symmetry

MDPI

Article

EAMA: Efficient Adaptive Migration Algorithm for Cloud Data Centers (CDCs)

Muhammad Ibrahim [1], Muhammad Imran [2], Faisal Jamil [1], Yun-Jung Lee [3] and Do-Hyeun Kim [1,*]

[1] Department of Computer Engineering, Jeju National University, Jeju 63243, Korea; ibrahimmayar@jejunu.ac.kr (M.I.); faisal@jejunu.ac.kr (F.J.)
[2] Department of Electronics and Computer Engineering, Hongik University Sejong Campus, Sejong 2639, Korea; royimranpk@gmail.com
[3] Department of Computer Science and Statistics, Jeju National University, Jeju 63243, Korea; rheeyj@jejunu.ac.kr
* Correspondence: kimdh@jejunu.ac.kr

Abstract: The rapid demand for Cloud services resulted in the establishment of large-scale Cloud Data Centers (CDCs), which ultimately consume a large amount of energy. An enormous amount of energy consumption eventually leads to high operating costs and carbon emissions. To reduce energy consumption with efficient resource utilization, various dynamic Virtual Machine (VM) consolidation approaches (i.e., Predictive Anti-Correlated Placement Algorithm (PACPA), Resource-Utilization-Aware Energy Efficient (RUAEE), Memory-bound Pre-copy Live Migration (MPLM), m Mixed migration strategy, Memory/disk operation aware Live VM Migration (MLLM), etc.) have been considered. Most of these techniques do aggressive VM consolidation that eventually results in performance degradation of CDCs in terms of resource utilization and energy consumption. In this paper, an Efficient Adaptive Migration Algorithm (EAMA) is proposed for effective migration and placement of VMs on the Physical Machines (PMs) dynamically. The proposed approach has two distinct features: first, selection of PM locations with optimum access delay where the VMs are required to be migrated, and second, reduces the number of VM migrations. Extensive simulation experiments have been conducted using the CloudSim toolkit. The results of the proposed approach are compared with the PACPA and RUAEE algorithms in terms of Service-Level Agreement (SLA) violation, resource utilization, number of hosts shut down, and energy consumption. Results show that proposed EAMA approach significantly reduces the number of migrations by 16% and 24%, SLA violation by 20% and 34%, and increases the resource utilization by 8% to 17% with increased number of hosts shut down from 10% to 13% as compared to the PACPA and RUAEE, respectively. Moreover, a 13% improvement in energy consumption has also been observed.

Keywords: VM migration; VM placement; CloudSim; cloud computing; server consolidation

check for updates

Citation: Ibrahim, M.; Imran, M.; Jamil, F.; Lee, Y.; Kim, D.-H. EAMA: Efficient Adaptive Migration Algorithm for Cloud Data Centers (CDCs). *Symmetry* **2021**, *13*, 690. https://doi.org/10.3390/sym13040690

Academic Editor: Raúl Baños Navarro

Received: 15 March 2021
Accepted: 11 April 2021
Published: 15 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Cloud computing has revolutionized industry and academia by the provisioning of on-demand computing resources. These on-demand resources are based on pay-as-you-use. Organizations and individuals outsource Cloud services for the diverse nature of tasks that demand high-performance computing, large memory instances, licensed software applications, large-scale simulations [1] and development platforms, etc. Specifically, organizations prefer to outsource Cloud services rather than designing their own private data centers for their business needs due to the huge investment, maintenance, and management cost. The high demands of organizations for Cloud services resulted in the establishment of large-scale Cloud Data Centers (CDCs). Enormous energy consumption is a big problem in CDCs. Energy consumption in CDCs has increased by 56% from 2005 to 2010, representing 1.1 to 1.5% of the global electricity use in 2010 [2,3]. It is reported that the high energy consumption is due to the inefficient use of computing resources at the data center,

i.e., approximately 93% of the energy consumed is by the Cloud computing resources [4,5]. The authors of [6] considered the grid computing environment (having 5000 servers) to monitor energy consumption behavior. Most of the servers in the data center remain running, and the utilization of these servers is approximately 10–50% recorded for 5000 servers over a six-month period. The study shows that there are significant opportunities to reduce energy consumption in the CDCs via switching servers to low power or powering off. This energy consumption results in the wastage of resources as well as the extra provisioning of resources that ultimately increases the total cost of ownership. These idle servers in the CDC consume 70% of its total power, and this extra energy consumption also contributes to higher carbon emissions. It is estimated that emission of CO_2 across CDCs is 2% of the overall global emission of CO_2 and this affects sustainable IT solutions [7]. The two of the main reasons for excessive energy consumption in CDCs are inefficient use of resources and imbalanced Virtual Machine (VM) placement on hosts (overloading and under-loading), etc. When a host utilizes all available resources and demands more resources to complete the user jobs, this state is called host overloaded. The host is said to be in an under-loaded state if the resources of the hosts are below or equal to a certain utilization threshold. The host overload problem further leads to the Service-Level Agreement (SLA) violation [8]. In general, SLA violation is considered a bad experience for the provisioning of Cloud services to the Cloud customers, which means that Cloud customers would not be satisfied with the Cloud services. Therefore, Cloud providers have to pay a penalty to the customers based on the SLA agreements.

If a host experiences an under-loaded problem in the CDC, all Virtual Machines (VMs) of that host are migrated to other hosts using live migration [9] and concerned under-loaded hosts are then powered off accordingly. In this way, the number of active hosts is reduced across the CDCs which ultimately leads to reduced energy consumption. However, aggressive consolidation of VMs can cause performance degradation due to the high demands of resources. Moreover, if the required resources are not available to the application for its processing, it will ultimately lead to an increase response time, which may result in application failure or late response. Numerous approaches have been proposed by researchers to improve the resource utilization with reduced energy consumption in CDCs. Dynamic VM consolidation and management approaches proposed in [10–16] have shown improvements in the energy consumption. These VM consolidation approaches provide benefits of dynamic workload adjustment, where the VMs are periodically re-allocated according to their demands of resources. Although these approaches minimize the number of active hosts in the CDCs, most of these approaches ignore the SLA requirements and efficient use of resources. Few dynamic approaches conduct aggressive consolidation of VMs, which leads to SLA violations and inefficiencies in the utilization of resources. Moreover, these approaches also have not considered the user's location during the VM migration and placement. To address these issues, we propose an efficient VM migration approach, which performs VM consolidation across CDCs to maximize the resource utilization and reduce energy consumption, while keeping SLA violation to a minimum.

The proposed approach is comprised of four phases: The first two phases are related to the detection of the overloaded and under-loaded hosts. The third phase deals with the VM migration from overloaded hosts to under-loaded hosts according to nearby locations. The last phase detects the remaining overloaded/under-loaded hosts and initiates the final migration and placement of VMs. The proposed approach has two main features: first, selection of the location of the PM which has less delay as compared to the other PMs where the VMs are required to be migrated. Second, the number of migrations is reduced (i.e., first the VMs from the overloaded PMs are migrated to the under-loaded PMs and then the remaining VMs from the lightly under-loaded PMs are migrated to the remaining under-loaded PMs. At the end, the final migrations of the VMs from the under-loaded/overloaded to the new PMs are performed (where required).). As a result, the number of migrations is reduced and a higher number of under-loaded PMs are shut down, leading to efficient resource utilization. Moreover, the reduction in energy consumption is also achieved.

The rest of the paper is organized as follows. The related work is presented in Section 2. In Section 3, the proposed approach is delineated. The simulation setup is presented in Section 4. Section 5 illustrates the results and discussion. In Section 6, the comparative analysis and recommendations are discussed. Finally, Section 7 presents the conclusions and future work.

