New Energy Efficient and SLA-Aware Cloud Resource Management Methodology

Gunjan Bhatnagar[1], Luxmi Sapra[2], Ankit Mathani[3]

[1, 2]Assistant Professor, Computer Science & Engineering, School of Computer Science & Engineering, Dev Bhoomi Uttarakhand University, Chakrata Road, Manduwala, Naugaon, Uttarakhand 248007

[3]Associate Professor, Computer Science & Engineering, School of Computer Science & Engineering, Dev Bhoomi Uttarakhand University, Chakrata Road, Manduwala, Naugaon, Uttarakhand 248007

[1]socse.gunjan@dbuu.ac.in, [2]socse.luxmisapra@dbuu.ac.in, [3]socse.ankit@dbuu.ac.in

Article Info
Page Number: 2567-2581
Publication Issue:
Vol. 71 No. 4 (2022)

Article History
Article Received: 25 March 2022
Revised: 30 April 2022
Accepted: 15 June 2022
Publication: 19 August 2022

Abstract
A well-known method for consolidating and placing virtual machines (VMs) is server virtualization, which has been thoroughly investigated by a number of researchers. This article describes a revolutionary strategy known as an iCloud that promotes categorizing hosts or physical machines (PMs) into four distinct classes to make it easier to pick PMs quickly and decrease the amount of time needed to search host machines (also known as host search time) (HST). In addition, the framework adds VM Acceptance State, a condition that prevents host overloading and significantly lowers SLA time per active host (SLATAH), which in turn lowers the risk of SLA violation (SLAV). Using typical workload traces, the performance of iCloud has been compared to that of alternative methods. An appealing solution for effective management of cloud resources is shown by the empirical assessment that iCloud has the lowest HST and outperforms other ways in terms of SLA violations and ESV (Energy and SLA Violation).

Keywords: Heuristic algorithms, host search time, server virtualization, service level agreements, virtual machines, and virtualized resources are terms used to describe cloud computing, cloud data centers, and cloud resource management.

1. INTRODUCTION

The term "cloud" refers to a broad category of virtualized resources, including hardware (processing power, memory, storage, and bandwidth), platforms for software development, and services, that may be accessed by users across a network on demand. With the aid of scalability and load balancing, these virtualized resources are dynamically adjusted in accordance with varying demands. Large cloud data centers with thousands of servers, such as those operated by Amazon, Google, Microsoft, IBM, and others to mention a few, have been created throughout the world to meet the rising demand for computing resources from clients. Every data center uses a significant quantity of electricity for its cooling systems, network equipment, and servers. According to research issued by Digital Power Group, power consumption increased continuously from 2010 to 2015 by at least 100%, and from 2015 to

2020, it is expected to increase by at least 80%. (Mills, 2013). Large data centers emit carbon dioxide (CO2), which has an impact on the environment. The ineffective utilization of computer resources in data centers may be the cause of this excessive energy usage. However, since applications often face unpredictable workloads that call for dynamic resource utilization, resource management poses the biggest problem for data centers. Applications may have issues with longer response times or crashes if resource demands are not met.

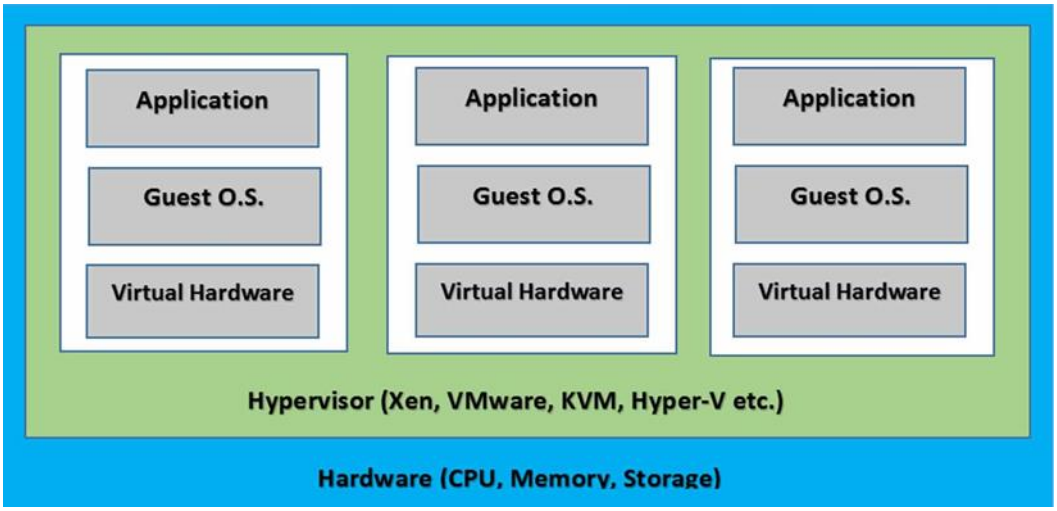

Figure 1. VM placement

In order to guarantee customers' Quality of Service (QoS), data centers must implement Service Level Agreements (SLA). Therefore, data centers need effective resource management strategies that address the trade-off between energy performance and lowest energy usage and SLA violation. Server virtualization is the foundation for resource management solutions, which must handle the resource management issue from two angles. placement of virtual machines (VM) and consolidation of virtual machines (VM). By addressing SLA issues between clients and cloud service providers, VM Placement is the process of mapping virtual machines (VMs) to the appropriate PM (Physical Machine) or host depending on the current resource needs of the VMs, as illustrated in Figure 1. Let m be the number of VMs to be put and let n be the number of PMs in the data center. There are nm many ways that m VMs might be mapped to n PMs (He & Guo, 2011). Therefore, it is very difficult, if not impossible, to manually investigate every potential mapping and choose the optimal one for a big number of n and m. Therefore, automation of VM deployment is greatly desired. The process of consolidating virtual machines (VMs) involves moving them to other PMs in order to meet their rising runtime resource demands, which are otherwise incompatible with those of their underlying PMs. Making the right choices about VM placements and consolidations over their life cycles might help with data center resource efficiency. Different VM placement and consolidation techniques have been created by researchers, with trade-offs made between many competing performance factors including power use, SLA violation, number of VM migrations, Host Search Time, etc. The unique method described in this research, known iCloud, enables sensible VM placement and consolidation choices.

2. LITERATURE REVIEW

1. By using virtualization to map a collection of applications onto a set of host servers, resource allocation during application deployment in the data center is modeled. Numerous academics have investigated this issue and offered numerous solutions in an effort to improve performance metrics like energy usage, SLA violation, cost, etc. Researchers have suggested techniques for effective resource management, including cost reduction, performance, availability, traffic patterns, and energy conservation.

2. For some academics, cost savings is the main consideration for more effective use of computer resources. In order to save money and improve the usage of computer resources, Hyser et al. (2007) offered a high-level overview of VM placement and suggested a system architectural design of an autonomous VM placement. A policy for resource management is, however, lacking. The best virtual machine placement technique put out by Chaisiri et al. (2009) might reduce the amount of money spent on resource provisioning in on-demand and reservation-based schemes. The goal of this study is to employ as few servers or nodes as possible. An autonomous resource manager was created by Nguyen et al. (2009) and includes a preset set of virtual machine classes depending on resource capacity. Each VM class has a specified set of CPU and memory specifications. Among a list of preset VM classes, a suitable VM has been chosen for applications based on the present workload. It seeks to maximize a global utility function that incorporates the level of SLA fulfillment and operational expenses.

3. Another factor that many researchers take into account for the VM placement issue is performance. By evenly dividing the application loadoveralll physical servers, this may be improved. Building a framework with a load balancing policy, in which data center load is distributed among all available physical servers by balancing resource usage as much as possible across all resource types such as processing power, memory, bandwidth, storage, etc., was the main goal of research work in (Hyser et al., 2007). Goudarzi et al. (2012) developed a similar load balancing strategy in which several VM instances are set up on various servers and incoming requests are divided among them. This method lowers the amount of resources needed for each VM instance and aids the cloud provider in making better use of servers. However, using such methods adds to the work required to keep many VM instances coordinated and consistent. This method may also result in resource fragmentation, which is the underutilization of resources at certain servers under mild demand. By suggesting a method that chooses virtual machines with the least amount of interference from one another and groups them on the same server, Roytman et al. (2013) focused on performance deterioration that happens as a result of resource competition among VMs. Local Decision Module (LDM) uses the performance model developed by Nguyen et al. (2009)'s autonomous resource manager to evaluate the level of service delivered with a certain resource capacity and the current application workload. The main disadvantage of this strategy is the potential for over-provisioning of resources allotted to a certain VM's application. Gupta et al. first described the original VM placement challenge for deploying high-performance Computing (HPC) applications (2012). The location of VMs for HPC

applications may be optimized using topology and hardware awareness approaches, according to this study. However, run time placement with live VM migration is not taken into account in their work as the resource requirements of the application grow. More frequent VM migrations might cause the system's performance to suffer. Sharma et al. (2017) suggested dynamic consolidation or resource management solutions that avoid needless VM migration while providing performance efficiency. Additionally, it cuts down on SLA breaches and the total length of the VM consolidation process.

4. The availability of cloud applications is a crucial consideration in cloud data centers, in addition to cost savings and performance. According to Goudarzi et al., maintaining several copies of VMs on various PMs may preserve the availability of cloud applications (2012). It lowers the amount of resources needed for each copy and increases server efficiency. However, the problem of preserving consistency and coordination amongst several copies of VMs is not the major emphasis of this study. Jayasinghe et al. (2011) devised an approach that, by distributing VMs across various data center isolation levels, increases service availability while also enhancing performance.

5. By taking into account the traffic patterns between them, VM placement may be maximized. The same physical host or nearby host machines are allocated to virtual machines with high mutual bandwidth utilization (Meng, Pappas, & Zhang, 2010). However, because virtual machine traffic load is essentially dynamic, it would be challenging to estimate the cost of communication or traffic between two VMs. Researchers have developed a dynamic virtual machine placement technique in (Dias & Costa, 2012; Vu & Hwang, 2014) to reallocate VMs across servers in a data center based on the current traffic matrix in the data center. Virtual machine migrations may result in dynamic changes to a data center's traffic matrix. A traffic matrix is built using the traffic that is transferred between each pair of virtual computers.

6. One of the most important modern resource management characteristics is energy conservation (Beloglazov, Abawajy, & Buyya, 2012). Energy conservation is essential for reducing CO2 and greenhouse gas (GHG) emissions in addition to lowering electricity costs (Bilal, Khan, & Zomaya, 2013; Khosravi, Garg, & Buyya, 2013). Reducing the number of active (running) physical servers in data centers might easily reduce energy usage in such facilities (Beloglazov et al., 2011; Buyya et al., 2009; Li et al., 2013; Wang et al., 2012). Applications are loaded onto virtual machines, therefore several types of resources are needed, including CPU cores, memory, bandwidth, and storage space. A VM is only hosted on a certain host server or PM when there are enough resources for all dimensions. As a result, certain PM can have resource pieces, which are unused resources. A method termed EAGLE, developed by Li et al. (2013), decreases the quantity of PMs and the sizes of resource pieces. However, apps could use more resources under high load, hence this study has not taken into account their potential future resource demands. Beloglazov and Buyya (2012) introduced unique adaptive heuristics for VM consolidation that are both energy and performance efficient. These heuristics anticipate future resource use based on a study of previous data. The majority of these studies used the assumption that all physical servers in

data centers are uniform, although server configurations in data centers vary widely, which may affect how much energy they consume. In order to build a VM placement strategy that reduces energy consumption and enhances QoS by taking into account diverse servers available in National Cloud Data Centers, Wang et al. (2016) investigated the Particle Swarm Optimization (PSO) method (NCDC).

