Computers and Electrical Engineering 73 (2019) 227–244

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Energy-Efficient virtual machine selection based on resource ranking and utilization factor approach in cloud computing for IoT[☆]

Mahammad Shareef Mekala[a], P Viswanathan[b,*]

[a] School of Computer Science and Engineering, VIT University, Vellore, 632014, India
[b] School of Information Technology and Engineering, VIT University, Vellore, 632014, India

ARTICLE INFO

Article history:
Received 21 June 2018
Revised 21 November 2018
Accepted 29 November 2018

Keywords:
Cloud computing
Resource requirement rate
CRB Ranking
Processing element cost
Task categorization
VM Migration

ABSTRACT

IoT leads to abrupt variations producing an immense number of data streams for storage, which is a considerable task in the heterogeneous cloud computing environment. Extant techniques consider task deadlines for virtual machine (VM) allocation and migration. This creates a resource famine leading to haphazard and numerous VM migrations, high energy consumption and unbalanced resource utilization. To solve this issue, an energy-efficient resource ranking and utilization factor-based virtual machine selection (ERVS) approach is proposed. ERVS encompasses the resource requirement rate for task classification, comprehensive resource balance ranking, processing element cost and the resource utilization square model for migration. It evaluates overloaded and underloaded hosts and types of VM by predicting CPU utilization rate and energy consumption. Based on this, tasks are sorted and VMs are optimally assigned, which enhances the resource utilization rate, reducing the number of live VM migrations. The experiments evaluate the ability of the proposed approach to diminish energy consumption without violation of service level agreements.

1. Introduction

IoT is regarded as a new epoch for green IT computational fields. It enables a cloud-based computer system to manipulate things remotely using sensor devices. The deployed sensors gather environmental data, which are analysed to determine suitable actions [1]. Smart agriculture, transportation, cities, grids and healthcare, and novel inventory system applications, all use IoT technology. According to a statement from IBM, each day approximately 2.5 exabytes of data are produced from sensor devices, and in the year 2020, nearly 49 billion devices will be linked [2]. A suitable infrastructure and platforms are required to accumulate and process the sensor data. For instance, in a smart agriculture environment, soil and weather data are measured and stored in the cloud. The computational analysis of stored data is carried out based on an optimal threshold value, which is used to regulate the production of agriculture yield, and the analysed reports are sent to farmers or end users to enable them to take accurate action for preserving the loss [3,4]. A fertilizer recommendation is made based on the computational analysis of weather predictions and soil nutrition parameters.

☆ This paper is for CAEE special section SI-scss. Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. P. Pandian.
* Corresponding author.
E-mail addresses: mekalashareef@gmail.com (M.S. Mekala), viswatry2003@gmail.com (P. Viswanathan).

https://doi.org/10.1016/j.compeleceng.2018.11.021
0045-7906/© 2018 Elsevier Ltd. All rights reserved.

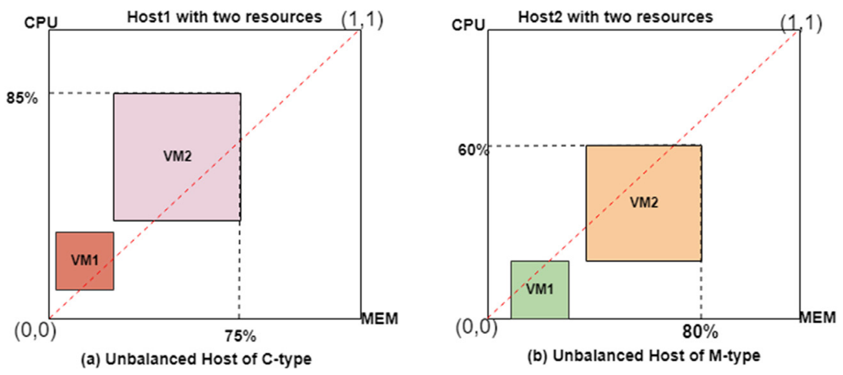


Fig. 1. Unbalanced resources of virtual machines on host.

Cloud computing is an emerging technology which plays a vital role in promoting the rapid development of IoT and it provides adaptable, extensible services to its subscribers based on a pay/utilization model [5]. Cloud users are free from limitations on obtaining cloud services and have no obligation to offer a standard configuration of hardware or to install particular software systems. They can freely access IaaS services for saving sensed data, PaaS for running the computing process and SaaS software to analyse IoT data over the Internet [6]. The computing storehouse provided by the cloud allows for many inventive IoT applications, such as the declaration of invented systems and IoT intelligent optimization, which serves to remotely control systems, agricultural areas and manufacturing industries.

Virtualization technology offers an accurate method of reallocating physical resources between different physical nodes in a cloud data centre [7]. It formulates normalized VMs on physical nodes. The jobs/tasks are concurrently submitted to different VMs to run on the same physical machine (PM). Furthermore, live migration is employed to aggregate VMs, in order to diminish the number of active PMs.

The cloud stores the sensed data for processing and computational analysis based on a standard threshold value. The type of task is categorized based on the computation process and assigned to a suitable VM in the cloud. However, a sudden flow of data occurs due to fluctuations in the sensing area. The cloud system also allows certain variations to select a suitable VM for the live migration process. The abrupt change in the sensing environment may lead to unavailability of resources, known as resource famine or starvation. In such cases, the VMs are migrated from one physical node to another. This leads to a greater number of VM migrations, causing resource wastage, higher energy consumption and intra-VM interference. Hence, we introduce an optimally energy-efficient VM selection and migration approach for IoT in the cloud environment. The principal aspiration of the technique is to provide an uninterrupted IoT service environment without violation of service level agreements (SLAs).

IoT computational jobs/tasks are categorized and scheduled to the VMs of the PMs in the cloud. If a resource famine is identified on a particular VM of a host, then that VM is migrated to the destination server based on the comprehensive resource balance (CRB) ranking scheme, processing element cost (PEC) function, energy consumption model and resource utilization square model.

Resource categorization is another important problem in selecting a suitable VM, referred to as virtual machine consolidation. It is illustrated in Fig. 1. The physical resources, (e.g., CPU or memory) of the host are distributed differently based on the requests of different VMs [8]. Sometimes, the PMs are unable to allocate resources, due to unbalanced resource management, resulting in wastage of resources. Fig. 1(a) shows that if a PM has less memory utilization than CPU utilization, then this leads to memory wastage. Hence, the PM cannot allocate the memory resource to other VMs. Fig. 1(b) shows the wastage of CPU resource due to high memory resource unavailability. Therefore, the unwanted utilization of PM resources can be reduced, based on the accurate selection of VMs, referred to as VM consolidation.

In [9], a framework is introduced to solve the heterogeneity problem which uses a mathematical model for predicting the performance of a system and the job migration of multiple VMs in clusters. It operates based on predefining two threshold values: a high threshold t_h and a low threshold t_l. If the PM resource utilization exceeds t_h, then the system will migrate some VMs to another PM for hotspot avoidance. Conversely, if the PM resource utilization falls below t_l, then the system will relocate all of the VMs of the PM to another targeted PM, for better performance and energy saving. In dynamic cloud systems, the central manager is responsible for defining these two thresholds and managing the information of all PMs and VMs. In [10], the live migration (LM) feature is utilized to migrate the task/job from one sub-cluster to another and to examine the migration time and required parameters using Xen 3.3.0 simulations. In [11], the MapReduce algorithm is utilized to mine the large volumes of geospatial data using high-performance computing resources.

Our principal goal is to manipulate the task classification, task assignment issues, VM selection and allocation mechanism, to reduce the number of active PMs. This results in accurate service provisioning, maximization of resource utilization and a reduction in the number of VM migrations (VMMs) while ensuring achievement of SLA guarantees. Hence, we introduce an ERVS approach by taking into consideration the CRB ranking scheme, PEC function, energy consumption model and resource utilization square model. This solves the above-mentioned difficulties by improving resource utilization of PMs and reducing the energy consumption of data centres. Therefore, the research contribution of accurate VM migration during unexpected rises in workload, and the principal benefits of this research, are as listed below.

1. An adaptive resource utilization square model is proposed to direct the adjustment of unbalanced hosts and the selection of VMs to enhance resource utilization and reduce the active number of PMs.
2. The CRB ranking scheme is streamlined based on the resource utilization rate and the energy consumption of the host, to assess the number of overloaded hosts.
3. A PEC function is formulated based on the CPU resource utilization rate, to find underloaded nodes, in order to reduce the quantity of live VM migrations.
4. A task classification method is proposed based on the resource requirement rate for classifying the types of task.

The rest of the paper is organized as follows. In Section 2, related work is reviewed and the difficulties of existing systems addressed. The problem formulation is discussed in Section 3. The solution algorithm is designed in Sections 4 and 5. Finally, the experimental results are evaluated in Sections 6 and 7 presents the conclusions, together with suggestions for future research directions.

2. Related work

Virtual machine deployment is an influential principal research problem in server virtualization applications. The energy dissipation and increasing size of data centres represent an outstanding challenge to researchers [12]. Many studies have been performed on resource integration and on reusing the facilities of physical devices to accomplish the best control of resource usage and reduced energy expenditure.

In [13], an SLA-aware resource scheduling technique was adopted to combine resource requirement prediction and the VM resource utilization process via a regression vector approach called the support vector regression approach. This was adapted to assess the basic resource provision. The author proposed a genetic algorithm for placing VMs based on a rearrangement pattern, but the predicted QoS was not accomplished using this approach. For instance, if a node becomes consistent, then the VM migration should have fewer SLA violation cross points and lower energy consumption during server consolidation. In [14], the variable item-size bin packing algorithm was developed based on a variable resource requirement and used to segregate VMs and PMs in a correct manner and also to reduce the number of active hosts, but the performance of the application has not been examined.

The work in [15,16] described heuristics-based server consolidation approaches such as the RF-aware algorithm and the Sercon algorithm, for VM migration. These two approaches were used to reduce resource fragmentation processes based on resource utilization. In the Sercon algorithm, the resource defragmentation phase plays a vital role in defeating the process of fragmentation of resources by executing the PM compression phase. RF-aware doubles the estimation of VM migrations and defeats the process of fragmentation of resources more successfully than the Sercon approach. However, this study did not consider the migration time and energy consumption.

In [17], a heuristics-based algorithm was adopted based on fit decreasing and the harmonic cardinality constraint approach for successful placement of VMs. However, this is not applicable for a dynamic workload. It is also unable to achieve QoS due to the process of live VM migrations under the policies of dynamic resource provisioning for VM consolidation. Therefore, the physical node energy consumption is nearly 45% and the data centre network load is 15%. Whenever CPU utilization is a maximum, 70% of resources such as memory and disks are wasted, due to the unavailability of required resources, leading to high energy consumption. Furthermore, there is no appropriate mechanism to maintain the utilization of unbalanced resources. In [18], resource utilization based on an ant colony algorithm to solve the VM resource balancing issue is summarized. However, resource starvation and the task resource requirement analysis problem of VM migration were not taken into consideration.

In [19], an energy consumption model was introduced based on the association of energy consumption, resource utilization rate and CPU processing performance before and after VM consolidation. This model considers only CPU resource utilization. It does not estimate the energy consumption of other resources (such as the bandwidth when the CPU is idle) and it ignores certain cases (such as when the tasks need to claim high bandwidth with low CPU requirement).

In [20], the swarm algorithm was introduced to evaluate the degree of load balance and also to sort the variance of each resource utilization rate of the node. The core idea is to group resources of the same degree. First, the swarm randomly segregates groups and subgroup spaces, and also the parameters are randomly assigned to each subgroup to avoid premature results. The fitness function is defined as the sum of resource utilization weights, which does not fully reveal the load balancing index.

In [21], various techniques are used to deal with unexpected spikes in the cloud data centre, leading to resource unavailability. This approach does not consider basic parameters to identify the respective destination host. Therefore, the execution time and energy consumption of VM migration does not satisfy the threshold value. In [22], the author discusses the difficulties that occur in the processes of live migration techniques. These cause low performance and impact on processing time. In [23,24], various live VM migration techniques were adopted based on resource utilization for load balancing. This study considers the task deadline parameter but not the energy consumption of VM migration. Therefore, achieving an accurate VM migration with shorter execution time, fewer SLA violations and lower energy consumption is a challenging task.

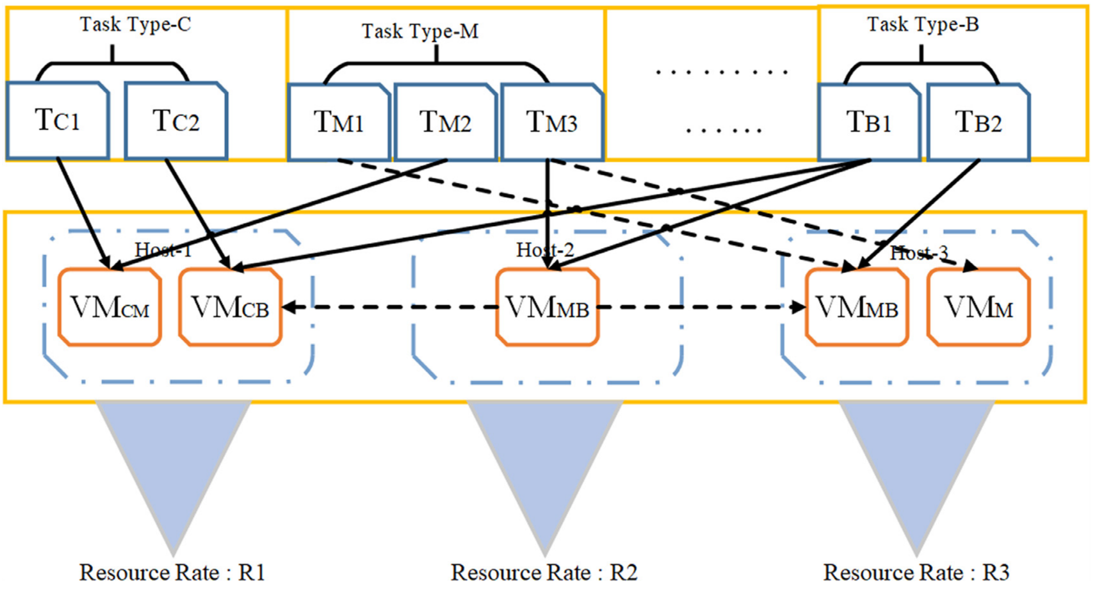


Fig. 2. Cloud system model.

Hence, in our proposed technique, we consider CPU, memory utilization rate and energy consumption of the virtual machine required to accommodate the sudden resource requirements of IoT applications. The CRB ranking scheme, PEC function, energy consumption model and resource utilization square model values are used to finalize the overloaded and underloaded hosts for selection of suitable VMs. Successful migration is accomplished based on the identification of overloaded hosts for migrating the respective VMs to a suitable destination server to maintain resource balance.

3. Problem formulation

In the cloud computing framework, optimization of the VM migration process is an important issue in the server consolidation mechanism. There are two issues in server consolidation: (1) task categorization based on resource requirement and (2) assigning tasks to suitable VMs. Similarly, there are two sub-issues: (a) selecting VMs which have been relocated from the highly loaded hosts (VM selection problem) and (b) setting these VMs to the targeted hosts (VM placement problem). If a task is assigned without leveraging functional requirements such as resource requirement capacity, selection of suitable VMs (underloaded) leads to enhancing the latency, energy consumption and number of live VMMs. The VM selection and allocation problems are defined as evaluating the effects of resource capacity, resource utilization and energy consumption of VMs on the respective host before submission of the task. A possible VM selection and allocation method is described as a mapping of PM to a set of VMs, such that $\sum_{j=1}^{VM_j}\sum_{m=1}^{h_m} VM^r_{jm} = \{VM^r_{11}, VM^r_{21}, VM^r_{31} \cdots VM^r_{jm}\}$, which must meet the following two conditions.

1. Each VM is allotted to at least one PM, and no VM is allotted to more than one PM, i.e., $\left.\begin{array}{l}\cup_{VM_j\in h_m} VM^r_j(h_m) = VM^r_{j,m},\\ VM^r_{i=1,m=1}\cap VM^r_{j=1,m=2}=\emptyset\end{array}\right\}\forall 1\le i, j\,\&i\ne j$
2. For every PM, the cumulative resource requirements of its hosted VMs do not surpass its available resources, i.e., $ToT^X_{j,m} \le \sum_{j=1}^{VM_j} VM^r_{j,m}(r_c) \le h_m(r_c)\ \forall 1\le j \le m$

In the above, X refers to a type of task and r_c refers to the capacity of the resource. The task resource requirement rate (RRR) value should be less than the capacity of the VM and the resource capacity of the host.

Fig. 2 demonstrates the task categorization method based on the resource requirement rate. The task set comprises four types of tasks based on CPU, memory, I/O and bandwidth using the RRR value. The sorted task type is updated and maintained based on the RRR value and the task assigned to the respective type of VM, e.g., (VM_C, VM_{CM}, $VM_{CB} or VM_{MB}$). The CRB ranking scheme is used to sort out the overloaded hosts and the PEC function is partially used to find the targeted host for an accurate VMM process.

The proposed approach classifies the task based on the requirements for CPU, memory, bandwidth and communication resources, using the resource requirement rate function. This accurately allocates the task to the appropriate VM, which reduces latency and energy consumption. The VM selection process is based on the evaluation of highly loaded hosts using the CRB ranking scheme and of underloaded hosts based on the PEC function, which ensures that VM migration is accurate. This is shown in algorithm 3. Hence, all these methods accurately predict a suitable VM for the task, which leads to low

Table 1
Table of notations.

Symbol	Meaning
ξ_i^{jm}	Energy of ith task on jth VM of mth host/node
F_{ijm}^d	Required time to transfer d size of file for executing ith task on jth VM of mth host
IT_m	idle time of mth host
δ_m	Resource utilization rate of mth host
UBM-type	Unbalanced memory resource type
BRU-type	Balanced resource utilization type
UBC-type	Unbalanced CPU resource type
$\delta_i^{mips}(t)$	CPU utilization (milli-seconds) at time interval t
VMM	Virtual Machine Migration
VM_{im} and VM_{jm}	No VM should be allotted to more than one PM ($i \neq j$)

latency and reduces the number of live VM migrations, resulting in a reduction in server energy consumption in the cloud data centre. The VM classification, selection and placement steps are explained in the following sections.

4. System architecture model

The cloud system model shown in Fig. 2 enables the process of task classification and allocation of the task to the proper virtual machine of the host. This mechanism is formulated based on the evaluation of resource utilization rate and the energy consumption of the host. Initially, the task is received from the working platform with the resource elements requirement. Based on these elements, the tasks are sorted into various types and these are updated in a type-of-task (ToT) set. The ToT set comprises four types of tasks (task type X where X = C, M, Co or B). If the task RRR value is equivalent to the threshold value of {MIPS:memory:I/O:bandwidth}, then X={C:M:Co:B} respectively. Therefore, the sorted task is allocated to the fitted VM of the host. The VM migration is carried out by classifying the hosts into two types, i.e., hosts with low utilization (underloaded) or high utilization (overloaded). To accomplish this process, the energy consumption and resource utilization of the host must be considered.

4.1. Energy consumption model

We have considered m independent heterogeneous hosts in a cloud system $H = \{h_1 + h_2 + h_3 + \ldots\ldots + h_m\}$. It includes a set of heterogeneous virtual machines on each host $Vj = 1 tom = \{V_{j1}^r + V_{j2}^r + V_{j3}^r + \ldots\ldots + V_{mn}^r\}$. Here, V_{mn}^r refers to the $r = \{1, 2, 3, 4, 5, 6\}$ *type* of the m^{th} host and n^{th} virtual machine. This means that $r = 1$ refers to the C-type VM, $r = 2$ refers to the M-type VM, $r = 3$ refers to the I/O-type VM, $r = 4$ refers to the Co-type VM, $r = 5$ refers to the B-type VM and $r = 6$ refers to the UB-type VM for that particular host. Let $T = \{t_1, t_2, t_3, \ldots\ldots, t_q\}$, q be the number of heterogeneous service requests (tasks), where each task t_i has a service length L_i of a million instructions (MI). The revised equation parameters are listed in Table 1.

Definition$_1$: The energy consumption of each task $t_{i,} 1 \le t_i \le n$ is represented by ξ_i^{jm}, i.e., the energy consumed by the i^{th} task on the j^{th} VM of the m^{th} host, is evaluated by adding the energy consumed in data transfer (ξ_{ijm}^T) and the energy consumed in the task execution (ξ_{ijm}^P), using Eq. (1).

$$\xi_i^{jm} = \xi_{ijm}^P + \xi_{ijm}^T \tag{1}$$

For executing the i^{th} task, supportable files are transferred, which consumes energy calculated using the file transfer time and the average data transfer power consumption (ξ_{avg}), based on a product-of-sum (PoS) form. Let the energy consumed by the m^{th} active host be denoted by ξ_{act}^m. Hence, the average energy consumed by the m^{th} host (ξ_{avg}) in the data centre can be assessed as $\zeta_{avg}^m = \sum_{h_1}^{h_m} \xi_{act}$. The energy consumption for the required data transfer (ξ_{ijm}^T) is defined in Eq. (2).

$$\xi_{ijm}^T = \left[\prod_{d=1}^{D} F_{ijm}^d \right] \times \zeta_{avg}^m \tag{2}$$

The energy utilized for processing the i^{th} task on the j^{th} VM (r-type) of the m^{th} node is evaluated via the task execution time (ψ_{ijm}^r) and the average energy utilization of the j^{th} VM (r-type) of the m^{th} host, i.e., ξ_{jm}^r is based on the PoS form. Hence, the energy consumed due to the execution of task (ξ_{ijm}^P) is evaluated using Eq. (3).

$$\xi_{ijm}^P = \psi_{ijm}^r \times \xi_{jm}^r \tag{3}$$

Each VM's energy utilization is assessed using the SoP form, i.e., the summation of the energy utilizations of all the same type of VMs for the respective host for every assigned task. This is represented as ξ_{jm}^r, $\{r = 1, 2, 3, 4, 5\}$ and is illustrated in

Eq. (4).

$$\xi_{jm}^{r} = \sum_{i=1}^{q} \{\psi_{ijm}^{r} \times Tempp_{ijm}\} \tag{4}$$

Temp is defined as the temporary task allocation matrix for all tasks on the various VMs of each host, and is computed based on Eq. (5). It is used as part of the assessment of the energy consumption of the host, i.e., the tasks are filtered by deadline time and resource requirement rate for assigning tasks to VMs for execution. Therefore, energy is consumed by the host. Other than this, there is no energy consumption while a task is waiting in the queue. Each host has a Temp matrix, expressed as follows.

$$Temp = \begin{cases} 0, & If\ t_i\, is\, not\, assigned\, to\, j^{th}\, VM\, of\, m^{th}\, host \\ 1, & If\ t_i\, is\, assigned\, to\, j^{th}\, VM\, of\, m^{th}\, host \end{cases} \tag{5}$$

The physical node energy utilization is defined as the aggregate of the energy utilization rate of all VMs of the respective host (when it is in an active state) and the energy utilization of the host in an unused state. The energy utilized by the m^{th} physical node or host is described as ξ_m and estimated using Eq. (6).

$$\xi_m = IT_m \times \xi_I + \sum_{j=1}^{VM_j} \xi_{jm}^{r} \tag{6}$$

Here, ξ_I is the energy utilized by the m^{th} node in an unused state and IT_m is the idle time (in milliseconds) of the m^{th} node, which is given in Eq. (7).

$$IT_m = Makespan - \sum_{m=1}^{h_m} \sum_{j=1}^{VM_j} \sum_{i=1}^{q} \psi_{ijm}^{r} \tag{7}$$

The system makespan is evaluated as the maximum time needed for a physical node to execute every input job of the system, as illustrated in Eq. (8). Specifically, this is the maximum time for which the physical node is in an active state.

$$Makespan = Max\left\{ \sum_{j=1}^{VM_j} \sum_{i=1}^{q} \psi_{ijm}^{r} \right\} ; \forall\, 1 \le j, m \le h_m \tag{8}$$

The entire energy utilization of the system (data centre or server) is the sum of the energy utilized by all the nodes, and is evaluated using Eq. (9).

$$\xi_S = \sum_{m=1}^{h_m} \xi_m \tag{9}$$

The aspiration is to decrease the energy utilization of the data centre. The energy utilization of the m^{th} node is shown in Eq. (10).

$$Minimize\{\xi_S\} \tag{10}$$

4.2. Resource utilization square model

Each physical machine, host or physical node is associated with three types of resources, i.e., memory, bandwidth and CPU. The term res, $\in$ (ram, bw, mips), refers to memory utilization, bandwidth requirement and CPU utilization respectively. For reference, TR represents the total resources and AR indicates the available resources. For instance, the total CPU resources of the m^{th} physical machine h_m are denoted TR_{mips}^{m}, while the available CPU resources of the host are AR_{mips}^{m}. Similarly, the total and available memory resources of the j^{th} VM are TR_{mips}^{j} and AR_{mips}^{j} respectively. When evaluating the proposed algorithm, the resource utilization rate (δ) of all the resources mentioned above is taken into consideration. The resource utilization rate (δ) is described as the ratio of the resources utilized by the VMs and the total resources of the physical node. It is illustrated in Eq. (11).

$$\delta_{mips} = \sum_{VM_j^m}^{H_m} \frac{TR_{mips}^{j}}{TR_{mips}^{m}} \tag{11}$$

The average resource utilization rate (δ) of the m^{th} physical machine is defined in Eq. (12).

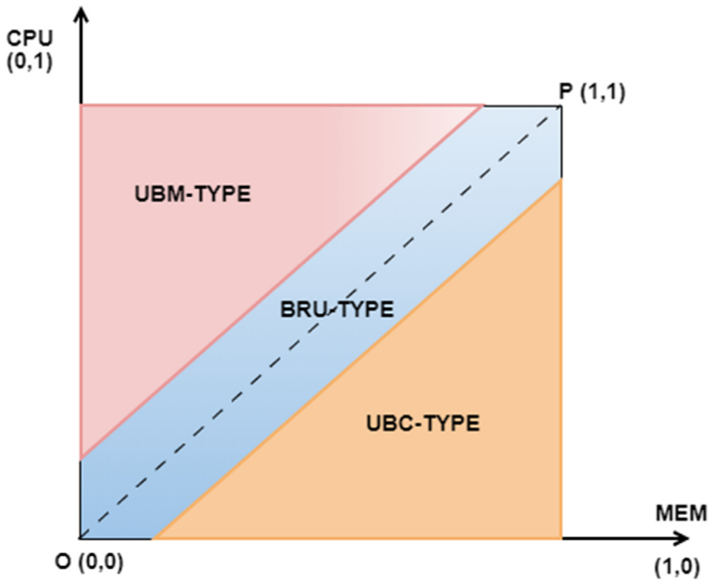


Fig. 3. Resource utilization square model.

$$\delta_m = 1 - \frac{\sqrt{2}}{2} \times \left(\sqrt{\frac{\sum_{res \in \{C,M,B\}} (1 - \delta_{mips})^2}{3}} \right) \tag{12}$$

If the VM resource requirement does not satisfy the threshold resource rate of a node, then the associated physical node becomes unavailable for migrating VMs. The resources of this unavailable host are simply wasted during its lifetime. These nodes have a negative effect on energy consumption in the cloud data centre. Therefore, to enhance the physical node's resource utilization, we introduced a square model based on utilization of a resource. It describes the status of the resource utilization rate of active PMs. This is used to distinguish new deployments for processing the migration of VMs.

In Fig. 3, the x-coordinate and y-coordinate values represent the status of different physical node resource utilizations, based on their resource utilization rate (δ). In the proposed resource utilization square model, the hosts are categorized into three types, based on resource (CPU and memory) utilization rate (δ). The unused physical node considered as the origin is shown by the point O. The point P, positioned at (1.0, 1.0) signifies fully balanced nodes with the best utilization of resources (memory and CPU). These hosts are called BRU-type region hosts, based on Eq. (13). These hosts are fully balanced with accurate resource utilization, which leads to lower energy consumption and high performance.

$$f(P(x,y)) = \begin{cases} BRU - type \ \ if \ (1-x) \le \delta_m^M \&\& (1-y) \le \delta_m^C \\ UBM - type \ \ elsif \ \ (1-x) > \delta_m^M \&\& (1-y) < \delta_m^C \\ UBC - type \ \ \ else \ \ (1-x) < \delta_m^M \&\& (1-y) > \delta_m^C \end{cases} \tag{13}$$

The remaining UBC-type and UBM-type region hosts are unbalanced, due to a lack of CPU and memory resource utilization respectively. Therefore, due to insufficient resources, the host is unable to accommodate the resource requirements. Some hosts in these two regions are able to accommodate the retrograde type of VMs, to enhance the rate of resource utilization and diminish resource wastage.

4.3. Comprehensive resource balance (CRB) ranking scheme

The lower-ranked physical nodes are treated as hosts with lower resource utilization. Therefore, these hosts can be shut down to reduce the number of migrations, which leads to preserving energy utilization. As shown in the investigations, the less-loaded hosts enable the given task to be accomplished using fewer VMs.

$$CRB\,rank = \frac{\left(1 - \frac{\sqrt{2}}{2} \cdot \delta_C\right) + \left(1 - \frac{\sqrt{2}}{2} \cdot \delta_M\right) + \xi_m}{3} \tag{14}$$

The CRB ranking scheme is defined as the mean value of the three parameters CPU (δ_C), memory (δ_M) and energy consumption (ξ_m) of the physical host and is calculated using Eq. (14). The ranks are sorted in declining order, i.e., hosts with high rank values are treated as highly loaded hosts.

4.4. Selection of optimal threshold value (δ^{res}_{Thr})

Evaluating the primary threshold value (δ^{res}_{Thr}) of a discriminating resource in a workload environment is a difficult task. The primary threshold value is formulated in Eq. (15). In our approach, δ^{res}_{Thr} is evaluated using the adaptive median absolute deviation (AMAD) technique.

$$\delta^{res}_{Thr} = median + w \times AMAD_{res} \quad res \in \{C, M, B, I\} \tag{15}$$

where w is a balancing factor with a value of 2. The adaptive median absolute deviation (AMAD) technique is used to evaluate the primary threshold value of the particular resource using Eq. (16), and the resource utilization rate δ_{res} is assessed using Eq. (11).

$$AMAD_{res} = median(|\delta_{res} - median(\delta_{res})|) \tag{16}$$

5. Proposed algorithm

We developed the ERVS approach in cloud computing for IoT to diminish the energy utilization rate and migration count systematically, to ensure consolidation of a server. The given solution is segregated into three parts: task categorization for accurate VM assignment, selection of highly utilized (overloaded) and less-utilized (underloaded) nodes and deployment of VMs. Initially, this approach identifies a number of active hosts and highly utilized (overloaded) nodes based on CRB rank. The PEC function is manipulated to identify the nodes with lower utilization which can be switched off in order to preserve energy utilization. Ultimately, the deployment of VMs part of the process is used to identify the conventional targeted host for performing migration of VMs. These three parts are linked together to constitute resource scheduling based on the type of task, for enhancing the utilization rate of the respective resources and diminishing energy utilization of servers in a data centre.

5.1. Task categorization for assigning accurate VM

Resource famine occurs due to violating the barring capacity (CPU, memory resources) of the VM, and imbalanced resource utilization initializes the VMM process. Haphazard task allocation leads to resource famine. Therefore, the ERVS approach initially evaluates the task resource requirement rate. Based on this rate, tasks are classified. Therefore, the task can be distributed to the appropriate VM by considering the resource utilization rate and bearing capacity. Hence, the scalability of VMM is minimized, leading to a smaller number of live VMMs. During the VMM process, VM selection and placement is a significant task. The VM selection process takes place by sorting out the highly loaded hosts with the help of the CRB ranking scheme. The process considers resource utilization and the energy consumption of the host. The VM placement process is carried out taking the PEC value into consideration. Using these two methods, the VMM process is carried out without any delay in progress and it directly enhances the performance of the server and preserves energy consumption better than the existing systems.

The Algorithm 1 is used to allocate a VM accurately by taking into consideration a limited type of task (q is a set of tasks, $\therefore \forall q \in$ T) with D deadlines. The VM types are classified based on the potential value of a resource. Lines [2–5] are used to sort the tasks in rising order, based on deadlines. The *RemoveMin* function is used to exclude the task from the array list and update the STQ (Sorted Task Queue). Line [6] groups the tasks into four levels. Lines [7–10] identify an appropriate VM type based on the task type and the resource requirement of the task. Lines [11–16] perform the sorting of the hosts based on the VM type.

Algorithm 2 represents the process of task categorization based on resource requirements. It is accomplished by considering the resource requirement parameters of the i^{th} task (RT$_i$ = L$_i$, D_i, M_i, I_i, CO_i). Here, L$_i$ represents length, expressed in millions of instructions per second (MIPS), D_i is the deadline of the task, expressed in seconds and M_i is the memory requirement expressed in gigabytes (GB). I_i is the input/output (IO) requirement of the task. CO_i is the bandwidth required for the task, expressed in Mbps. The tasks are classified as type of task [ToT] = {C-type,M-type,I/O-type,CO-type} tasks. Lines [3–6] are used to measure the RRR value of the task based on all task parameters. Here, $\beta_{C_i}, \beta_{M_i}, \beta_{I_i} and \beta_{CO_i}$ values should be in the range {0, 1}. The sum of all variable values is equivalent to 1, i.e., $\beta_{C_i} + \beta_{M_i} + \beta_{I_i} + \beta_{CO_i} = 1$. The predicted resource weight function ϕ_i is evaluated and the above variables are updated by multiplying with ϕ_i, which is applied to determine the resource quantity of the task. Line [7] is used to determine the maximum value of RRR, which plays a vital role in the task classification process. Lines [8, 11, 14 and 18] are used to assess the type of the task, based on satisfying the condition of maximum RRR value for the respective resource value. Lines [21 and 22] update the ToT set, the VM-type set with the finalized tasks and the VMs. Lines [23–27] usually assign the finalized task to the correct type of VM.

Each VM type has a limited number of VM subtypes for drafting the sorted tasks. For example, the (VM$_{C-Type}$) is sorted based on the RRR value. Therefore, it should satisfy the RRR(VM$_{CC-Type}$) $\leq$ RRR(VM$_{CM-Type}$) $\leq \cdots \leq$ RRR(VM$_{CI-Type}$) condition. This phenomenon is equated to all VM-type sets. For instance, if the expected task resource is not accessible in VM$_{CC-Type}$, at that point the searching process is applied to VM$_{CM-Type}$. If the expected task resource is satisfied/coordinated with the VM$_{CM-Type}$, the algorithm halts and returns VM$_{CM-Type}$. This is the best approach to selecting a VM type from among the remaining VMs based on types of tasks.

Algorithm 1: Resource based task assigning to an accurate VM.

```
input : 1. Host set: $H = \{h_1 + h_2 + h_3 + ...... + h_m\}$,
        2. Task set: $T = \{t_1, t_2, t_3, ......, t_q\}$,
        3. Deadline of Task set: $D = \{d_1, d_2, d_3, ......, d_q\}$,
        4. VM type set: $VM_t ype = \{VM_{C-type}, VM_M - type, VM_{I/O} - type, VM_{CO} - type\}$
output: Accurate VM selection
1  $t_i \leftarrow user_i$
2  for $i=1$ to $q$ do
3      STQ[i] $\leftarrow RemoveMin(T_i)$
4  end
5  Update STQ $\leftarrow SortTaskQueue(T, D)$
6  $[Q_C, Q_M, Q_{I/O}, Q_{CO}] \leftarrow TaskCateg$(STQ); Using algorithm-2
7  for each task $t_i \in T$ do
8      vm $\leftarrow selectVM_{Type}(t_i, Type(t_i, T))$ Using algorithm-2
9      Update VMList $\leftarrow$ vm
10 end
11 for each $h_i \in H$ do
12     if vm is fit in HostList $h_i$ then
13         $h_i \leftarrow$ vm
14         update HostList $h_i$
15     end
16 end
17 Return Selected vm $\in$ VMList
```

5.2. Selection of underloaded and overloaded host

This section aims to select underloaded nodes which have fewer VMs. This type of node becomes inactive, once all VMs are aggregated at the destination physical node. All active hosts remain arranged in three types, combined to form a host set. The underloaded host set is updated with the resource utilization rate. This can be carried out with the help of the PEC function value. The PEC value is determined as the ratio of the sum of available and predicted resources to the total resources of the host, as defined in Eq. (17).

$$PEC\ value = \sum_{VM_j^m}^{H_m} \frac{(AR^j_{mips} + PR^j_{mips})}{TR^m_{mips}} \times K,$$
$$\therefore \forall\, VM_j^{m \in H} \in H_m = (VM_1^m + VM_2^m + \cdots + VM_j^m). \tag{17}$$

The PEC value is calculated by considering variations in CPU resources. The memory resource factor is not considered, because the VM allocated memory becomes greater than the potential range of the physical node. This part of the process adopts the host which has the lowest PEC value, considered as an underloaded host. Since this particular node becomes a target node to accommodate migration of all VMs, this leads to shutting down PMs to preserve energy utilization after migrations. If not, the host with the highest PEC value from the host list is elected as a target host to facilitate these VMs; otherwise, if some VMs are unable to migrate to another node, the migration process is cancelled, and this node remains in an active state.

5.3. Placement of migrating VM

The Algorithm 3 deploys the VM to a suitable host. To accomplish a high resource utilization rate, unbalanced nodes need to be shut down in the cloud data centre, making it essential to deploy VMs on suitable physical hosts. This results in live migration, which uses up the resources of the system and creates more SLA violations. The host selected from the host set is the target host leading to fewer migrations and increased service performance. The square model, energy consumption model, PEC function and CRB ranking scheme are used to guide the VM deployment process. Whenever it is necessary for VMs to migrate to another node, the target host initially identified from the host set achieves a high utilization rate of a resource.

In Algorithm 3, lines [1 and 2] are used to sort the host set with lower loads according to the PEC function. Hosts are sorted based on the CRB ranking scheme. Therefore, if the placement of VM impacts on the host, causing it to fall into the BRU type, then the VM is instantly migrated to the node derived in lines [4 and 5]. If not, the host is selected with the lowest CRB rank from the host list after deployment of the VM derived in lines [7–9]. If a target host is still not identified, then the host from the underloaded host set with the highest PEC value is selected to facilitate the VMs and is considered

Algorithm 2: Task categorization for assigning accurate VM.

```
input : 1. Resource based Task Parameters set: $RT_i = \{L_i, D_i, M_i, I_i, CO_i\}$,
        2. Task set: $T = \{t_1, t_2, t_3, \ldots\ldots, t_q\}$,
        3. Optimal capacity of all parameters: CPU:(CLB,CUB), Memory:(MLB,MUB), Deadline of task: (DLB,DUB),
I/O:(ILB,IUB), Bandwidth: (COL,COU), 4. VM$_{sub-types}$= VM$_{C-type}$:{VM$_{CM-Type}$,VM$_{CI-Type}$,VM$_{CCO-Type}$},
        VM$_{M-type}$:{VM$_{MC-Type}$,VM$_{MI-Type}$,VM$_{MCO-Type}$},
        VM$_{I-type}$:{VM$_{IC-Type}$,VM$_{IM-Type}$,VM$_{ICO-Type}$},
        VM$_{CO-type}$:{VM$_{COC-Type}$,VM$_{COM-Type}$,VM$_{COI-Type}$}
output: VM$_{Type}$,Types of the task based on Resource requirementC-type,M-type,I/O-type,CO-type
1  CUB ← CUB/DUB
2  for $t_{i=1} to T_i$ do
3      L$_i$ ← $L_i/D_i$, $\beta_{C_i} \leftarrow L_i/CUB$, $\beta_{M_i} \leftarrow M_i/MUB$, $\beta_{I_i} \leftarrow I_i/IUB$, $\beta_{CO_i} \leftarrow CO_i/COU$.
4      $\phi_i \leftarrow \sqrt{2}/2 \times (\frac{1}{\beta_{C_i}+\beta_{M_i}+\beta_{I_i}+\beta_{CO_i}})$
5      $\beta_{C_i} = \phi_i \times \beta_{C_i}$, $\beta_{M_i} = \phi_i \times \beta_{M_i}$, $\beta_{I_i} = \phi_i \times \beta_{I_i}$, $\beta_{CO_i} = \phi_i \times \beta_{CO_i}$
6      Update all $\{\beta_{C_i}, \beta_{M_i}, \beta_{I_i}, \beta_{CO_i}\}$
7      Max=$\{\beta_{C_i}, \beta_{M_i}, \beta_{I_i}, \beta_{CO_i}\}$
8      if ($\beta_{C_i} = Max$) then
9          The $t_i$ is assign to C-type.
10     else
11         if ($\beta_{M_i} = Max$) then
12             The $t_i$ is assign to M-type.
13         else
14             if ($\beta_{I_i} = Max$) then
15                 The $t_i$ is assign to I-type.
16             end
17         end
18         The $t_i$ is assign to CO-type.
19     end
20 end
21 Update $TypeofTask[ToT] \leftarrow \{C-type, M-type, I/O-type, CO-type\}$
22 VM$_{type}$ ← $VM_{sub-types}$
23 for each $VM_{sub-types}$ and $VM_{ToT} \in VM_{types}$ do
24     if $t_i$ is fit in $VM_{ToT}$ then
25         VM$_{type}$ ← $VM_{ToT}$
26     end
27 end
28 Return Type of Task,$VM_{Type}$
```

as the target host, as seen in lines [13–20]. Finally, lines [21–24] are used to update the node set with the chosen node after deployment of the VM, if it does not relate to the set.

6. Experimental result and performance analysis

6.1. Experimental setup

The investigations are performed using a CloudSim 3.0.3 simulator [25]. Xen is adopted as the VM monitor (hypervisor). The ERVS approach is developed in Java and examined on a Lenovo workstation with an Intel Core i3-5005U @ 4 GHz, 4 GHz CPU and 50GB of memory. CloudSim is a fully customized toolkit, enabling simulation and modelling of real cloud computing systems. It also facilitates an application environment which includes structural and behavioural modelling of servers, VMs and resource provisioning policies. Our experiments were performed on heterogeneous resources, i.e., hosts, VMs and input service requests.

During each experiment, the hosts were fixed and the VMs were varied from 20 to 500. If VM resources are randomly produced, then the aggregate resources of all VMs of the host should be less than the range of that host's resources, i.e., 650 MIPS. Approximately three VMs are administered by each host (five or six hosts). After allotment of tasks or jobs to three VMs, the host is full. Therefore, a VM relocation process is initiated to service the resources. The categorization and configuration of each host are represented in Table 2 and the configuration of VMs is shown in Table 3. To determine the accuracy and response time of the ERVS approach, we used the Sercon algorithm, the initial distribution algorithm and first

Algorithm 3: Process of VM migration.

```
input  : 1. Host list set: $RT_i = \{L_i, D_i, M_i, I_i, CO_i\}$,
         2. underloaded list set 3. VMList
output: Placement of VM migration
1  Sort Underloaded HostList in ascending order based on PEC value
2  Sort HostList in ascending order based on CRB ranking scheme
3  for each $h_i$ from HostList do
4      if f(P(x,y))=BRU-type after accept VM then
5          allocateHost= $h_i$
6      else
7          if host ($h_i$) has lowest CRB rank after vm placement then
8              allocateHost= $h_i$
9          end
10     end
11 end
12 if allocateHost=NULL then
13     for Each host $h_i$ has higher PEC value in underloaded HostList do
14         if host $h_i$ can support vm then
15             allocateHost= $h_i$
16             HostList.add(allocatedHost)
17             UnderloadedHostList.remove(allocatedHost)
18         end
19     end
20 else
21     if allocateHost $\neq$ NULL then
22         migrationSched.put(vm,allocatedHost)
23     end
24 end
25 Return VM migration
```

Table 2
Host configuration.

Resource/HostType	Host1	Host2	Host3	Host4	Host5
MIPS	5000	4500	6000	5500	6500
Cores	1	2	5	3	4
RAM(GB)	24	24	24	24	24
Bandwidth(Mbps)	100,000	100,000	100,000	100,000	100,000
Storage(GB)	50	70	80	80	80

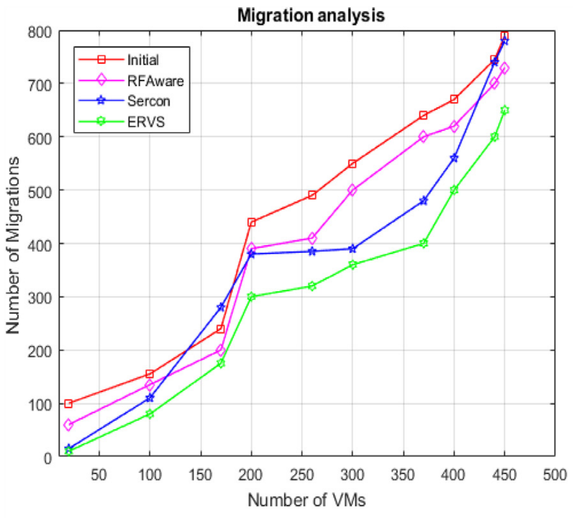
Table 3
VM configuration.

Resource/VMType	VM_{C-type}	VM_{M-type}	$VM_{CM-type}$	VM_{I-type}	$VM_{CO-type}$
MIPS	5000	4500	6000	5500	6500
Cores	1	1	2	2	3
RAM(MB)	512	512	1024	1024	512
Bandwidth(Mbps)	1000	1000	1000	1000	1000
Storage(GB)	5	7	8	8	8

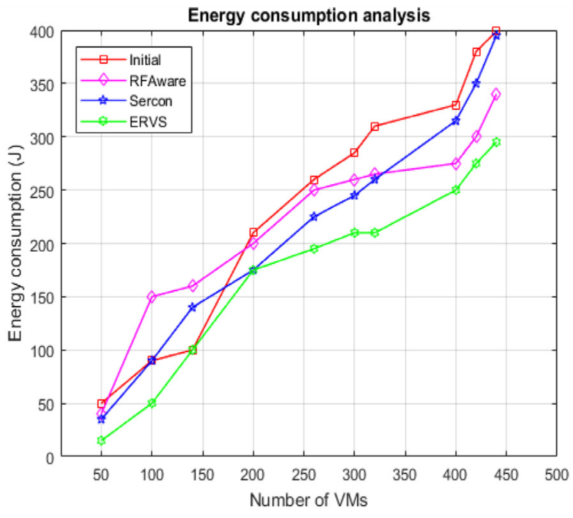
come first serve (FCFS), round-robin, and RF-aware algorithms for resource rescheduling and energy utilization of the system. The results are shown in Table 4, which describes the comparative analysis of energy consumption, VMM, SLA violations (SLAV) and SLA violation time of each active host (SLATAH) for all approaches with 50 to 500 VMs. We concentrated on resource utilization rate $\delta = 0.25, 0.31$, to examine its impact, and the results are illustrated in Figs. 4–7.

Table 4
Comparison of proposed and existing approaches experimental results.

Algorithm	Simulation platform	Energy (kWh)	VMM	SLAV (10^{-5})	SLATAH(%)
Initial	Cloudsim Tool Kit (CTK)	285.05	570	0.9210	0.03549
RFAware	CTK	260.34	515	0.8321	0.03975
Sercon	CTK	245.79	426	0.7456	0.03112
ERVS ($\delta = 0.31$)	CTK	210.14	369	0.4123	0.02559
ERVS ($\delta = 0.25$)	CTK	211.50	375	0.4910	0.02971



(a) Migration analysis



(b) Energy consumption analysis

Fig. 4. Analysis of number of migrations and energy consumption.

6.2. Performance metrics

The metrics described below are used to estimate the accuracy and response time of the algorithms.

1. Energy consumption (EC) is defined in Eq. (18),

$$EC = \sum_{m=1}^{h_m} \int_t \left(\begin{array}{l} \xi_u \cdot Max(\xi_i) + \\ (1-\xi_u) \cdot Max(\xi_i) \cdot \delta_i^{mips}(t) \end{array} \right) dt \tag{18}$$

where m refers to the number of PMs, $\xi_u = 0.7$ refers to the unused server energy consumption, Max(ξ_i) = 250W is the approximate energy utilization of a host in its running state and $\delta_i^{mips}(t)$ refers to the utilization of the CPU at time t. This evaluates the total energy utilization of the cloud data centre.
2. SLAV is defined as a combination of both SLATAH and PDF metrics.

a) SLATAH is the proportion of time during which the active host has 100% CPU utilization and is defined in Eq. (19).

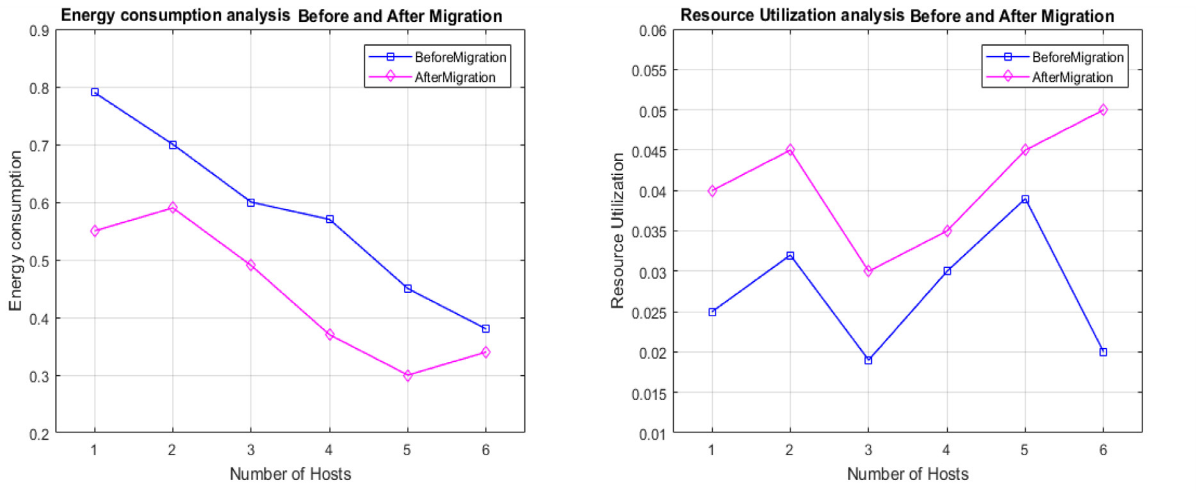
$$SLATAH = \frac{1}{M} \sum_{i=1}^{M} \frac{T_{SLA_i}}{T_{ACT_i}} \tag{19}$$

where M is the number of PMs, T_{SLA_i} is the SLA violation time of the i^{th}PM and T_{ACT_i} is the active time of the i^{th} PM.

b) The performance degradation caused by the migration process (PDM). This is determined using Eq. (20).

$$PDM = \frac{1}{P} \sum_{j=1}^{P} \frac{PD_j}{CD_j} \tag{20}$$

where P is the total number of VMs, PD_j is the determined performance degradation (which can be measured for an extra 10% of CPU utilization) of the j^{th} VM caused by the migrations and CD_j is the total CPU capacity demanded by the j^{th} VM.
Therefore SLAV = SLATAH × PDM.



(a) Energy consumption analysis before and after migration (b) Resource utilization analysis before and after migration

Fig. 5. Analysis of energy consumption and resource utilization.

6.3. Evaluation of energy consumption of VM migration

We recognize that live migration risks having an impact on the execution of the current jobs. The PEC function assesses all underloaded hosts based on the CPU resource utilization rate of all the VMs of each host. The CRB ranking scheme is used to evaluate the rank of every host based on utilization of resources and energy. Thus, the ERVS algorithm uses the minimum number of VM migrations, which is advantageous for energy efficiency in cloud computing. The ERVS estimates the cloud energy utilization based on the consumed energy of all host migrations in a given time period and is illustrated in Fig. 4. It also represents the schedule of all live migrations using three algorithms throughout the simulation. During the RF-aware approach, the number of physical nodes is reduced by adding further migrations to accomplish minimum resource fragmentation. Therefore, this creates a significant enhancement in live migrations. Fig. 4a shows that, among the three approaches, the ERVS approach has achieved fewest live migrations.

Hence, the ERVS approach manages the differences in resource utilization of working VMs when electing underloaded hosts for VM migrations. If the host enables a lower resource prediction rate, then running VMs have to choose underloaded physical nodes. Therefore, the workload disparity has less influence on the target host after migrations. This increases the robustness of the target host and diminishes migrations due to changes in the resource request. The estimation of the energy utilization of a system is carried out using the migration count and is represented in Fig. 4b. In view of this investigation, every energy-preserving technique has to avoid frustrating end users by denying requests for service resources. Otherwise, QoS requirements will not be met. The ERVS approach consumed less energy than the RF-aware and Sercon approaches during the experiments. Therefore, the second and third parts effectively diminish the number of blocked hosts and adjust the nodes with respect to utilization of the resource.

6.4. Comparison of resource utilization between proposed and existing approaches

The correlations among the running VMs of the nodes and migrations of the target host and the resulting energy utilization are shown in Fig. 5. We assessed the energy consumption before migration and after migration of the hosts, as illustrated in Fig. 5a. The correlation regarding VM migrations of nodes and resource utilization before and after migration is represented in Fig. 5b. Greater energy capability can be achieved whenever an energy-efficient VM is added to the same node. If several VMs are processed on the same host, this will enhance migrations and result in low system performance. We have assessed the counts before and after migrations, and their respective energy utilizations are shown in Fig. 5. Fig. 6 reveals the SLAV results for the three approaches. The SLAV metric accommodates an updated pattern to analyse the performance of the remaining approaches with regard to SLA. The exact SLA in a cloud computing (CC) environment is facilitated by a cloud provider. The association among the trends is illustrated in Fig. 6. The energy utilization and SLAV are in inverse proportion.

The ERVS approach consumed less energy than the Sercon and RF-aware approaches in the experiments. The three approaches returned numerous migrations under the consolidation process. Normally, if the hosts are processed with lower resource utilization in the cloud data centre, then the consolidation approach reschedules numerous VMs at the first stage, to reduce the number of active physical nodes. The live migrations influence the changes in resource requirements, which affects SLA violations. The ERVS algorithm is aware of resource utilization, resource request rate and identification of

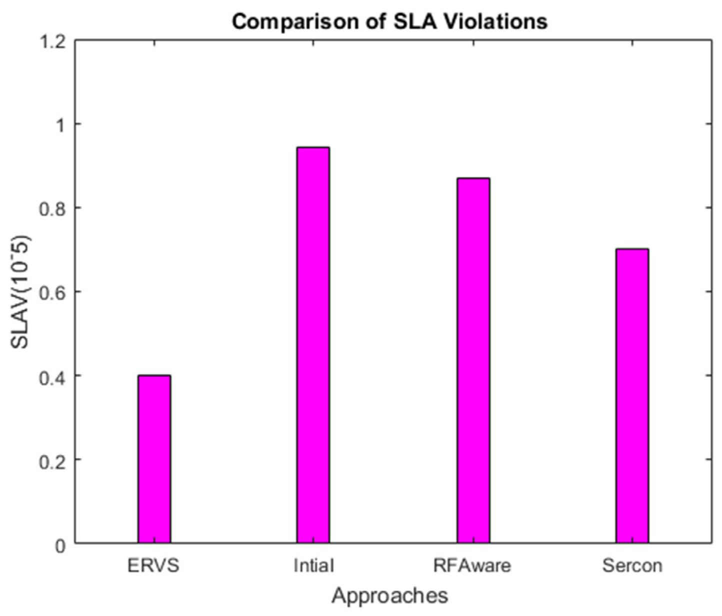


Fig. 6. Comparison analysis of SLA violations between existing and proposed approaches.

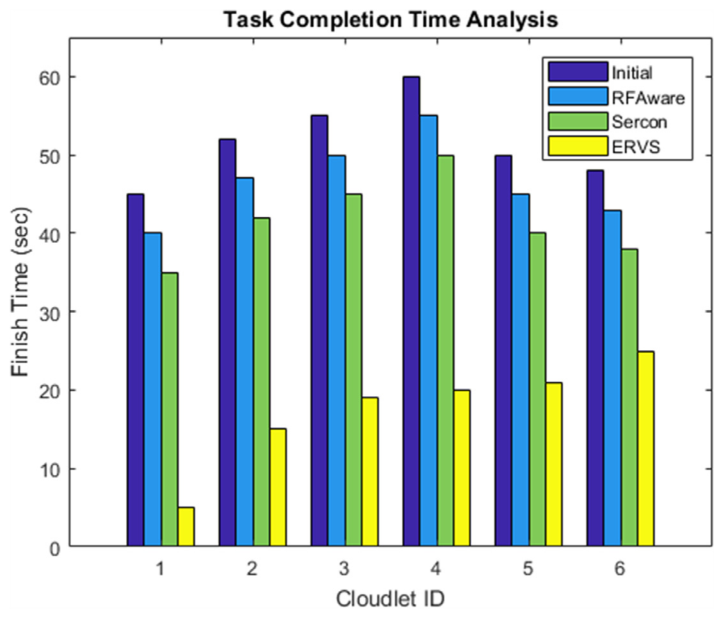


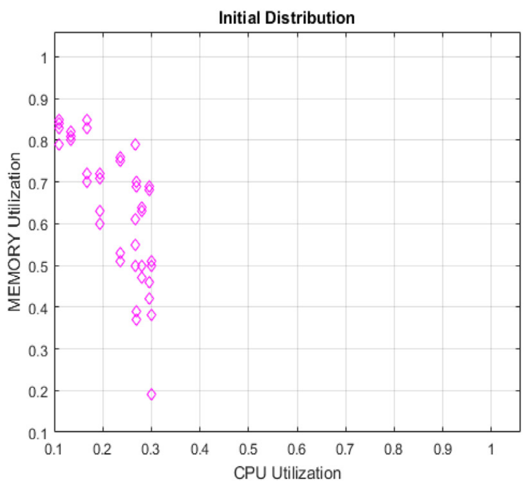
Fig. 7. Comparison analysis of task completion time between existing and proposed approaches.

underloaded hosts, while placing migrations. Compared with the three other algorithms, it schedules the minimum number of migrations to cope with variations in workload, as shown in Fig. 4a.

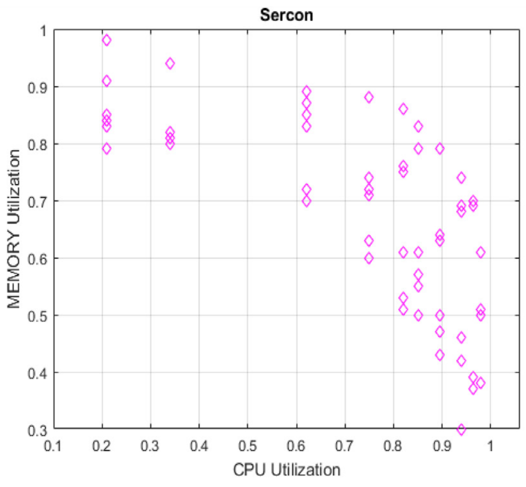
The comparison of task completion times among the proposed and existing systems is illustrated in Fig. 7. The accurate evaluation of task classification is based on resource requests and it perfectly assigns tasks to the appropriate VM. It should be examined together with the energy consumption and resource utilization rate. Hence, the performance of the proposed approach is enhanced, based on the accurate sorting of VMs for a given type of task (ToT), which leads to less time being required to accomplish the task.

Fig. 8 shows the combinations of resource utilizations of a node formed by the four approaches. Each diamond point in the picture symbolizes a node and the (x,y) coordinates signify the memory and CPU utilization of that node.

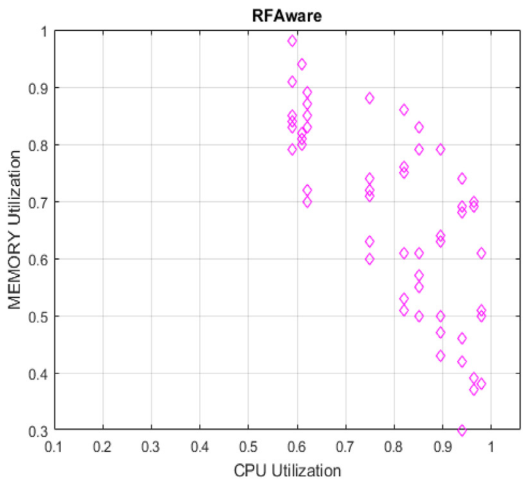
Fig. 8a refers to resource states during the initial distribution of the hosts and also represents the nodes of UBM type. It indicates the nodes with a lack of memory resource. The Sercon approach results are compared and illustrated in Fig. 8b. Fig. 8c shows the analysis for the RF-aware approach. This has comparatively inadequate deployment of hosts and also ad-



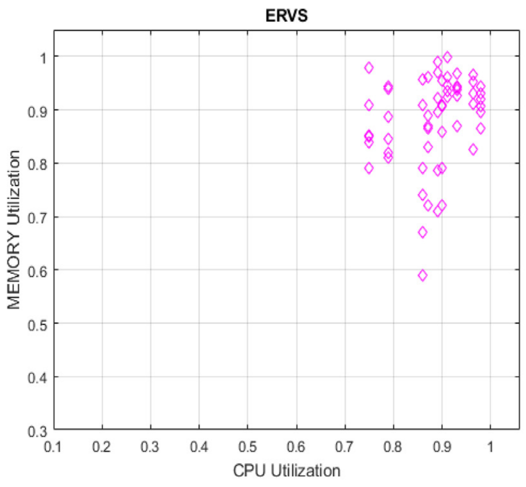
(a) Unbalanced hosts of UBM type due to lack of high CPU resource utilization rate using initial distribution approach



(b) Unbalanced hosts of UBC type and UBM type due to lack of both memory and CPU resource utilization rate, using Sercon approach



(c) Unbalanced hosts of UBC type due to lack of high memory resource utilization rate, using RF-aware approach



(d) Accurately balanced hosts due to high resource utilization rate using ERVS approach

Fig. 8. Analysis of resource utilization rate between proposed and existing approaches.

ditional node adjustments, which causes a greater resource balance rate than the Sercon approach. The majority of nodes have a higher utilization rate, but some nodes still comprise resource fragments that relatively increase the scope of unbalanced resource measures, which enhances energy consumption. Fig. 8d illustrates the status of resource utilization rates of the host for ERVS. Most of the nodes have extremely good resource utilization and balance. The investigations proved that ERVS has adequately reduced the migration count and preserved 70–80% of the energy consumption. Hence, the scope of resource fragments decreases, and the resource utilization of the hosts increases.

7. Conclusion

The proposed comprehensive model reduces resource famine and the energy consumption of the cloud server without affecting the quality of service or service level agreement violations. The predicted resource weight function (ϕ) applied for resource requirement rate (β) improves 95% of successful task categorizations. The categorized task is then sorted and assigned to the virtual machine, resulting in an 87% reduction in task completion time and a 91% increase in resource utilization. The virtual machine selection and placement scheme comprises a comprehensive resource balance ranking scheme to identify overloaded hosts and a processing element cost function for underloaded hosts. The comprehensive resource balance ranking scheme evaluates the individual resource (CPU, memory) utilization rate of active hosts via the resource utilization square model (δ) and host energy consumption, to predict overloaded hosts. The processing element cost function evaluates CPU utilization rate to predict underloaded hosts. Therefore, the virtual machine selection and placement scheme predicts host selection with 90% accuracy for virtual machine consolidation, thereby reducing resource famine by 75.9% in the process of virtual machine migration. The adoption of a suitable target server is accomplished with an energy-efficient resource-based virtual machine selection approach, to migrate the virtual machines from the overloaded server. This also reduces the live migration count, which preserves 70–80% of the energy consumption.

With regard to the future direction of research, we intend to test our approaches in a real-world cloud computing environment. In the case of IoT, the decision can be taken to migrate a portion of the application to the cloud using a fog environment for small intended applications, leading to high performance in local execution.

Acknowledgment

The authors would like to thank the computing center and information technology professors of VIT University for their sincere cooperation in providing facilities and knowledge to enhance the quality of the manuscript.

Appendix

During VM migration, host list sorting is achieved by considering low energy consumption, resource utilization through CRB ranking, the PEC function and the resource utilization square model. Transferring the memory resources of host does not have an impact on results. In this section, we denote the total iterations of the Algorithm 3 by i^k and the output by VM^{k+1}=VM. The optimized solution is denoted by VM_{opt}, corresponding resources by $h_{opt} = \cup_{VM_{opt}}^{r} h(vm)$ and minimum CPU utilization as $CPU_{\min} = \min(\delta_m^{mips} \cup \xi_m)$. The CPU utilization of all VMs are evaluated as $\sum_{j \in VM_j} CPU(VM) \leq \xi_i - \xi_u$ and $|VM| = [\frac{\xi_i - \xi_u}{CPU_{\min}}] + 1 = \phi$.

We indicate $\Omega^{i^k} = \delta_j^{mips}$ as the CPU utilization of the j^{th}VM selected in the i^{th} round of iteration. Hence

$\Omega^k = \xi_i - \xi_u - \sum_{k \neq 0}^{k-1} CPU(VM_j^m) = \alpha^k$

Lemma A.1. $\left|h^i(vm^i)\right| \leq \frac{\Omega^i}{\alpha^i} \cdot \phi \cdot h_{opt} \, \forall \, i \in (1, i^k)$

Proof. There are three cases □

Case 1: If $|h^i(vm^i)| \leq h_{opt}$ and sorted host list based on CRB ranking, then If $\forall \, i \in (1, i^k - 1)$ such that,

$\left|h^i(vm^i)\right| = CPU(VM_j^m) \cdot \frac{h^i(vm^i)}{\min(\alpha^k, CPU(VM_j^m))}$

Because algorithm 3 used to select the sorted host, it should be less energy consumed value $\frac{h^i(vm^i)}{\min(\alpha^k, CPU(VM_j^m))}$.

$CPU(VM_j^m) \cdot \frac{h^i(vm^i)}{\min(\alpha^k, CPU(VM_j^m))} \leq CPU(VM_j^m) \cdot \frac{h^i(\overline{vm^i})}{\min(\alpha^k, CPU(\overline{VM_j^m}))}$

Given $\overline{vm} \in vm_{opt}$ and $c(\overline{vm}) \geq \alpha^i$, we have $\frac{h^i(\overline{vm^i})}{\min(\alpha^k, CPU(\overline{VM_j^m}))}$ then

$$\begin{aligned}
&= \Omega^i \cdot \frac{h^i(\overline{vm^i})}{\alpha^i} \\
&\leq \frac{\Omega^i}{\alpha^i} \cdot |h_{opt}| \\
&\leq \frac{\Omega^i}{\alpha^i} \cdot \phi \cdot |h_{opt}|
\end{aligned}$$

Case 2: For i=i^k, give $\alpha^{i^k} = \Omega^{i^k} \leq CPU(vm^{i^k})$ finding and selecting VM from BRU-type list, we have

$\frac{h^{i^k}(\overline{vm^{i^k}})}{\alpha^{i^k}} = \frac{h^{i^k}(\overline{vm^{i^k}})}{\min(\alpha^{i^k}, CPU(\overline{VM_j^m})^{i^k})}$

$\leq \frac{h^{i^k}(\overline{vm})}{\min(\alpha^{i^k}, CPU(\overline{VM_j^m}))} = \frac{h^{i^k}(\overline{vm})}{\alpha^{i^k}}$

Hence,$h^{i^k}(\overline{vm^{i^k}}) \leq h^{i^k}(\overline{vm}) \leq |h_{opt}|$

Since, we have $\Omega^{i^k} = \alpha^{i^k}$

$\left|h^{i^k}(vm^{i^k})\right| = \frac{\Omega^{i^k}}{\Omega^{i^k}} \cdot \left|h^{i^k}(vm^{i^k})\right| = \frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot \left|h^{i^k}(vm^{i^k})\right|$

Based on $\left|h^{i^k}(vm^{i^k})\right| \leq |h_{opt}|$, we have

$$\begin{aligned}
&= \frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot \left|h^{i^k}(vm^{i^k})\right| \\
&\leq \frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot \left|h^{i^k}(\overline{vm})\right|
\end{aligned}$$

Since $\overline{vm} \in vm_{opt}$, we have

$$\frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot \left| h^{i^k}(\overline{vm}) \right| \\ \leq \frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot |vm_{opt}| \\ \leq \frac{\Omega^{i^k}}{\alpha^{i^k}} \cdot \phi \cdot |vm_{opt}|$$

Case 3: For selecting VM $\forall vm \in vm_{opt}$ from sorted host list based on PEC function, we obtain $CPU(vm_j^m) < \alpha^i$

$$\left.\begin{array}{l} \frac{h^i(vm^i)}{\alpha^i} = CPU(vm^i) \cdot \frac{h^i(vm^i)}{\min(\alpha^i, CPU(vm^i))} \\ \leq CPU(vm^i) \cdot \frac{h^i(vm^i)}{\min(\alpha^i, CPU(vm))} \end{array}\right\} or \begin{array}{l} CPU(vm) \cdot \left|h_i(vm^i)\right| \leq CPU(vm^i) \cdot \left|h_i(vm^i)\right| \\ = \Omega^i \cdot \left|h_i(vm^i)\right| \end{array} \tag{A.1}$$

Add sum on both sides of the equation apparently $vm \in vm_{opt} \backslash vm^i$

$\left|h^i(vm^i)\right| \cdot \sum_{vm \in vm_{opt}} CPU(vm) \leq CPU(vm^i) \cdot \sum_{vm \in vm_{opt} \backslash vm^i} \left|h^i(vm^i)\right|$

Resource consumption of all VMs of the respective host $\forall i \in [1, i^k]$, *let* $vm_j^m \cup vm_{opt}$, such that $\left|h^i(vm_j^i)\right| \leq |h_{opt}|$

$CPU(vm^i) \cdot \sum_{vm \in vm_{opt} \backslash vm^i} \left|h^i(vm^i)\right|$

$$\leq CPU(vm^i) \cdot \left|h_i(vm_j)\right| \cdot \left|vm_{opt} \backslash vm^i\right| \\ \leq CPU(vm^i) \cdot \left|h_i(vm_j)\right| \cdot \phi^i \\ \leq CPU(vm^i) \cdot \phi^i \cdot |vm_{opt}| \tag{A.2}$$

On the other side, we have

$$\sum_{vm} CPU(vm) \geq \xi_m - \xi_u - \sum_{vm_{opt}} CPU(vm) = \alpha^i \tag{A.3}$$

Based on Eq. (A.2) and (A.3), we obtain

$\alpha^i \cdot \left|h^i(vm^i)\right| \leq \left|h^i(vm^i)\right| \cdot \sum_{vm \in vm_{opt}} CPU(vm) \leq CPU(vm^i) \cdot \phi \cdot |vm_{opt}|$

That is, $\left|h^i(vm^i)\right| \leq \frac{\Omega^i}{\alpha^i} \cdot \phi \cdot h_{opt}$

Theorem A.1 (Algorithm 3)**.** *is an approximation polynomial time complexity and its ratio is*

$\Omega \cdot h(\xi_m - \xi_u) = \Omega \cdot (1 + \frac{1}{2} + \cdots + \frac{1}{\xi_m - \xi_u})$

Proof. Algorithm 3 time complexity is $O|n|^2$, we have

$$\sum_{i=1}^{i^k} \frac{\Omega^i}{\alpha^i} = \left(\frac{\Omega^1}{\sum_{j,m=1}^{vm_{j,m}} \xi_m - \xi_u} + \frac{\Omega^2}{\sum_{j,m=1}^{vm_{j,m}} (\xi_m - \xi_u) - \Omega^1} + \cdots + \frac{\Omega^i}{\Omega^{i-1}} \right) \\ \leq \left(\frac{1}{\xi_m - \xi_u} + \cdots + \frac{1}{\xi_m - \xi_u - \Omega^1 + 1} \right) + \cdots + \left(\frac{1}{\Omega^{10}} + \cdots + \frac{1}{1} \right) = 1 + \frac{1}{2} + \cdots + \frac{1}{\xi_m - \xi_u}$$

Therefore, we have
$$|vm| = \sum_{i=1}^{i^k} h^i(vm^i) \\ \leq \sum_{i=1}^{i^k} \frac{\Omega^i}{\alpha^i} \cdot \phi \cdot vm_{opt} \\ \leq \phi \cdot h(\xi_m - \xi_u) \cdot vm_{opt}$$ □

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.compeleceng.2018.11.021.

References

[1] Ray PP. Journal of king saud university-computer and information sciences. Surv Internet Thing Architect 2016:786–97 ISBN:9781509049264, ISSN:22131248. doi:10.1016/j.jksuci.2016.10.003.
[2] Bello O, Zeadally S. Intelligent device-to-device communications in the internet of things. IEEE Syst J 2016;10(3):40–50. doi:10.1109/JSYST.2014.2298837. ISSN:1932-8184 pp.1172–1182
[3] Mekala MS, Viswanathan P. CLAY - MIST:Iot - cloud enabled CMM index for smart agriculture monitoring system. Measur J 2018 ISSN:0263-2241. doi:10.1016/j.measurement.2018.10.072.
[4] Mekala MS, Viswanathan P. International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). A novel technology for smart agriculture based on IoT with cloud computing 2017:75–82. doi:10.1109/I-SMAC.2017.8058280.
[5] Varasteh A, Goudarzi M. Server consolidation techniques in virtualized data centers: a survey. IEEE Syst J 2015;99:1–12.
[6] Botta A. Integration of cloud computing and internet of things: a survey. Future Gener Comput Syst 2016;56:684–700.
[7] Xu M, Tian W, Buyya R. A survey on load balancing algorithms for virtual machines placement in cloud computing. Concurr Comput: Pract Exper 2017;29(12):1–16. doi:10.1002/cpe.4123.
[8] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener Comput Syst 2012;28(5):755–68.
[9] Wang W, Jiang Y, Member S, Wu W. Multiagent-based resource allocation for energy minimization in cloud computing systems. 2017. 47, 2, 205–220.
[10] Sun G, Liao D, Anand V, Zhao D, Yu H. A new technique for efficient live migration of multiple virtual machines. Future Gener Comput Syst 2016;55:74–86. doi:10.1016/j.future.2015.09.005.

[11] Gao S, Li L, Li W, Janowicz K, Zhang Y. Constructing gazetteers from volunteered big geo-data based on hadoop. computers. Environ Urban Syst 2017;61:172–86.
[12] Kim N, Cho J, Seo E. Energy-credit scheduler: an energy-aware virtual machine scheduler for cloud systems. Future Gener Comput Syst 2014;32:128–37.
[13] Li X, Qian Z, Lu S, Wu J. Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center. Math Comput Model 2013;58(5):1222–35.
[14] Huang CJ, Guan CT, Chen HM, Wang YW, Chang SC, Li C, Weng CH. An adaptive resource management scheme in cloud computing. Eng Appl Artif Intell 2013;26(1):382–9.
[15] Rao KS, Thilagam PS. Heuristics based server consolidation with residual resource defragmentation in cloud data centers. Future Gener Comput Syst 2015;50:87–98 ISSN:0167-739X. doi:10.1016/j.future.2014.09.009.
[16] Murtazaev A, Oh S. Sercon: server consolidation algorithm using live migration of virtual machines for green computing. IETE Tech Rev 2011;28(3):212–31.
[17] Jeyarani R, Nagaveni N, Ram RV. Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence. Future Gener Comput Syst 2012;28(5):811–21.
[18] Zhang Z, Zhang N, Feng Z. Multi-satellite control resource scheduling based on ant colony optimization expert systems with applications. 2014. ISSN:0957-4174. 41, 6, 2816–2823, doi:10.1016/j.eswa.2013.10.014.
[19] Kusic D, Kephart JO, Hanson JE, Kandasamy N, Jiang G. Power and performance management of virtualized computing environments via lookahead control. Clust Comput 2009;12(1):1–15.
[20] Hieu N.T, Francesco M.D. 2014 IEEE international conference on cloud computing a virtual machine placement algorithm for balanced resource utilization in cloud data centers. 2014. ISSN:2159-6190 doi:10.1109/CLOUD.2014.70, 474–481.
[21] Tarighi M, Motamedi SA, Sharifian S. A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making. J Telecommun 2010;1(1):40–51. https://arxiv.org/abs/1002.3329.
[22] Yufan H, Liu P, Wu J-J. Server consolidation algorithms with bounded migration cost and performance guarantees in cloud computing, utility and cloud computing (UCC). In: Proceedings of the 2011 Fourth IEEE International Conference on; 2011. p. 154–61.
[23] Zhang J, Huang H, Wang X. Resource provision algorithms in cloud computing: a survey. J Netw Comput Appl 2016;64:23–42.
[24] Mekala MS, Viswanathan P. A survey: smart agriculture iot with cloud computing. 2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS) 2017:1–7 ISBN:9781538617168. doi:10.1109/ICMDCS.2017.8211551.
[25] Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software 2011;41(1):23–50.

Mahammad Shareef Mekala is a research associate at VIT University, Vellore, India. He received his B. Tech. (CSE) degree in 2011 and his M. Tech. (CSE) degree in 2013 from JNTU University, Anantapur. He has also received a VIT research award in 2017. His research interests include IoT, cloud computing, wireless networks, digital image processing and, watermarking.

P. Viswanathan is an associate professor at VIT University, Vellore, India. He received his B.E. (CSE) degree from Madurai Kamaraj University in 2002, his M.E. (CSE) degree from Annamalai University in 2006 and his Ph.D. from VIT University in 2014. He received the best poster award from ISCA in 2007 and VIT researcher award from 2011 to 2017. His research interests include IoT, cloud computing and, digital image processing.