2. Related Work

In the last few years, a number of approaches have been proposed to resolve the VM migration and placement problems across the CDCs in the context of Cloud computing. Nathuji and Schwan [17] presented an architecture of CDC resource management that partitions the resources into local and global components. The operating systems are leveraged at the local level for power management strategies. A global manager is responsible for obtaining the resource allocation information for the VMs placement. Arroba et al. [18] proposed a Dynamic Voltage Frequency Scaling (DVFS) policy to optimize the energy consumption in CDCs while considering the QoS and mapping of VMs to the resources. The authors performed simulation experiments using the CloudSim toolkit and proved that the proposed DVFS manages workload across the CDC in an appropriate manner and saves 41.62% energy under dynamic workload conditions. Reddy et al. extended the Particle Swarm Optimization (PSO) in [19] by contributing a Modified Discrete Particle Swarm Optimization (MDPSO) approach. For VM selection, the authors presented the Memory Bandwidth and Size (MBS-VM) algorithm. The MBS-VM selection algorithm selects the VMs from the under-loaded and overloaded PMs for migration. The authors obtained the simulation results by using the CloudSim toolkit and compared the results with the Modified Best Fit Decreasing (MBFD) algorithm and PSO-based VM placement approaches. The MDPSO approach reduces the energy consumption by 32%, VM migrations up to 30%, and idle PMs up to 40% as compared to MBFD and PSO-based VM placement approaches. A mathematical model-based approach is proposed by Mazumdar and Pranzo in [20] that reduces the energy consumption in data centers. The proposed approach is a snapshot-based server consolidation technique for the Cloud Infrastructure Provider (CIP). The authors proposed a solution that considers multiple objectives including (i) failing and mapping of incoming VMs (ii) consolidation of server workload (iii) and reducing the number of VM migrations. The proposed mathematical formulation improves the results compared to Best-Fit heuristics techniques by 6–15%. Tarahomi and Izadi, in [8], proposed an online resource management technique with live VM migration in CDCs. The authors also designed a prediction-based and power-aware VM allocation algorithm. The proposed algorithms were compared with the well-known Power-Aware Best Fit Decreasing (PABFD) algorithm by using the CloudSim toolkit. Results show that the proposed algorithms outperformed the PABFD algorithm by reducing energy consumption with decreased number of active hosts, VM migrations, and SLA violation.

To reduce energy consumption in CDCs, the authors of [21] consider CPU temperature/cooling infrastructure, server workload, and dynamic workload using a novel scheduling algorithm called GRANITE. The proposed technique reduces the energy consumption as well as the probability of critical temperature violations by 99.2%, and keeps a low SLA violation rate of 0.17%.

An improved genetic algorithm for efficient VM scheduling is proposed in [22]. The proposed algorithm keeps CPU utilization up to the specified threshold and below the host's capacity. In this approach, chromosomes denote each physical machine and the virtual machines are considered genes of the chromosomes. This approach allocates the tasks and VMs in such a way that reduces the overall energy consumption of the CDC.

The authors of [23] present a unique scheduling algorithm called Static Scheduling using Chemical Reaction Optimizer (SS-CRO). The SS-CRO approach test schedules with different instructions dependencies before making the selection. It provides different schedule orders and considers the most optimum scheduling solution. The proposed approach was evaluated experimentally as well as analytically and compared with Static Scheduling using Duelist Algorithm (SS-DA) and Static Scheduling using Genetic Algorithm (SS-GA) approaches. The obtained results show that the proposed approach achieves higher accepted solutions and better execution time.

Karda and Kalra designed a novel algorithm entitled Ant Colony Optimization (ACO) in [24], which places VMs on the physical hosts while considering the CPU utilization. ACO maps VMs on those hosts that have no probability to be overloaded in the future. Host selection for VM placements is initiated in such a way that it provides the benefits of minimum migrations across CDC. The ACO approach outperformed the existing Threshold Energy Saving Algorithm (TESA) approach considering energy consumption, number of migrations, and SLA violations.

A metaheuristic algorithm for task scheduling called the Humpback Whale Optimization Algorithm (HWOA) is proposed in [25]. The humpback whale vocalization behavior is mimicked to provide an optimum task scheduling in Cloud computing. The proposed HWOA scheduler outperformed the traditional Whale Optimization Algorithm (WOA) and Round Robin (RR) concerning the load imbalance, makespan, resource utilization, cost, and energy consumption.

Fatima et al. proposed a PSO-based novel algorithm that solves the problem of inefficient utilization of resources in [26]. The results are compared with existing PSO, Levy Flight, and Particle Swarm Optimization (LEPSO) algorithms. The results reveal that the proposed algorithm increased resource utilization and decreased the number of active physical machines across the CDCs. An algorithm proposed by Li et al. [27] minimizes the number of migrations and reduces the active number of PMs in CDCs. The algorithm packs the VMs using the page-sharing technique that has two steps: (1) it packs the maximum number VMs on the host and (2) it uses the approximation algorithm to find the approximation ratio. The algorithm approximation ratio is better than the other approaches in terms of memory and server utilization. The experimental results show that the proposed algorithm outperforms as compared to Greedy-Flow and First Fit VM placement approaches and it requires 40% fewer memory pages as well as 25% fewer PMs.

A novel energy-aware cost prediction framework for CDCs is proposed in [28], which predicts the workload, power consumption, and the total cost of VMs. This framework assesses the cost and workload predictions from the workload patterns. The proposed approach is also used to analyze energy pricing for resource utilization and can place the VMs in a load-balanced way.

A load-balancing approach called Cluster Dimension Exchange Method (CDEM) is proposed in [29], which focuses on efficient load-balancing on the promising Optical Transpose Interconnection System (OTIS)-Hypercube interconnection networks. The experimental evaluation and analytical model proved the effectiveness of OTIS-Hypercube over Hypercube in terms of load-balancing accuracy, execution time, speed, and number of communication steps. A load-balancing approach based on Chained-Cubic Tree (CCT) interconnection networks was proposed in [30]. The proposed algorithm is called Hybrid Dynamic Parallel Scheduling Algorithm (HD-PSA). This algorithm is a combination of two load balancing strategies: parallel scheduling and dynamic load balancing. The proposed algorithm performance is evaluated both experimentally and analytically in terms of execution time, communication cost, load balancing accuracy, task locality, and number of tasks hopes.

A Dynamic Resource Allocation (DRA) approach is proposed in [31] by Yang et al. that saves energy utilization in CDCs. This approach consists of various steps, i.e., deployment of OpenStack-based infrastructure platform for VMs live migration, allocation of dynamic resources and energy-saving algorithms, and monitoring of power distribution to record energy consumption. The experimental results show that the proposed approach reduces energy consumption by 39.89%. Nashaat et al., in [32], proposed an approach that solves the problem of performance degradation, energy consumption, and number of migrations. The authors proposed two algorithms: Smart Elastic Scheduling Algorithm (SESA) and Worst Fit Decreasing Virtual Machine Placement (AWFDVP). The SESA algorithm arranges the virtual machines in the cluster according to their CPU and memory utilization. After cluster formation, these algorithms analyze the co-located VMs that share memory pages on the same physical node in a group. Next, the VM placement algorithm, i.e., AWFDVP performs the migration of VMs. The obtained results demonstrate that the system reduces the performance degradation by 57%, the number of VM migrations by 57.7%, and energy consumption by 28%.

Table 1 delineates the comparison of different state-of-the-art VM migration approaches in terms of their contributions (i.e., VM Placement, VM migration, etc.) and metrics (i.e., Energy Consumption, SLA Violation, Resource Utilization, Host Shutdown, and Number of Migrations) utilized for the performance evaluation.

Table 1. Summary of state-of-the-art Virtual Machine (VM) migration approaches.

Approaches	Contributions				Performance Metrics			
	VM Selection	**VM Migration**	**Location Aware**	**Energy Consum.**	**SLA Violation**	**Resource Util.**	**Host Shutdown**	**No of Migration**
Freq-Aware Optimization approach [18]	✓	✓	-	✓	✓	✓	✓	✓
MDPSO [19]	✓	✓	-	✓	✓	-	-	✓
Snapshot based approach [20]	✓	✓	-	✓	✓	-	✓	✓
Prediction-based VM placement [8]		✓	-	✓	✓	✓	✓	✓
GRANITE [21]	-	✓	-	✓	-	-	✓	-
GA-LS [22]	-	✓	-	✓	-	✓	-	-
ACO [24]	✓	✓	-	✓	✓	-	-	✓
CBPSM [27]	✓	✓	-	-	-	✓	✓	-
DRA method [31]	-	✓	-	✓	-	✓	-	-
SESA-AWFDVP [32] algorithms	-	✓	-	✓	✓	✓	-	✓
Proposed VM Migration Approach	✓	✓	✓	✓	✓	✓	✓	✓

The major issues with most of the VM migration for power efficiency include aggressive consolidation decisions, uncontrolled migrations, and non-optimal VM placement [33]. To address these issues, we have proposed an efficient VM migration approach, which performs VM consolidation across CDCs to maximize the resource utilization and reduce energy consumption, while keeping SLA violation to a minimum.

3. Efficient Adaptive Migration Algorithm (EAMA)

In this work, we have addressed the problem of VM migration by proposing an Efficient Adaptive Migration Algorithm (EAMA).

The proposed approach consists of four main phases: The first two phases are related to the detection of the overloaded and under-loaded hosts. The third phase deals with the VM migration from overloaded hosts to under-loaded hosts according to nearby locations. The last phase detects the remaining overloaded/under-loaded hosts and initiates the final migration and placement of VMs. For VM migration, we extended the SIM-Cumulus framework [34] by adding a Migration Management module (as shown in Figure 1). The Modified SIM-Cumulus comprises four layers: System Accessibility Layer (SAL), Cloud Instance Management Layer (CIML), Modified Virtual Platform Layer (MVPL), and Physical Infrastructure Layer (PIL). The SAL layer provides a GUI to users to interact with the Cloud services. The CIML layer performs the configuration and management of virtual machine instances. The MVPL has been modified to support VM migrations. At PIL layer of the architecture, the physical computing resources like machines, clusters, data center, and network storage are managed. The details pertaining to each layer are discussed in [34]. At the MVPL layer, EAMA is implemented at the *Migration Management Module*, which is responsible for the migration of VMs across the hosts in the CDCs. For the VM migrations, EAMA works in four phases shown in Figure 2. This is the flowchart of how different stages are employed for the VM migration. Initially, the under-loaded and overloaded hosts are detected across the Cloud data centers and are provided as an input to the VM migration and placement step. In this step, the VMs from overloaded PMs are migrated to the under-loaded PMs according to a nearby location. The nearby location gives the minimum delay in services provisioning to the Cloud user that is possible if the VMs are migrated to nearby located DCs. This process continues until all overloaded PMs go to normal states. At the end of this step, a list of remaining overloaded or under-loaded PMs is returned to the final migration algorithm. In last step, if there are some remaining overloaded/under-loaded hosts, the VMs are migrated to normal or new hosts. Moreover, the under-loaded PMList is divided into two halves called the First Half List (FHL) and Second Half List (SHL). The VMs in the FHL PMs are migrated to the PMs of the SHL. Once all the PMs in the SHL becomes normal, the VMs from the remaining PMs are migrated to new or other normal PMs. The detailed discussion about each phase is discussed in next subsections.

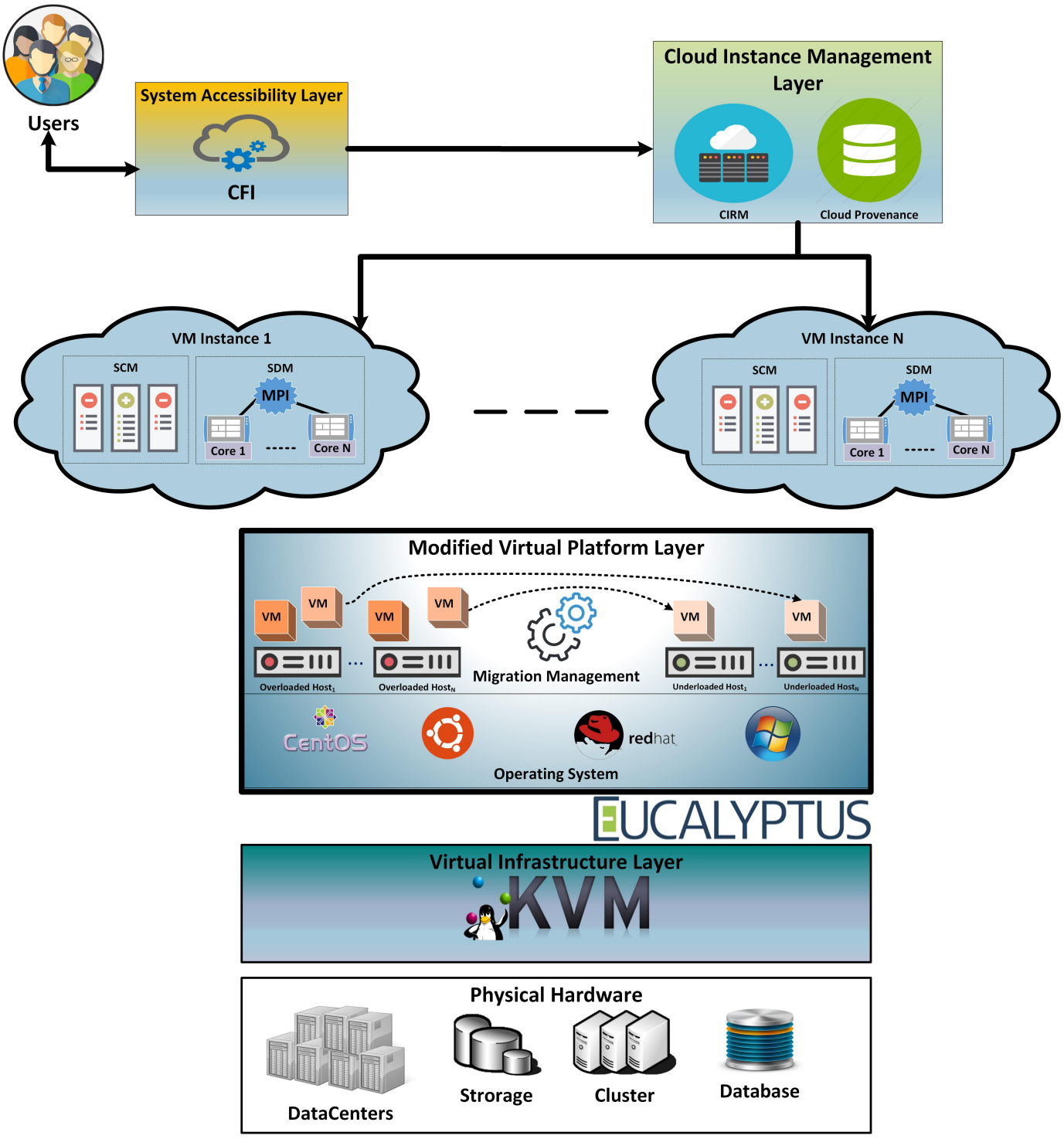


Figure 1. Migration-enabled Sim-Cumulus architecture.

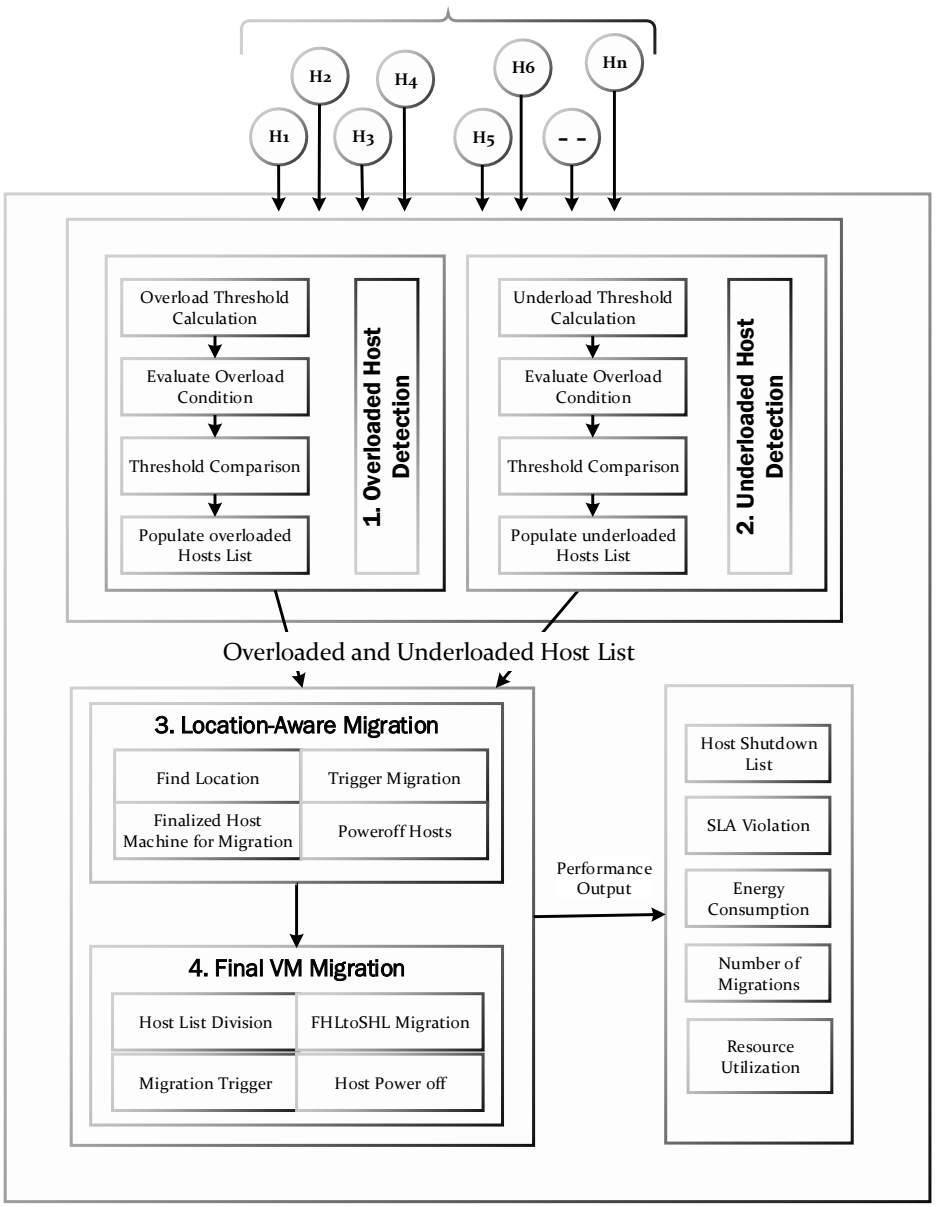


Figure 2. VM migration flowchart.

3.1. Overloaded Host Detection

After the initial allocation of VMs, the algorithm periodically analyzes the overloaded PMs across the data center. The overloaded host detection algorithm detects the overloaded hosts running in DC and places overloaded hosts into the overloaded host list, and this list is then passed to the migration and placement algorithm. Algorithm 1 performs the detection of overloaded hosts in CDCs. The input parameters pmUtilization, pmTotalCapacity, and overUtilizedPmList are declared in lines 1 to 3. In Algorithm 1, line 4 evaluates all the PMs in the Cloud data center one by one for overloaded host detection. Line 5 of the Algorithm 1 represents the PM's total capacity in MIPS. The total MIPS capacity corresponds to all the MIPS which are allocated to PM in the data center. At line 6, the Algorithm 1 gets the current utilization of MIPS of the PM. The Algorithm checks the current utilization of PM in line 7; if it is equal to or greater than the total MIPS of the PM, then the PM is placed to the overloadedPMsList. This list is then passed to the migration and placement algorithm.

Algorithm 1 Overloaded Host Detection

Input: PM_{List} = List of PMs $(i.e., PM_1, PM_2, PM_3, \ldots, PM_n)$
Output: *overloadedPMsList*

1: $pmUtilization \leftarrow 0$
2: $pmTotalCapacity \leftarrow 0$
3: $overUtilizedPmList \leftarrow 0$
4: **for all** PM *in* pm_{List} **do**
5: $pmTotalCapacity.get.TotalMips[PM]$
6: $pmUtilization.get.Utilization[PM]$
7: **if** $(pmUtilization \geq pmTotalCapacity)$ **then**
8: **Add** PM *to* $overloadedPMsList$
9: **end if**
10: **end for**
11: **Return** $overloadedPMsList$

3.2. Under-Loaded Host Detection

In this phase, all PMs that are running below a lower threshold is detected. The lower threshold in our approach is 30% of a host total CPU (MIPS). A list of under-loaded hosts is populated at the end of this phase and is passed to the migration and placement algorithm for VMs migration.

Once all the overloaded hosts are detected, the next step is to find out the list of hosts that are in the under-loaded state.

Algorithm 2 detects the under-loaded hosts in CDC. The hosts' utilization is checked according to a lower threshold. Parameter pmTotalCapicity holds the total MIPS capacity of a PM. Parameter *pmUtalization* holds current utilized MIPS of a PM. *maximumUtalization* contains the lower threshold value. At line 4 (of Algorithm 2), *foreach* analyzes all the PMs in the data center one by one. The lower threshold is defined on line 7. At line 8 (of Algorithm 2), if the current utilization of PM is less than or equal to the lower threshold value, then this PM is considered as under-loaded. The under-loaded PM is placed in *under-loadedPMsList*. If the current utilization of PM is greater than the lower threshold, then the next PM will be analyzed. After evaluating all PMs in the DC, a list of under-loaded hosts *under-loadedPMsList* is prepared. This list is returned to the migration and placement algorithm for the final migration and placement of VMs.

Algorithm 2 under-loaded Host Detection

Input: PM_{List} = List of PMs $(i.e., PM_1, PM_2, PM_3, \ldots, PM_n)$
Output: *underloadedPMsList*

1: $pmUtilization \leftarrow 0$
2: $pmTotalCapacity \leftarrow 0$
3: $lowerThreshold \leftarrow 0$
4: **for all** PM *in* pm_{List} **do**
5: $pmTotalCapacity = PM.get.TotalMips[PM]$
6: $pmUtilization = PM.get.Utilization[PM]$
7: $lowerThreshold = pmTotalCapicity \times 30/100$
8: **if** $(pmUtilization \leq lowerThreshold)$ **then**
9: **Add** PM *to* $underloadedPMsList$
10: **end if**
11: **end for**
12: **Return** $underloadedPMsList$

3.3. VM Migration and Placement

In the third phase, the VMs of the overloaded hosts are migrated to under-loaded hosts according to the users' nearby location (the data center that is closer to VM or the data

center that has a minimum delay). After overload and under-load host detection, the next step is to migrate the VMs from the overloaded hosts to the under-loaded hosts as given in Algorithm 3. The VM migration and placement algorithm is used for VM migration and placement from overloaded PMs to the under-loaded PMs according to a nearby location. The nearby location gives the minimum delay in services provisioning to the Cloud user that is possible if the VMs are migrated to nearby located DCs. The algorithm obtains the list of overload/under-loaded PMs from the overloaded host detection algorithm and under-loaded host/detection algorithm. The under-loaded PMs are then sorted in descending order for VMs migration. At first, the algorithm places VMs from overloaded PMs to the under-loaded PMs. The algorithm analyzes the overloaded PM from the overloaded PM list (line 7, Algorithm 3). VMs of the selected overloaded host are sorted in ascending order by their MIPS utilization. At line 11, the algorithm compares the size of VM in MIPS with the available capacity of the under-loaded PM. If the capacity on the under-loaded PM is enough to service the VMs, then the VM is placed to the under-loaded PM; otherwise, the next PM is selected for the VM. After migration of VM from overloaded PM to the under-loaded PM, according to a nearby location, the selected under-loaded PM goes to the normal utilization list (line 14, Algorithm 3). This process continues until all overloaded PMs go to normal utilization. At the end of execution, a list of remaining overloaded or under-loaded PMs is returned to the final migration algorithm.

Algorithm 3 VM Migration and Placement

Input: *underloadedPMsList and overoadedPMsList*
Output: *underloadedPMsList*

1: $overloadedPMsList$
2: $underloadedPMsList$
3: $undrloadedPMs.sortDec()$
4: **for all** OPM *in* $overloadedPMsList$ **do**
5: $RemOverLoadedHosts = overloadedPMsList - 1$
6: $OPM.VM.sortAsc()$
7: **for all** VM *in* $overloadedPMsList$ **do**
8: **for all** UPM *in* $underloadedPMsList$ **do**
9: $RemUndLoadedHosts = underloadedPMsList - OPM$
10: **if** $(uderloadedPM.location == VM.location)$ **then**
11: **if** $(VM.size \leq underloadedPM.size)$ **then**
12: **Add** PM *to* $underloadedPMsList$
13: $PlaceVM(undrloadedPM, VM)$
14: $addNormalList1(undrloadedPM, PM.ID)$
15: **end if**
16: **else**
17: **if** $((VM.size \leq underloadedPM.size) \quad \&\& \quad (underloadedPMs.DC.delay.smallest))$ **then**
18: $PlaceVM(underloadedPM, VM)$
19: $addNormalList1(underloadedPM, PM.ID)$
20: **end if**
21: **end if**
22: **if** $((underloadedPMlist == 0) \quad \&\& \quad (RemOverLoadedHosts == 0))$ **then**
23: $break$;
24: **end if**
25: **end for**
26: **end for**
27: **end for**
28: **Return** $underloadedPMsList$

3.4. Final Migration

This is the final phase of our proposed algorithm. In this phase, if there are some remaining overloaded/under-loaded hosts, the VMs are migrated to normal or new hosts. The final migration is executed once the under-load and overload hosts detection being finalized and the VMs from the overloaded PMs are migrated to some of the under-loaded PMs. The final migration algorithm gets the remaining overloaded and under-loaded PM lists from VM migration and placement algorithm. The algorithm divides the under-loaded PMList into two halves called the First Half List (FHL) and Second Half List (SHL). The VMs in the FHL PMs are migrated to the PMs of the SHL. Once all the PMs in the SHL becomes normal, the VMs from the remaining PMs are migrated to new or other normal PMs. Migrating in this way has two benefits: the number of migrations is reduced and the number of host shutdown is increased. Algorithm 4 moves the VMs from the FHL PMs one by one by comparing the size of the destination PMs in the SHL as given in Lines 7 and 8. The utilization of the PMs in checked in Line 10 (of Algorithm 4), and if no load is remaining on the PM, then that PM is removed from the list and powered off. In this way, all the VMs from the PMs of FHL are moved to PMs of SHL.

Algorithm 4 Final Migration

Input: $underloadedPMsList$
Output: $FinalizedNormalList$

1: $FirsthalfUnderPM$
2: $SecondhalfUnderPM$
3: $undrloadedPMs.sortDec()$
4: **for all** $FHPMs$ *in* $FirsthalfUnderPM$ **do**
5: **for all** $SHPMs$ *in* $SecondhalfUnderPM$ **do**
6: **for all** VM *in* $SecondhalfUnderPM$ **do**
7: **if** $(VM.size < FHPMs)$ **then**
8: $PlaceVM(FHPMs, VM)$
9: *Break*
10: **if** $(FHPMs.utilization == 0)$ **then**
11: *Break*
12: **end if**
13: **else**
14: *Break*
15: **end if**
16: **end for**
17: **end for**
18: **end for**

3.5. RAUEE and PACPA Approaches

In this subsection, a brief overview of two contemporary migration approaches (compared in this work) is presented. The work in [35] proposed an energy-efficient server consolidation approach that has increased the resource utilization and reduced the number of live migrations across the CDC. The RUAEE model works in four sections: finding the overloaded hosts, adjusting the unbalanced machines, under-loaded host selection, and finally placement of VMs. At the start, the overloaded module classifies the hosts into either normal hosts or overloaded hosts. In the next module, it then finds out the unbalanced hosts to be used for workload adjustment. In the third step, the low-utilized machines are detected using the SF value and then migrating all running VMs to power off to save energy. In the last step, the hosts are selected for final VM migration using the VM placement module. The PACPA algorithm predicts the short-term CPU requirements of the VMs that need to be reallocated due to the high fluctuating application workloads demands [11]. This approach detects the overloaded hosts and the selection of VMs considering two key features of dynamic VM resource optimization. The chosen VMs are then

provided as input to the PACPA approach. The PACPA uses the selected forecasting model to estimate the costs of migration Ctij keeping in view the RAM specifications for the VM used. The algorithm uses penalty value M to avoid an increased number of migrations. The proposed approach assumes that any VM migration that will execute after more than five time steps will be penalized based on the anticipated delay at time t.

4. Simulation Setup

To investigate the performance of any protocol, experimental, analytical, or simulation approaches are generally used. The experimental approaches are the most powerful type; however, these approaches are costly and are difficult to configure and thus sometimes need an expert to set up the testbeds for experiments. Moreover, executing the simulation multiple times on the real Cloud platform may result in high monitory cost, whereas the analytical approaches are often limited in evaluating the migration approaches. Therefore, the simulation approaches are widely employed for performance evaluation of the underlying migration techniques using a variety of configurations. To evaluate the performance of the migration approaches including the proposed EAMA approach, Cloudsim [36,37] simulation platform is employed. The simulation experiments were performed on a workstation equipped with Intel Core i5-8500 Quad-core processor (3.0 GHz clock speed) and 8 GBs of main memory. Table 2 shows the details regarding the parameters (i.e., total data centers, number of PMs, and number of VMs) used for the simulation.

Table 2. Simulation parameters.

Parameter	Value
Number of Data centers	5
Total Number of PMs	50
Host MIPS	15,000
Total Number of VMs	150
Number of Runs	10

5. Result and Discussion

Obtained results of the proposed EAMA approach are compared with PACPA [11], RUAEE [35] in terms of SLA violation, energy consumption, number of hosts shut down, resources utilization, and number of migrations. The proposed EAMA approach migrates and places VMs on the PMs in a way that results in a reduced number of migrations with minimum SLA violation and better resource utilization. The proposed approach migrates VMs from overloaded PMs to under-loaded PMs according to the location of VMs/users, thus making the proposed approach superior over the compared approaches in terms of metrics (i.e., energy consumption, number of hosts shut down, resources utilization, SLA violation, and number of migrations) compared. The experiments are carried out by utilizing a well-known CloudSim simulator [38]. The simulation experiments are executed ten times with different seed values and results are obtained in terms of (i) energy consumption, (ii) SLA violation, (iii) the number of migrations, and (iv) the number of hosts shutdown. The obtained results and relevant discussions are presented as follows.

The CPU utilization model shows the utilization of all PMs in the data center. The model sums the utilized MIPS of all the PMs in DC. To evaluate the performance in terms of resource utilization (CPU), Equation (1) has been utilized [33].

$$Ut = \sum_{i=1}^{n} PMUtil_i \quad (1)$$

The simulations were executed multiple times and the results concerning the resource utilization for the proposed approach and two state-of-the-art VM migration approaches (i.e., PACPA and RUAEE) are obtained and illustrated in Figure 3. In Figure 3, the X-axis

represents the migration approach, and resource utilization (i.e., CPU) is plotted on the Y-axis. The EAMA approach leads to 88% resource utilization showing 8% and 17% improvement as compared to the PACPA and RUAEE approaches, respectively. The obtained results assert the effectiveness of the proposed approach against the existing compared approaches.

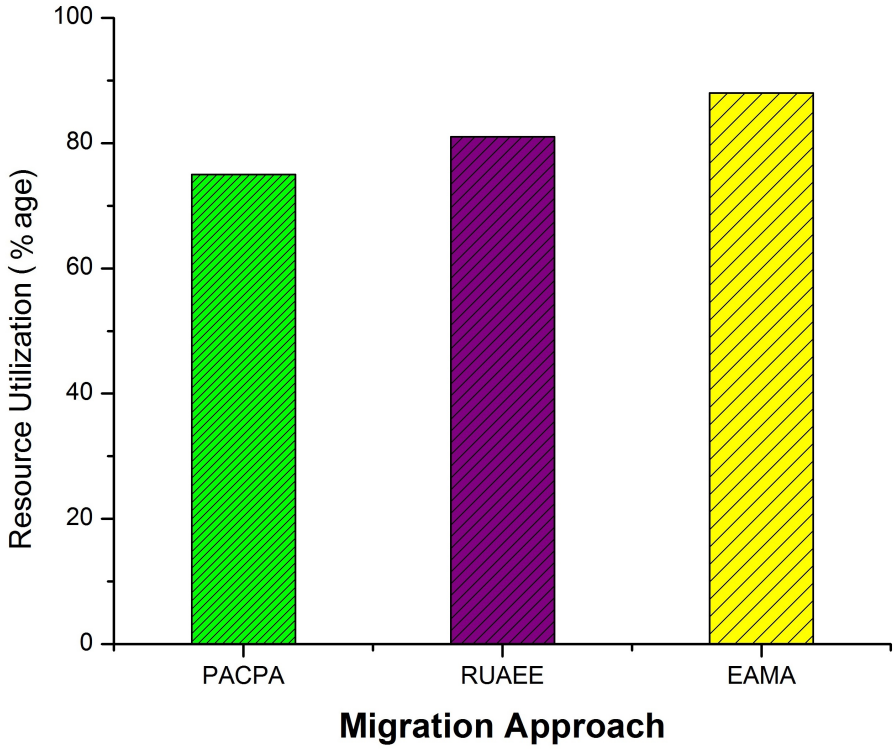


Figure 3. DC resources utilization.

The energy consumed metric is another important parameter used to evaluate the performance of any given migration algorithm across the Cloud data centers. To evaluate the energy consumption across the Cloud data centers, the power model used in [33] is utilized for server components that are considered for energy calculation. The main components of the server are CPU, memory, cooling, network, and storage. For this work, we have only considered the CPU as the main component for energy consumption calculation. Equations (2) and (3) correspond to the equation used to calculate the energy consumption [33].

$$P(u) = k \cdot P_{max} + (1 - k) \cdot P_{max} \cdot u \tag{2}$$

In this model, k shows the energy consumed when the server in DC is in the idle state that consumes 70% of the maximum power of the server. *Pmax* exhibits the maximum power consumption when the host is 100% utilized, and u denotes the utilization of CPU. At different instances of time, CPU utilization varies according to the workload on the host [33] as shown in Equation (3):

$$E = \int_{t0}^{t1} P(u(t))dt \tag{3}$$

where $u(t)$ corresponds to the utilization of CPU at time t. CloudSim has two servers used for power consumption calculation: one is HP ProLiant ML110 G4 and the other is HP ProLiant ML110 G5 [39]. Figure 4 demonstrates the energy consumption in the DC for the proposed *EAMA* and existing approaches (i.e., *PACPA* [11] and *RUAEE* [35]). The improvement in resource utilization also has a clear impact on the energy consumption. The proposed EAMA approach reduced the energy consumption by approximately 13%

as compared to the existing (PACPA and RUAEE) approaches. This is due to the fact that migration was performed in a controlled way by firstly migrating the VMs from the overloaded PMs to the lowest under-loaded PMs. Then, the rest of the under-loaded PMs are divided into two halves, where VMs from one set of under-loaded PMs are migrated to another set of under-loaded PMs.

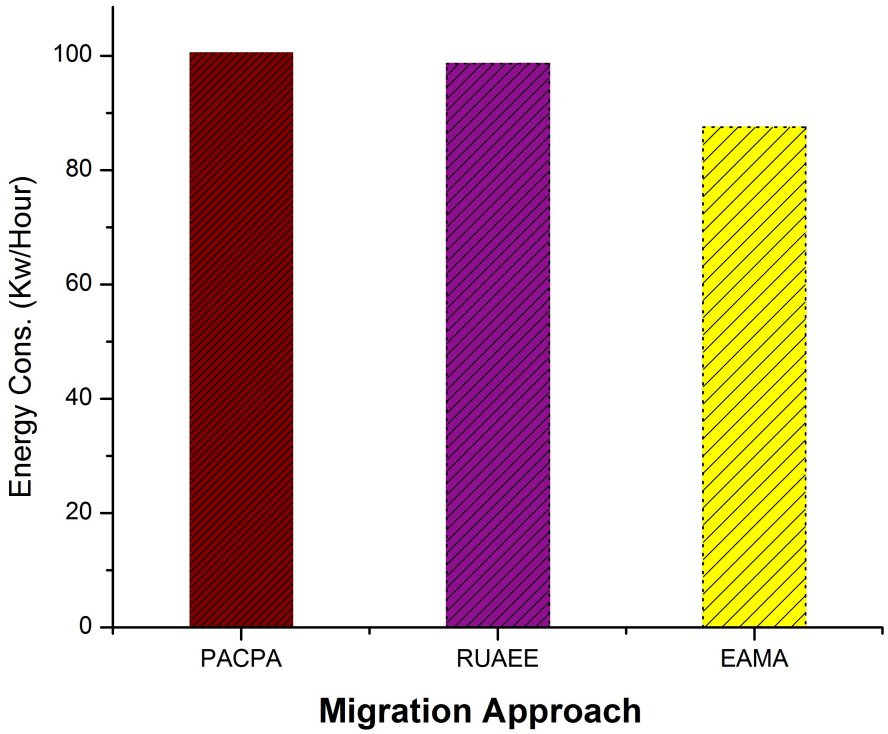


Figure 4. Energy consumption comparison.

SLA violation occurs when the hosts reach their 100% CPU utilization and the user application still requires more CPU for its execution. Equation (4) corresponds to the SLA violation calculation [33]. In the given equation, SLA violation is calculated with respect to time:

$$SLA \quad Voilation = \sum_{i=1}^{n} \frac{T_{si}}{T_{ai}} \tag{4}$$

where T_{si} is the time when the host is in violation and T_{ai} shows the hosts total active time [33]. Figure 5 plots the results of SLA violation for the EAMA and compared approaches. The proposed approach reduces the SLA violation by 20% and 34% as compared to the PACPA and RUAEE, respectively. The obtained results advocates the effectiveness of the proposed approach for VM migration in the Cloud DCs.

Figure 6 plots the results concerning the number of hosts shutdown. One of the objectives of the proposed EAMA approach is to reduce the number of active hosts in the Cloud DC. The proposed approach leads to a reduction in the number of active hosts. It can be seen from Figure 6 that the proposed EAMA approach reduced the number of hosts shut down by 10% and 13% as compared to PACPA and RUAEE approaches, respectively.

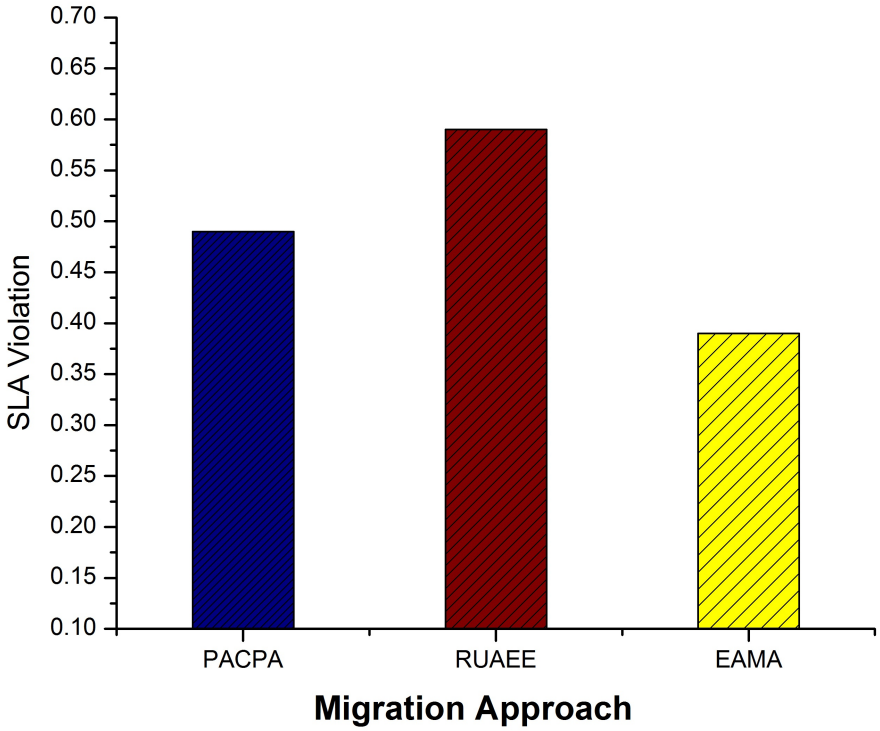


Figure 5. Service-Level Agreement (SLA) violation.

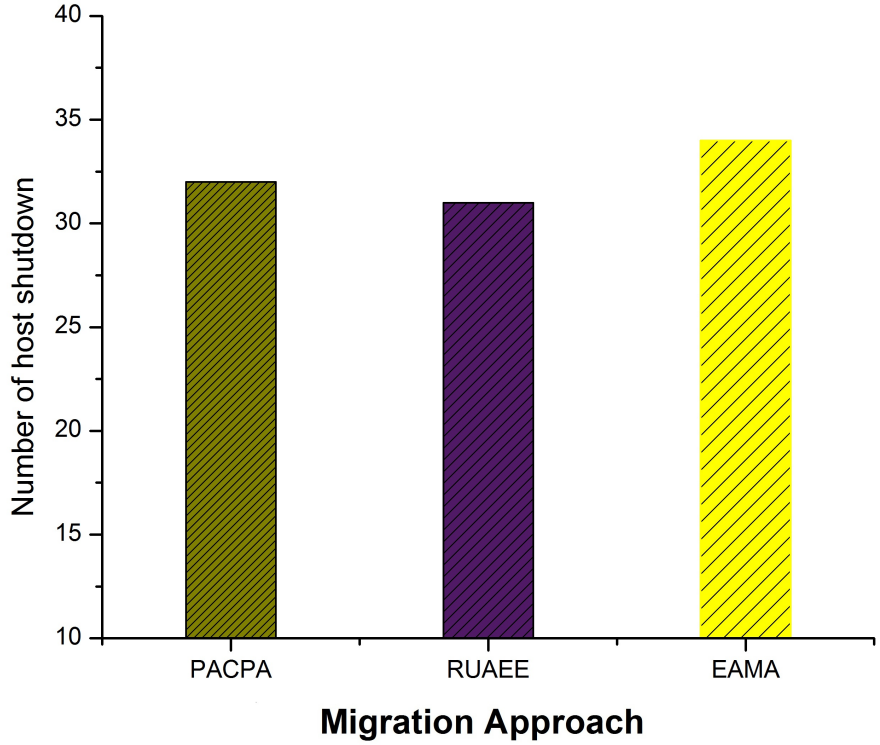


Figure 6. Number of hosts shut down.

Finally, Figure 7 provides details regarding the number of VM migrations. From the obtained results, 16% and 24% reductions in the number of migrations is observed for the proposed approach as compared to PACPA and RUAEE, respectively. The number of migrations is affected by the variation of resources required by the VMs. The reduction

in VM migrations is achieved by controlled migration across the CDCs. All these results advocates the adoption of EAMA approach for the VM migration across the CDCs.

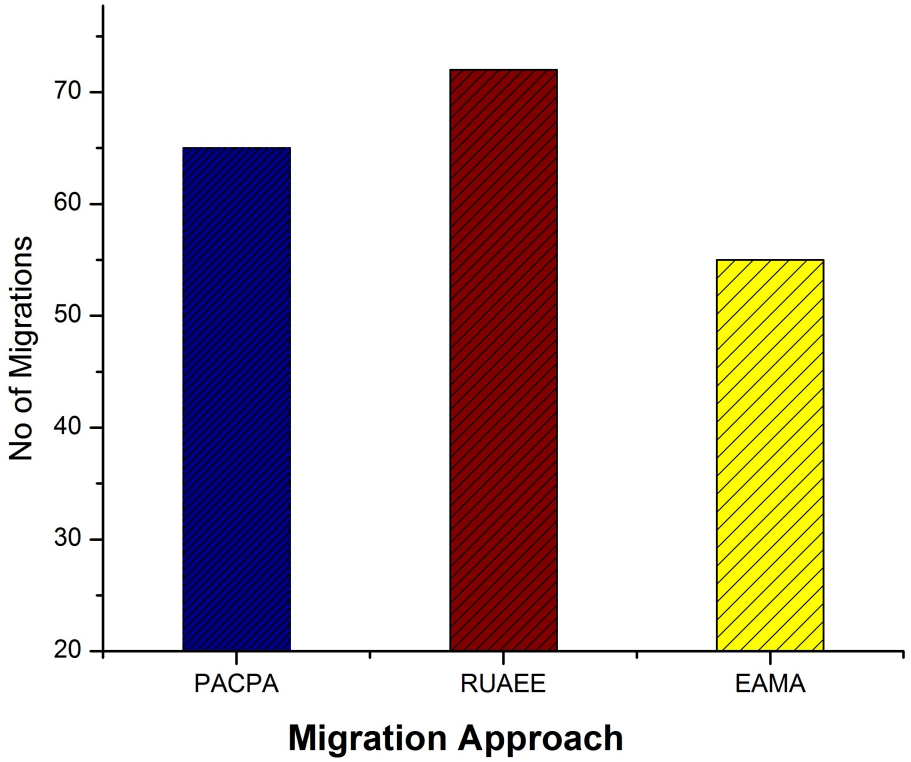


Figure 7. Number of migrations.

6. Comparative Evaluation

Following the experimental results and discussion, the comparative evaluation is delineated to extract important findings regarding the proposed approach against the compared migration approaches. The behavior of three different migration approaches including the proposed EAMA approach was investigated empirically concerning various performance metrics. Among the compared approaches, the proposed approach showed substantial performance concerning the number of migrations, resource utilization, SLA violation, host shutdown ratio, and energy consumption. The efficiency of any migration approach depends on the strategy used for the VM migrations. The proposed approach has been able to reduce the number of migrations by 16% and 24% against the PACPA and RUAEE, respectively. This is because migration was performed in a controlled way by first migrating the VMs from the overloaded PMs to the lowest under-loaded PMs. Then, the rest of the under-loaded PMs are divided into two halves where VMs from one set of under-loaded PMs are migrated to another set of under-loaded PMs. Migration in this way has three benefits: First, the number of hosts shutdown is increased, thus leading to better resource utilization; second, the number of migrations is also reduced; and third, reducing the chances of powering on the new PMs. From the user's point of view, the SLA is considered as one of the most important criteria in selecting the best Cloud service provider. Another important improvement is observed in the form of reduced SLA violation by 20% and 34% as compared to the PACPA and RUAEE, respectively. The RUAEE approach performance load balancing each time a VM is migrated from the overloaded/under-loaded host, thus leading to better resource utilization; however, it may result in SLA violation. On the other hand, the PACPA approach continuously monitors the resources and keeps this information in history for future prediction. The PACPA approach has improved SLA violation due to its predictive behavior to optimally select the best PM and time of VM migration. The Cloud Service Providers (CSPs) invest a huge amount of

capital and effort, thus finding ways to increase revenue generation. The efficient resource utilization of the existing Cloud resources is important to increase the service provisioning ratio and satisfy the SLA for the clients. The careful and localized migration strategy of the proposed EAMA approach has not only led to a reduction in the SLA violation, but also led to better resources utilization. In line with this objective, the proposed approach has been able to improve the resource utilization by 8% to 17% against the state-of-the-art compared migration approaches. The PACPA approach has been able to produce better resources utilization as compared to the RUAEE approach due to its predictive nature considered for optimal migration of the VMs. The efficient resource utilization has several impacts like the maximal utilization of the existing resources, powering off the underutilized machines, and most importantly resulting in improved energy consumption. The proposed approach not only reduced the number of migrations, but also led to better consolidation of the resources. The EAMA approach reduced the number of hosts shut down by 10 to 13% as compared to the contemporary approaches (i.e., PACPA and RUAEE, respectively). For the last few years, energy consumption has been considered an important parameter to evaluate the performance of various approaches concerning the Cloud computing. This work also utilized the energy consumption metric to evaluate the performance of the proposed EAMA approach against the approaches considered for comparative analysis. The proposed EAMA approach reduced the energy consumption by approximately 13% as compared to the PACPA and RUAEE approaches. The results of EAMA are promising, and it can be concluded from the results that the proposed approach is a suitable competitor for VM migration across the CDCs.

7. Conclusions and Future Work

Energy consumption and environmental sustainability of modern CDCs have become a major concern for Cloud service providers (CSPs). Due to the increasing demands for Cloud services, CSPs are interested in the realization of energy-efficient methods to significantly reduce energy consumption. In this paper, we present a EAMA VM migration and placement approach that significantly improves the energy consumption and QoS provisioning in CDCs. The proposed approach is based on the principle of migrating VMs from overloaded hosts to under-loaded hosts while keeping the location (of VMs/users) intact with less delay that ultimately leads to a reduction in energy consumption. The obtained results demonstrate that our approach reduces energy consumption by 13%, reduces SLA violation by 15%, the number of hosts shutdown by 10–13%, the number of migrations by 19%, and resource utilization by 20% as compared to the existing approaches (PACPA and RUAEE).

For the last few years, machine learning approaches have been widely used for accurate prediction and intelligent decision-making. During the VM migration, one important point is to perform the VM migration in a way to provide minimum SLA violation to the Cloud users. Inline with this objective, in our future work, we intend to propose a machine learning-based resource-aware and load-balanced VM migration approach with an aim to reduce the SLA violation to improve resource utilization.

Author Contributions: Conceptualization, M.I. (Muhammad Ibrahim); M.I. (Muhammad Imran); Methodology, M.I. (Muhammad Ibrahim); M.I. (Muhammad Imran); Validation,M.I. (Muhammad Ibrahim); M.I. (Muhammad Imran); Writing—original draft, M.I. (Muhammad Ibrahim); M.I. (Muhammad Imran); Writing—review & editing, M.I. (Muhammad Ibrahim); M.I. (Muhammad Imran) and D.-H.K.; F.J.; Visualization F.J.; Funding acquisition D.-H.K. and Y.-J.L.; Supervision, D.-H.K. and Y.-J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: This research was supported by Energy Cloud R&D Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT (2019M3F2A1073387), and this research was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-01456, AutoMaTa: Autonomous Management framework based on artificial intelligent Technology for adaptive and disposable IoT). Any correspondence related to this paper should be addressed to Do-hyeun Kim.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abbas, G.; Mehmood, A.; Lloret, J.; Raza, M.S.; Ibrahim, M. FIPA-based reference architecture for efficient discovery and selection of appropriate cloud service using cloud ontology. *Int. J. Commun. Syst.* **2020**, *33*, e4504. [CrossRef]
2. Koomey, J. Growth in data center electricity use 2005 to 2010. In *A Report by Analytical Press, Completed at the Request of The New York Times*; 2011; Analytics Press: Oakland, CA, USA, volume 9, p. 161.
3. Patel, C.D.; Bash, C.E.; Sharma, R.; Beitelmal, M.; Friedrich, R. Smart cooling of data centers. In *ASME 2003 International Electronic Packaging Technical Conference and Exhibition*; American Society of Mechanical Engineers: New York, NY, USA, 2003; pp. 129–137.
4. Open Compute Project. Available online: http://opencompute.org/ (accessed on 15 April 2021).
5. Ashrae, T. *Datacom Equipment Power Trends and Cooling Applications*; American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.: Atlanta, Georgia, 2005.
6. De Assuncao, M.D.; Gelas, J.-P.; Lefevre, L.; Orgerie, A.-C. The green grid'5000: Instrumenting and using a grid with energy sensors. In *Remote Instrumentation for eScience and Related Aspects*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 25–42.
7. Gartner, I. Gartner Estimates Ict Industry Accounts for 2 Percent of Global co2 Emissions. *Press Releases*. Available online: http://www.gartner.com/it/page.jsp (accessed on 15 April 2021).
8. Tarahomi, M.; Izadi, M. A prediction-based and power-aware virtual machine allocation algorithm in three-tier cloud data centers. *Int. J. Commun. Syst.* **2019**, *32*, e3870. [CrossRef]
9. Cui, H.; Zhang, B.; Chen, Y.; Yu, T.; Xia, Z.; Liu, Y. Sdn-based optimization model of virtual machine live migration over layer 2 networks. In *Advances in Computer Communication and Computational Sciences*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 473–483.
10. Alharbi, F.; Tian, Y.-C.; Tang, M.; Zhang, W.-Z.; Peng, C.; Fei, M. An ant colony system for energy-efficient dynamic virtual machine placement in data centers. *Expert Syst. Appl.* **2019**, *120*, 228–238. [CrossRef]
11. Shaw, R.; Howley, E.; Barrett, E. An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions. *Simul. Model. Pract. Theory* **2019**, *93*, 322–342. [CrossRef]
12. Khan, A.A.; Zakarya, M.; Khan, R. Energy-aware dynamic resource management in elastic cloud datacenters. *Simul. Model. Pract. Theory* **2019**, *92*, 82–99. [CrossRef]
13. Han, Z.; Tan, H.; Wang, R.; Chen, G.; Li, Y.; Lau, F.C.M. Energy-efficient dynamic virtual machine management in data centers. *IEEE/ACM Trans. Netw. (TON)* **2019**, *27*, 344–360. [CrossRef]
14. Xu, H.; Liu, Y.; Wei, W.; Xue, Y. Migration cost and energy-aware virtual machine consolidation under cloud environments considering remaining runtime. *Int. J. Parallel Program.* **2019**, *47*, 481–501. [CrossRef]
15. Li, L.; Dong, J.; Zuo, D.; Wu, J. Sla-aware and energy-efficient vm consolidation in cloud data centers using robust linear regression prediction model. *IEEE Access* **2019**, *7*, 9490–9500. [CrossRef]
16. Cao, G. Topology-aware multi-objective virtual machine dynamic consolidation for cloud datacenter. *Sustain. Comput. Inform. Syst.* **2019**, *21*, 179–188. [CrossRef]
17. Nathuji, R.; Schwan, K. Virtualpower: Coordinated power management in virtualized enterprise systems. In *ACM SIGOPS Operating Systems Review*; ACM: New York, NY, USA, 2007; Volume 41, pp. 265–278.
18. Arroba, P.; Moya, J.M.; Ayala, J.L.; Buyya, R. Dynamic voltage and frequency scaling-aware dynamic consolidation of virtual machines for energy efficient cloud data centers. *Concurr. Comput. Pract. Exp.* **2017**, *29*, e4067. [CrossRef]
19. Reddy, V.D.; Gangadharan, G.; Rao, G.S.V. Energy-aware virtual machine allocation and selection in cloud data centers. *Soft Comput.* **2019**, *23*, 1917–1932. [CrossRef]
20. Mazumdar, S.; Pranzo, M. Power efficient server consolidation for cloud data center. *Future Gener. Comput. Syst.* **2017**, *70*, 4–16. [CrossRef]
21. Li, X.; Garraghan, P.; Jiang, X.; Wu, Z.; Xu, J. Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy. *IEEE Trans. Parallel Distrib. Syst.* **2017**, *29*, 1317–1331. [CrossRef]
22. Basu, S.; Kannayaram, G.; Ramasubbareddy, S.; Venkatasubbaiah, C. Improved genetic algorithm for monitoring of virtual machines in cloud environment. In *Smart Intelligent Computing and Applications*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 319–326.
23. Murad, O.; Jabri, R.; Mahafzah, B.A. A metaheuristic approach for static scheduling based on chemical reaction optimizer. *J. Theor. Appl. Inf. Technol.* **2019**, *97*, 3144–3165.
24. Karda, R.K.; Kalra, M. Bio-inspired threshold based vm migration for green cloud. In *Advances in Data and Information Sciences*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 15–30.

25. Masadeh, R.; Sharieh, A.; Mahafzah, B. Humpback whale optimization algorithm based on vocal behavior for task scheduling in cloud computing. *Int. J. Adv. Sci. Technol.* **2019**, *13*, 121–140
26. Fatima, A.; Javaid, N.; Sultana, T.; Hussain, W.; Bilal, M.; Shabbir, S.; Asim, Y.; Akbar, M.; Ilahi, M. An efficient virtual machine placement via bin packing in cloud data centers. In *International Conference on Advanced Information Networking and Applications*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 977–987.
27. Li, H.; Li, W.; Zhang, S.; Wang, H.; Pan, Y.; Wang, J. Page-sharing-based virtual machine packing with multi-resource constraints to reduce network traffic in migration for clouds. *Future Gener. Comput. Syst.* **2019**, *96*, 462–471. [CrossRef]
28. Aldossary, M.; Djemame, K.; Alzamil, I.; Kostopoulos, A.; Dimakis, A.; Agiatzidou, E. Energy-aware cost prediction and pricing of virtual machines in cloud computing environments. *Future Gener. Comput. Syst.* **2019**, *93*, 442–459. [CrossRef]
29. Mahafzah, B.A.; Jaradat, B.A. The load balancing problem in OTIS-Hypercube interconnection networks. *J. Supercomput.* **2008**, *46*, 276–297. [CrossRef]
30. Mahafzah, B.A.; Jaradat, B.A. The hybrid dynamic parallel scheduling algorithm for load balancing on chained-cubic tree interconnection networks. *J. Supercomput.* **2010**, *52*, 224–252. [CrossRef]
31. Yang, C.-T.; Chen, S.-T.; Liu, J.-C.; Chan, Y.-W.; Chen, C.-C.; Verma, V.K. An energy-efficient cloud system with novel dynamic resource allocation methods. *J. Supercomput.* **2019**, *75*, 4408–4429. [CrossRef]
32. Nashaat, H.; Ashry, N.; Rizk, R. Smart elastic scheduling algorithm for virtual machine migration in cloud computing. *J. Supercomput.* **2019**, *75*, 3842–3865. [CrossRef]
33. 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*, 755–768. [CrossRef]
34. Ibrahim, M.; Iqbal, M.A.; Aleem, M.; Islam, M.A. SIM-Cumulus: An Academic Cloud for the Provisioning of Network-Simulation-as-a-Service (NSaaS). *IEEE Access* **2018**, *6*, 27313–27323. [CrossRef]
35. Han, G.; Que, W.; Jia, G.; Zhang, W. Resource-utilization-aware energy efficient server consolidation algorithm for green computing in iiot. *J. Netw. Comput. Appl.* **2018**, *103*, 205–214. [CrossRef]
36. Ibrahim, M.; Nabi, S.; Hussain, R.; Raza, M.S.; Imran, M.; Kazmi, S.A.; Oracevic, A.; Hussain, F. A Comparative Analysis of Task Scheduling Approaches in Cloud Computing. In Proceedings of the 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), Melbourne, VIC, Australia, 11–14 May 2020; pp. 681–684.
37. Ibrahim, M.; Nabi, S.; Baz, A.; Naveed, N.; Alhakami, H. Toward a Task and Resource Aware Task Scheduling in Cloud Computing: An Experimental Comparative Evaluation. *Int. J. Netw. Distrib. Comput.* **2020**, *8*, 131–138. [CrossRef]
38. Ibrahim, M.; Nabi, S.; Baz, A.; Alhakami, H.; Raza, M.S.; Hussain, A.; Salah, K.; Djemame, K. An in-depth Empirical Investigation of state-of-the-art Scheduling Approaches for Cloud Computing. *IEEE Access* **2020**, *8*, 128282–128294. [CrossRef]
39. Standard Performance Evaluation Corporation. Available online: http://www.spec.org/power_ssj2008/ (accessed on 15 April 2021).