7. The virtualized architecture present in the data center is used to meet the resource needs of cloud applications. Due to the dynamic nature of application demand, allotted resources must be scaled up or down. To meet these needs of web applications, the widely utilized technology of VM-based dynamic scaling is used. The implementation of VM-based scaling may be done in one of two ways: by increasing the number of VM instances on servers or by altering how resources (such as processing power, memory, and storage) are divided within a VM (Wang et al., 2012). Both of these scaling methods are known as horizontal scaling and vertical scaling, respectively. Vertical scaling increases computing power by providing more resources for virtual machines, while horizontal scaling increases computing power by adding more virtual machines (Liu, Shie, Lee, Lin, & Lai, 2014). The effects of vertical and horizontal scaling approaches on resource management have been contrasted by Wang et al. (2012). While horizontal scaling increases application availability generally, vertical scaling provides the benefit of performance. Vertical scaling was used in the Jadhav et al. (2015) built eNlight cloud to prevent over-provisioning of resources for applications and significantly lower costs for clients. A list of notable studies that have been published in the literature is included in Table 1. As is evident, every described piece of work focuses on a certain performance metric (s).

8. A technique that balances several competing performance metrics, such as energy consumption, SLA violation, and cost, is both a fascinating and difficult challenge, according to the literature review that was done. The following research concerns are addressed in the work presented here:

9. A plan for consolidating and placing VMs that will shorten the time needed to find the right PM or host;

10. To prevent the issue of overloading, a system for the deployment of VM into PM must be established;

11. The creation of a plan to quicken the deployment of VMs;

12. To minimize performance deterioration brought on by migration, a method must be designed to delay VM migration as long as feasible throughout the VM consolidation process.

Table 1. Survey on cloud resource management

Publication	Year	Resource Management Characteristics					
		CA	PA	AA	TA	EA	SA
Chase et al.	2009	√	√			√	

Hyser et al.	2009	√					
Chaisiri et al.	2009	√	√				
Nguyen et al.	2010						
Li et al.	2011					√	
Meng et al.	2011		√		√		
Jayasinghe et al.	2011			√			
Buyya et al.	2011		√			√	
Marston et al	2012					√	
Shen et al.	2012						√
Goudarzi et al.	2012		√	√		√	√
Gupta et al.	2012		√			√	√
Dias et al.	2012					√	
Wang et al.	2013		√			√	
Beloglazov et al.	2013		√				
Roytman et al.	2013					√	
Bilal et al.	2013					√	
Khosravi et al.	2013					√	
Li et al.	2013					√	
Mastroianni et al.	2014		√				
Zhuang et al.	2014	√				√	√
Hieu et al.	2014				√		
Liu et al.	2015						√
Teyeb et al.	2016				√		
Jadhav et al.	2017		√			√	
Wang et al.	2018		√				
Sharma et al.	2018						

CA – Cost Aware, PA-Performance Aware, AA-Availability Aware, TA- Traffic Aware, EA-Energy Aware, SA-Scalability Aware

13. Due to the unpredictable nature of demand patterns in applications, an algorithm should be devised to scale up or scale down the resources given to VMs.

3. SUGGESTIVE SYSTEM

The proposed method known as iCloud takes into account a data center with several heterogeneous PMs or hosts with varying resource capabilities. In a data center, PMs or hosts are divided into four distinct types or states: "Offline," "Target," "Greedy," and "Contented" (Shelar, Sane, Kharat & Jadhav, 2014, 2017). All PMs start out in the "Offline" class and go on to the subsequent classes as described below:

1. Offline class: This class includes all PMs that are in a power-saving mode of operation. Since idle PMs that are in an online state use at least 50% of their peak power, it is preferable to maintain idle hosts or PMs in a data center in power-saving (offline) states to guarantee energy efficiency (Beloglazov, Buyya, Lee, & Zomaya, 2011);

2. Specified class: During initial setup, virtual machines need extra resources in order to install the guest operating system and client applications. As a result, certain hosts or PMs claim that they may always be kept available just for setup and installation purposes, and these PMs fall within the target class. To expedite setup and installation, every VM is first deployed on one of the PMs from the Target class;

3. Greedy class: This class only includes PMs whose resource allocation falls short of a predetermined resource cap (RCAP), also known as an upper threshold. RCAP sets a limit on how much of a resource PMs may use. Equation 1 below states that the total amount of resources of each kind Rk allotted to the m virtual machines running on PM Pi should be less than the resource cap:

(TOTALik * RCAPk) = (PLACEij VALLOCjk)

j =1 k =1

(1) with d denoting the number of resource dimensions. Similar to the research described in (Beloglazov et al., 2012), this study simply takes into account the CPU as a resource dimension.

The terms VALLOCjk and TOTALik in Equation 2 provide the resource requirements of VM Vj of resource type Rk, the total resource capacity of PM Pi of resource Rk, and whether VM Vj is deployed on PM Pi or not.

ACEij ∈ {0,1}

(2) Greedy class makes it easier for PMs to do a restricted search for VM migration.

1. Contented class: Members of this class have received resources up to their selected resource maximum. Even if just a small amount of resources have yet been allocated to enable dynamic scaling, PMs of this type are regarded to be complete or full and no further VMs will be deployed on such PMs.

3.1. AI Cloud algorithm

Algorithm 1 depicts how iCloud works generally. As shown, iCloud establishes a unique thread for every new VM. When necessary, newly created threads carry out VM placements on the proper PM (algorithm 2), which in turn triggers VM consolidation (algorithm 3).

Algorithm for VM Placement, Section 3.

This approach uses a notion known as "VM acceptance state" to put the VM on the proper PM.

• VM Acceptance State: This is a prerequisite (described in Equation 3) that has to be met before VM Vj is put on PM. Pi: iCloud, algorithm 1. ()

(TOTALik RCAPk) = (PALLOCik + REQUESTjk) (3) where:

• TOTALik is the total resource capacity of PM Pi for resource Rk; • PALLOCik is the resources of type Rk that are already assigned from PM Pi; • REQUESTjk is the request of VM Vj for resource type Rk; • RCAPk is the resource-cap specified for resource type Rk for all PMs like 0.5, 0.6, 0.7, etc. Only when the resource Rk's total resource allocation on PM Pi (PALLOCik + REQUESTjk) does not exceed its resource cap (TOTALik RCAPk) will the condition "VM Acceptance State" be true. This idea makes it easier to choose the best PM without overtaxing the host. Virtual machine allocation using iCloud is shown in algorithm 2. The target class's first new VM is generated in the proper PM Pi when it reaches the "VM acceptance state." A PM from the Offline class is chosen and switched on to make the transition to the Target class if there are no such PMs in the Target class. The chosen PM allows lots the necessary resources to the chosen VM, which then configures the operating system and applications. After completing the setup process, the VM in PM Pi from the Target class is migrated to the suitable PM Pg of the Greedy class in accordance with the PM selection policy, such as first fit, best fit, round robin, etc. The chosen PM Pg stays in the Greedy class and is used to host one or more VMs until the total resources allocated fall below the set resource limit. Thus, it is conceivable to encounter a circumstance in which none of the PMs in the Greedy class fulfill the "VM acceptance state." In this case, the PM from the Target class is moved to the Greedy class. One of the Offline PMs will be chosen and switched on to make the transition to the Target class if there are any available when the number of PMs in the Target class is less than the threshold value t. A PM is moved to the Contented class if it has received resources up to the resource limit while in the Greedy class. Due to its spare resources being preserved for dynamic resource scaling in response to the peak demand of applications, such a PM is considered to be "full."

3.3. VM Consolidation Algorithm

Computing resources from PMs, such as processing power and memory, are assigned in accordance with their requirements during the first deployment of applications in VMs. However, since the resource demands of applications are dynamic in nature, VMs must scale up the resources they have been allotted to handle rising application loads. On the other side, resources allotted to VMs must scale down as application demand falls. Because application workload patterns might be unexpected, VM consolidation is crucial.

Algorithm 2. Virtual_Machine_placement ()

Assumptions: *n* – Number of Physical Machines in a data center with varied configurations *d* – Number of resource dimensions *t* – Target class threshold value (number of PMs to be always in Target class) *SP* – Selected Scheduling Policy for Greedy Class	
1	Find PM P_i from the Target class that satisfies '*Acceptance State'*
2	If no such PM found then
3	If there is PM in offline class then
4	Select and turn on offline PM and shift into Target class
5	Else
6	Display Error ….No PM in Offline class
7	End If
8	Else
9	Place VM at PM P_i
10	Setup of OS and applications.
11	End If
12	If VM is ready then
13	Find Greedy PM P_g as per sched policy(*SP*) satisfies '*Acceptance State'*
14	If PM P_g found then
15	Migrate VM from Target PM P_i into Greedy PM P_g
16	Else
17	Shift Target PM P_i into Greedy class
18	End If
19	If number of PMs in Target class < *t* then
20	If there is PM in Offline class then
21	Select and turn on Offline PM and Shift it to Target class
22	Else
23	Display Error …. No PM in Offline class
24	End If
25	End If
26	End If
27	While True Do
28	Call Virtual_Machine_Consolidation () //Algorithm 3
29	End While

The basics of VM consolidation employing dynamic scalability for resource allocation at runtime are shown in Algorithm 3. When VM Vj is deployed in PM Pi and its current resource needs for resource types Rk exceed the resources allotted, further resources (which are kept available owing to resource caps) are allotted from PM Pi using the vertical scaling approach. However, live VM Migration is started if there are not enough free resources on PM Pi to meet this increased demand for resources. Then, VM Vj is moved to another PM Pg in the Greedy class to meet the "VM acceptance" requirement.

state' as previously stated. If the Greedy class lacks a PM Pg that meets the "VM acceptance state," a PM from the Target class is moved into the Greedy class, and the VM Vj is migrated on this PM. As the algorithm must maintain the target-threshold value 't' in this case, a PM from the Offline class is activated and added to the Target class if the number of PMs in the Target class drops below the threshold 't'. The VM Vj is then migrated to the newly selected PM if no PM from the Target class that satisfies the "VM Acceptance state" is available. Instead, a PM from the Offline class is chosen and switched on to make the transition to the Target class. The resources on VM Vj are then scaled up by assigning more resources after the transfer of VM Vj into PM Pg.

When VM Vj's current resource demand falls below its allotted resources, it scales down by releasing resources from PM Pi, which may result in PM Pi's allotted resources falling below their resource limit. If this happens, the PM will return to the Greedy class.

Any PM in the Greedy class who has received more resources than allowed due to scale-up will be moved to the Contented class by aiCloud. In the event that none of its resources are used due to scale down, aiCloud additionally makes sure that any PM from the Greedy or Contented classes switches to the Offline class.

When compared to host overloading detection methods like Static Threshold (THR), Inter Quartile Range (IQR), Median Absolute Deviation (MAD), Local Regression (LR), and Local Regression Robust (LRR) suggested in, it is anticipated that proposed algorithms would result in minimum and lesser energy consumption (Beloglazov, & Buyya, 2012). To reduce power usage, PABFD assigns each virtual machine (VM) individually to PMs after sorting them in decreasing order of CPU use. The PABFD method for placing VMs scans a list of hosts and determines whether or not each one is overcrowded. When a host's usage goes over the upper threshold, it is recognized as being overloaded. Although the THR approach, which is based on set utilization thresholds, is straightforward to apply, the data center's unpredictable and dynamic workload patterns make fixed upper threshold values for hosts inappropriate. Hence Adaptive utilization-based algorithms, such MAD and IQR, that are based on statistical analysis of past VM data were suggested by Beloglazov et al. in 2012. Using the intensity of the divergence in CPU use during the course of the VM, MAD and IQR carry out automatic changes of the upper threshold. If the divergence is greater, the upper threshold value is lowered and the CPU usage may go to 100%, violating the SLA. Additionally, these algorithms do a poor job of anticipating host overloading (Abdelsamea,

El-Moursy, Hemayed, & Eldeeb, 2017). Since regression-based techniques like LR and LRR are based on estimates of future CPU consumption, they may provide more accurate predictions of host overloading. The complexity of these algorithms, however, is higher than that of adaptive utilization-based algorithms (Abdelsamea et al., 2017). By achieving the "VM acceptance state," aiCloud installs virtual machines (VMs) onto hosts, preventing host overloading both at initial placement and after VM consolidation.

When a host is found to be overloaded, it migrates the virtual machines to another host. The VMs that need to be moved from overloaded hosts are chosen by aiCloud using one of the VM selection rules, including Minimum Migration Time (MMT), Random Choice (RC), and Maximum Correlation (MC). In comparison to other VMs on the same host, MMT policy migrates a VM that requires the least amount of time to perform a migration. RC policy uses a uniformly distributed discrete random variable to choose one of the VMs at random for migration from an overloaded host. The MC policy chooses the VMs with the strongest correlation between their CPU consumption and that of other VMs. AiCloud checks for host overloading after migrating a chosen VM. If the host is discovered to be overloaded once again, the VM selection policy is once more used to choose another VM. Until the host is determined to not be overburdened, this procedure is repeated. As a result, aiCloud may provide an efficient technique for managing resources by balancing numerous opposing factors.

Algorithm 3. Virtual_Machine_Consolidation()

Assumptions: *n* – Number of Physical Machines in a data centre *d* – Number of resource dimensions *t* – Target class threshold value (number of PMs to be always in Target class) V_j - VMs deployed on PM P_i $TOTAL_{ik}$ – Total capacity of PM P_i of resource type R_k where 1<=*i*<=*n* & 1<=*k*<=*d* $NEED_{jk}$ - Current need of VM Vj of resource type R_k where 1<=j<=m & 1<=*k*<=*d* $PALLOC_{ik}$ – Allocated resources in PM *Pi* of type R_k where 1<=*i*<=*n* & 1<=*k*<=*d* $VALLOC_{jk}$ – Allocated resources to VM V_j of type R_k where 1<=*k*<=*d*	
1	If ($NEED_{jk} > VALLOC_{jk}$)
2	If ($TOTAL_{ik} - PALLOC_{ik}$) > $NEED_{jk}$) then //Vertical Scaling
	//Allocate additional resources for V_j from PM P_i
3	$PALLOC_{ik} \leftarrow PALLOC_{ik} + NEED_{jk}$
4	$VALLOC_{jk} \leftarrow NEED_{jk}$
5	Else //Insufficient free resources on PM P_i - Do VM Migration
6	Find PM P_g in Greedy class that satisfy Acceptance State
7	If no such PM in Greedy class found then

8	Find PM P_g from Target class that satisfy Acceptance state
9	If not such PM in Target class found then
10	If there is PM in Offline class then
11	Turn on Offline PM and shift into Target class
12	Else
13	Display Error …. No PM in Offline class
14	End If
15	Else
16	Shift Target PM P_g into Greedy class
17	If number of PMs in Target class < target-threshold *t* then
18	If there is PM in Offline class then
19	Turn on Offline PM and Shift it to Target class
20	Else
21	Display Error ….No PM in Offline class
22	End If
23	End If
24	End If
25	End If
26	Migrate VM *Vj* from PM P_i into PM P_g.
27	$PALLOC_{gk} \leftarrow PALLOC_{gk} + NEED_{jk}$
28	$VALLOC_{jk} \leftarrow NEED_{jk}$
29	End If
30	Else
31	If ($NEED_{jk} < VALLOC_{jk}$) //To Scale down resources
32	$PALLOC_{ik} \leftarrow PALLOC_{ik} - NEED_{ik}$
33	$VALLOC_{ik} \leftarrow NEED_{ik}$
34	If ($PALLOC_{ik} < TOTAL_{ik} * RCAP_k$) then //Below *RCAP*
35	Shift PM P_i to Greedy Class
36	End If
37	End If
38	End If
39	If ($PALLOC_{ik} >= TOTAL_{ik} * RCAP_k$) then //PM P_i Full
40	Shift Greedy PM P_i to Contented Class
41	End If
42	If ($TOTAL_{ik}$ - $PALLOC_{ik}$ == NULL) //PM P_i Empty
43	Shift PM P_i to Offline Class
44	End If

Table 2. Workload traces of 10 days from PlanetLab

Dates	03/03/2011	06/03/2011	09/03/2011	22/03/2011	25/03/2011	03/04/2011	09/04/2011	11/04/2011	12/04/2011	20/04/2011
Number of VMs	1052	898	1061	1516	1078	1463	1358	1233	1054	1033

4. EXPERIMENTAL SETUP AND RESULTS

Since iCloud is built on the Architecture as a Service (IaaS) concept, it is important to assess how well it performs when used with virtualized data center infrastructure. To assess the effectiveness of the suggested algorithm, a simulation platform is selected, where tests may be carried out in a controlled environment. A platform for simulating and testing new cloud computing infrastructures and application services is provided by the CloudSim toolkit (Calheiros, Ranjan, Beloglazov, De Rose, &Buyya, 2011). It was created at the CLOUDS lab at the University of Melbourne's Computer Science and Software Engineering Department in Australia. Data centers, computational resources, virtual machines, users, applications, resource schedulers, and provisioning rules are all described using Java classes.

On CloudSim, 800 servers of two different types of the HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores at 1860 MHz, 4 GB) and HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores at 2660 MHz, 4 GB) are used to mimic a data center (Beloglazov et al., 2012). Results of the SPECpower benchmark (n.d.) were used to determine the power consumption characteristics of a subset of servers, as shown in Table 2. The sorts of virtual machines that are being tested follow the use cases recommended in the Amazon EC2 (n.d.) instance types. For each kind of virtual machine, the quantity of RAM is broken down by the number of cores: High-CPU Medium Instance (2500 MIPS, 0.85 GB); Extra Large Instance (2000 MIPS, 3.75 GB); Small Instance (1000 MIPS, 1.7 GB); and Micro Instance (500 MIPS, 613 MB). At first, VMs distribute resources in accordance with the specifications set forth by the VM types. With the use of dynamic scaling, VMs consume resources during their lifespan in accordance with workload statistics. MMT is used as a VM selection policy to migrate a VM that, in comparison to other VMs assigned to the host, needs the least amount of time to accomplish a migration. The common project, a monitoring infrastructure for Planet Lab, provides the actual workload traces from 10 randomly selected days for the experiments (n.d.). This data, which can be seen in table 2, represents the CPU use of tens of thousands of virtual machines on servers spread over more than 500 different locations. It may be found on https://github.com/beloglazov/planetlab-workload-traces and on https://github.com/Cloudslab/cloudsim together with the CloudSim 3.0 toolkit. The utilization period is 5 minutes (Beloglazov et al., 2012). Each virtual machine (VM) in simulations has been given a workload trace from the relevant day. Upper thresholds or resource caps for VM allocation rules ranged between 0.5 and 1.0 with a 0.1-step increase.

To evaluate how effective the proposed iCloud method is compared to existing algorithms, median values were calculated across 10 days of workload traces. CloudSim is used to build and evaluate the suggested methods for virtual machine placement and consolidation. The outcomes are assessed and contrasted with those of the PABFD algorithm and the host overloading detection methods explained by Beloglazov et al (2012).

Performance Metrics (4.1)

The following indicators were used to assess the performance of the proposed resource management strategy in comparison to current policies.

4.1.1. Power Consumption

Power consumption by servers is described by CPU utilization as well as the amount of memory used inmulticore servers.

Table 3. Power consumption by servers at different loads (watts)

Server	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant ML110 G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant ML110 G5	93.7	97	101	105	110	116	121	125	129	133	135

Because CPU usage is often proportional to the total system load (Beloglazov et al., 2012), actual data from the SPEC power benchmark (n.d.) was used to determine power consumption at different loads as shown in Table 3.

4.1.2. Number of VM Migrations

As its name implies, VM migration count gives the total number of VM migrations required by the method for VM consolidation. It's just a counter that gets tallied up after each VM transfer. The performance of apps operating on virtual machines while the migration is taking place is negatively impacted. The frequency of VM migrations should be kept to a minimum as VM migration may potentially result in SLA violations. The HST is the total amount of time needed to locate an acceptable host for VM installation.

$$\overline{n \quad Ai}$$

4.2. Results

Tables 4 to 11 show the outcomes of simulations performed using CloudSim 3.0. Based on MMT as the VM selection policy throughout the VM migration process and first fit as the PM selection strategy in the Greedy class, all results were obtained. Upper thresholds or resource caps (RCAP) were adjusted in each trial, as indicated in Tables 4 to 10, ranging from 0.5 (50%) to 1.0 (100%) with an increase of 0.1 (10%). Average results for several strategies, including aiCloud, are shown. Energy usage is covered in Table 4, whereas VM migration count numbers are shown in Table 5. Table 6 deals with SLATAH, while Table 7 displays PDM values. Table 8 displays SLA violations for several methods, while Table 9 discusses ESV metrics. HST requirements for several methods, including aiCloud, are shown

in Table 10. Table 11 and the graphs in Figure 2 compare algorithms based on the average results of all higher threshold settings for different performance metrics.

4.3. Discussions and Observations

The suggested aiCloud uses, on average, less energy than existing algorithms (26%, 16%, and 12% less energy than THR, IQR, and MAD, respectively). Because all idle servers are initially in a power-saving condition and are only switched on when required, aiCloud saves electricity. The average number of VM migrations in aiCloud is significantly lower as compared to THR, IQR, and MAD algorithms (more than 40% of reduction as compared to THR, 34% and 29% reduction as compared to IQR and MAD respectively). However, energy consumption in aiCloud is slightly higher than LR and LRR algorithms, which may be due to predicted lower threshold. As a result, compared to these approaches, the average Performance Degradation due to Migration (PDM) is between 40 and 45% lower. The proposed aiCloud greatly exceeds all other policies in terms of average SLA Time per Active Host; nevertheless, average number of VM migrations in aiCloud is 57% and 62% more than LR and LRR algorithms owing to migrations of VM from Target to Greedy class and Greedy to Contented class; 3. (SLATAH). Average SLATAH in aiCloud is 67%, 31%, 37%, and 96% lower.

Table 4. Comparison of algorithms w.r.t. energy consumption (kWh)

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	152.7	106.76	104.17	**86.02**	86.22	86.82
60%	135.18	105.31	102.19	**84.82**	84.91	85.58
70%	121.44	104.45	100.75	**84.99**	85.08	**84.99**
80%	110.84	103.88	99.75	**86.56**	86.68	**86.56**
90%	101.77	103.56	98.88	**89**	89.1	**89**
100%	93.32	103.15	98.1	93.5	**93.25**	93.5
Average	119.21	104.52	100.64	**87.48**	87.54	87.74

Table 5. Comparison of algorithms w.r.t. VM Migration Count (VMMC) (x 10^3)

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	15.49	10.89	10.05	**3.61**	3.62	6.22
60%	12.96	10.6	9.89	**3.34**	**3.34**	5.94
70%	11.5	10.28	8.87	**3.17**	3.19	5.74
80%	10.79	10.36	9.67	**3.8**	3.87	6.45
90%	10.27	10.31	9.88	**5.13**	5.38	7.88

100%	8.45	10.09	9.51	**6.19**	6.73	8.84
Average	11.58	10.42	9.65	**4.21**	4.36	6.85

Table 6. Comparison of algorithms w.r.t. SLA Time per Active Host (SLATAH)

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	2.82%	2.77%	2.61%	61.83%	61.39%	**2.23%**
60%	3.15%	2.96%	2.71%	63.48%	63.27%	**2.23%**
70%	3.05%	2.59%	2.69%	63.20%	63.32%	**2.18%**
80%	2.89%	2.94%	2.70%	61.28%	61.04%	**2.15%**
90%	2.95%	2.98%	2.79%	53.30%	53.77%	**1.91%**
100%	21.52%	5.48%	5.59%	30.11%	30.42%	**1.14%**
Average	6.06%	2.86%	3.18%	55.53%	55.54%	**1.97%**

as compared to THR, IQR, MAD, and LR policies respectively because every VM is placed on appropriate PM by satisfying *VM acceptance state* that avoids host overloading. This lower valueof SLATAH leads to a reduction of SLA violations;

1. There is the reduction of 95%, 84%, 71%, and 69% average SLA Violation in *iCloud* as compared to LR, THR, IQR, and MAD policies respectively due to the lower value of SLATAH;

2. ESV metric shows the trade-off between energy and SLA violation. It is on average 72 to 95% lower in *iCloud* as compared to other policies due to a substantial reduction of SLA violations. The

Table 7. Comparison of algorithms w.r.t. Performance Degradation due to Migration (PDM)

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	0.06%	0.05%	0.04%	**0.01%**	**0.01%**	**0.01%**
60%	0.05%	0.05%	0.04%	**0.01%**	**0.01%**	**0.01%**
70%	0.05%	0.04%	0.04%	**0.01%**	**0.01%**	**0.01%**
80%	0.04%	0.04%	0.04%	**0.02%**	**0.02%**	**0.02%**
90%	0.05%	0.04%	0.04%	**0.02%**	**0.02%**	0.04%
100%	0.04%	0.04%	0.04%	**0.03%**	**0.03%**	0.05%
Average	0.05%	0.04%	0.04%	**0.02%**	**0.02%**	0.03%

Table 8. Comparison of algorithms w.r.t. SLA Violation (x 10^{-3})

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	1.61%	1.27%	1.12%	7.47%	7.43%	0.24%
60%	1.63%	1.34%	1.21%	7.40%	7.38%	**0.28%**
70%	1.42%	1.31%	1.17%	7.98%	8.04%	**0.22%**
80%	1.30%	1.32%	1.19%	10.16%	10.22%	**0.45%**
90%	1.41%	1.33%	1.25%	12.28%	12.81%	**0.86%**
100%	9.06%	2.47%	2.51%	8.22%	8.87%	**0.55%**
Average	2.74%	1.51%	1.41%	8.92%	9.13%	**0.43%**

Table 9. Comparison of algorithms w.r.t. ESV Metric (x 10^{-3})

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
50%	2.45	1.36	1.17	6.43	6.4	**0.21**
60%	2.2	1.41	1.23	6.27	6.26	**0.24**
70%	1.72	1.37	1.18	6.78	6.84	**0.19**
80%	1.44	1.37	1.19	8.8	8.86	**0.39**
90%	1.43	1.38	1.24	10.93	11.41	**0.77**
100%	8.46	2.54	2.46	7.69	8.27	**0.51**
Average	2.95	1.57	1.41	7.82	8.01	**0.39**

The lowest value of the ESV metric shows that *iCloud* is better with respect to energy consumption andSLA violation than all other algorithms;

3. The average HST during VM allocation in *iCloud* is at a minimum (5%, 12%, 47%, and 91% less than LRR, LR, MAD, IQR, and THR algorithms respectively) as during VM placement VMs are directly placed in appropriate PM from *Target* class only (as hosts in *Offline*, as well as Contented classes, are never searched).

Table 10. Comparison of algorithms w.r.t. Host Search Time (x 10^{3})

RCAP	THR-MMT	IQR-MMT	MAD-MMT	LR_MMT	LRR_MMT	*aiCloud*-MMT
40%	34.65	20.72	16.78	11.45	**9.96**	15.36
50%	142.12	21.33	14.25	9.16	**6.65**	9.89
60%	110.97	21.06	16.22	8.63	8.63	**8.56**

70%	96.15	21.16	23.15	9.96	8.66	**5.56**
80%	99.99	22.22	16.55	8.23	7.88	**6.72**
90%	76.22	25.54	14.32	11.12	19.66	**9.86**
100%	78.23	26.66	16.35	12.23	21.23	**8.85**
Average	91.19	22.67	16.80	10.11143	11.81	9.25

Table 11. Average results by algorithms w.r.t. all performance parameters

Performance Parameter	**THR-MMT**	**IQR-MMT**	**MAD-MMT**	**LR_MMT**	**LRR_MMT**	***iCloud*MMT**
Energy (kWh)	129.21	107.52	102.64	**87.48**	87.54	89.74
VMMC ($x10^3$)	13.58	12.42	7.65	**4.21**	6.36	6.85
SLATAH	8.06%	2.86%	5.18%	55.53%	55.54%	**1.97%**
PDM	0.07%	0.06%	0.04%	**0.02%**	**0.02%**	0.03%
SLAV ($x10^{-3}$)	3.74%	1.53%	1.41%	8.92%	9.13%	**0.43%**
ESV ($x10^{-3}$)	3.95	2.57	1.41	7.82	9.01	**0.39**
HST (10^3)	95.45	23.13	14.51	8.64	7.96	**7.55**

5. CONCLUSION

A detailed evaluation of the study is done on the deployment and consolidation of VMs using server virtualization in cloud data centers. To create VM placement and consolidation approaches, researchers have taken into account a number of performance metrics, including cost, energy use, and SLA violation. It is discovered that various performance criteria trade-off with one another. This work focuses on the design and implementation of a unique strategy dubbed iCloud for the placement and consolidation of energy-efficient virtual machines while minimizing SLA violations. With the aid of the CloudSim simulator, a large-scale experimental setup was used to develop and evaluate iCloud. The simulation made use of actual workload traces from 1,000 VMs on the Planet Lab dataset. The experimental findings demonstrate that the proposed iCloud significantly decreases the Host Search Time (HST) since the search area is decreased as a consequence of the classification of hosts in the data center. These algorithms use less energy than others like IQR, MAD, and THR because iCloud maintains idle hosts in a power-saving mode, and algorithms based on LR rules have anticipated lower thresholds. Additionally, it is seen that aiCloud prevents host overloading

owing to VM Acceptance State; as a result, it performs better than other methods in terms of SLA Time per Active Host (SLATAH), a key element in the large decrease of SLA violation and ESV. iCloud. Consequently, it can be a tempting technique for the efficient administration of cloud resources.

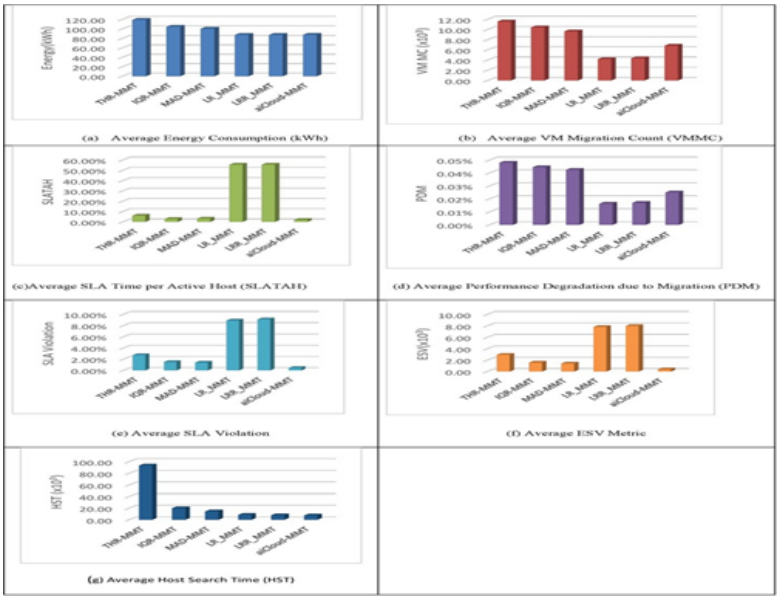

Figure 2. Comparative analysis of other algorithms with aiCloud

The problem of resource fragmentation will be taken into consideration as a future research path for improved resource usage in data centers. The effectiveness of iCloud will be evaluated using additional VM selection criteria and several resource dimensions with both constant and variable upper threshold values. The algorithm could be further modified to reduce energy consumption by shifting lightly loaded servers into an *Offline* state using a lower threshold value.

REFERENCES

1. Abdelsamea, A., El-Moursy, A. A., Hemayed, E. E., & Eldeeb, H. (2017). Virtual machine consolidation enhancement using hybrid regression algorithms. *Egyptian Informatics Journal*, *18*(3), 161–170. doi:10.1016/j.eij.2016.12.002
2. Amazon EC2. (n.d.) Instance Types retrieved from https://aws.amazon.com/ec2/instance-types/
3. Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computer Systems*, *28*(5), 755–768. doi:10.1016/j.future.2011.04.017
4. Beloglazov, A., & Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation*, *24*(13), 1397–1420. doi:10.1002/cpe.1867

5. Beloglazov, A., Buyya, R., Lee, Y. C., & Zomaya, A. (2011). A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers, 82*, 47–111. doi:10.1016/B978-0-12-385512- 1.00003-7

6. Bilal, K., Khan, S. U., & Zomaya, A. Y. (2013, December). Green data center networks: Challenges and opportunities. In *2013 11th International Conference on Frontiers of Information Technology (FIT)* (pp. 229-234). IEEE.

7. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems, 25*(6), 599–616. doi:10.1016/j.future.2008.12.001

8. Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software, Practice & Experience, 41*(1), 23–50. doi:10.1002/spe.995

9. Chaisiri, S., Lee, B. S., & Niyato, D. (2009, December). Optimal virtual machine placement across multiple cloud providers. In *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific* (pp. 103-110). IEEE. doi:10.1109/APSCC.2009.5394134

10. Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., & Doyle, R. P. (2001). Managing energy and server resources in hosting centers. *Operating Systems Review, 35*(5), 103–116. doi:10.1145/502059.502045

11. Dias, D. S., & Costa, L. H. M. (2012, December). Online traffic-aware virtual machine placement in data center networks. In *2012 Global Information Infrastructure and Networking Symposium (GIIS)*, (pp. 1-8). IEEE. doi:10.1109/GIIS.2012.6466665

12. Goudarzi, H., & Pedram, M. (2012, June). Energy-efficient virtual machine replication and placement in a cloud computing system. In *2012 IEEE 5th International Conference on Cloud Computing (CLOUD)* (pp. 750-757). IEEE. doi:10.1109/CLOUD.2012.107

13. Gupta, A., Milojicic, D., & Kalé, L. V. (2012, September). Optimizing VM Placement for HPC in the Cloud. In *Proceedings of the 2012 workshop on Cloud services, federation, and the 8th open cirrus summit* (pp. 1-6). ACM. doi:10.1145/2378975.2378977

14. He, S., Guo, L., & Guo, Y. (2011, July). Real time elastic cloud management for limited resources. In *2011 IEEE International Conference on Cloud Computing (CLOUD)* (pp. 622-629). IEEE. doi:10.1109/CLOUD.2011.47

15. