks," *IEEE Access*, vol. 7, pp. 90476–90486, 2019.

[27] X. Xiao, W. Zheng, Y. Xia, X. Sun, Q. Peng, and Y. Guo, "A workload-aware VM consolidation method based on coalitional game for energy-saving in cloud," *IEEE Access*, vol. 7, pp. 80421–80430, 2019.

[28] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, and Y. Huang, "Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 162–170, Jan. 2015.

[29] S. Pang, H. Chen, H. Liu, J. Yao, and M. Wang, "A deadlock resolution strategy based on spiking neural P systems," *J. Ambient Intell. Hum. Comput.*, pp. 1–12, 2019.

[30] Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, and Z. Mai, "Qos prediction for service recommendation with deep feature learning in edge computing environment," *Mobile Netw. Appl.*, pp. 1–11, Apr. 2019.

[31] Y. Yin, L. Chen, and J. Wan, "Location-aware service recommendation with enhanced probabilistic matrix factorization," *IEEE Access*, vol. 6, pp. 62815–62825, 2018.

[32] H. Gao, H. Miao, L. Liu, J. Kai, and K. Zhao, "Automated quantitative verification for service-based system design: A visualization transform tool perspective," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 28, no. 10, pp. 1369–1397, 2018.

[33] T. Song, S. Pang, S. Hao, A. Rodríguez-Patón, and P. Zheng, "A parallel image skeletonizing method using spiking neural P systems with weights," *Neural Process. Lett.*, vol. 50, no. 2, pp. 1485–1502, 2018.

[34] S. Wang, S. He, F. Yuan, and X. Zhu, "Tagging SNP-set selection with maximum information based on linkage disequilibrium structure in genome-wide association studies," *Bioinformatics*, vol. 33, no. 14, pp. 2078–2081, 2017.

[35] H. Gao, D. Chu, Y. Duan, and Y. Yin, "Probabilistic model checking-based service selection method for business process modeling," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 27, no. 6, pp. 897–923, Feb. 2017.

[36] Y. Yin, Y. Xu, W. Xu, M. Gao, L. Yu, and Y. Pei, "Collaborative service selection via ensemble learning in mixed mobile network environments," *Entropy*, vol. 19, no. 7, p. 358, 2017.

[37] S. Pang, T. Ding, S. Qiao, F. Meng, S. Wang, P. Li, and X. Wang, "A novel YOLOV3-arch model for identifying cholelithiasis and classifying gallstones on CT images," *PLoS ONE*, vol. 14, no. 6, 2019, Art. no. e0217647.

[38] Y. Yin, W. Xu, Y. Xu, H. Li, and L. Yu, "Collaborative QoS prediction for mobile service with data filtering and SlopeOne model," *Mobile Inf. Syst.*, vol. 2017, Jun. 2017, Art. no. 7356213.

[39] H. Gao, Y. Duan, H. Miao, and Y. Yin, "An approach to data consistency checking for the dynamic replacement of service process," *IEEE Access*, vol. 5, pp. 11700–11711, 2017.

[40] S. Pang, S. Qiao, T. Song, J. Zhao, and P. Zheng, "An improved convolutional network architecture based on residual modeling for person re-identification in edge computing," *IEEE Access*, vol. 7, pp. 106749–106760, 2019.

[41] Y. Yin, F. Yu, Y. Xu, L. Yu, and J. Mu, "Network location-aware service recommendation with random walk in cyber-physical systems," *Sensors*, vol. 17, no. 9, p. 2059, 2017.

[42] H. Gao, W. Huang, and X. Yang, "Applying probabilistic model checking to path planning in an intelligent transportation system using mobility trajectories and their statistical data," *Intell. Automat. Soft Comput.*, vol. 25, no. 3, pp. 547–559, Jan. 2019.

[43] S. Pang, Q. Gao, T. Liu, H. He, G. Xu, and K. Liang, "A behavior based trustworthy service composition discovery approach in cloud environment," *IEEE Access*, vol. 7, pp. 56492–56503, 2019.

[44] Y. Yin, S. Aihua, G. Min, X. Yueshen, and W. Shuoping, "QoS prediction for Web service recommendation with network location-aware neighbor selection," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 26, no. 4, pp. 611–632, 2016.

[45] H. Gao, W. Huang, Y. Duan, X. Yang, and Q. Zou, "Research on cost-driven services composition in an uncertain environment," *J. Internet Technol.*, vol. 20, no. 3, pp. 755–769, 2019.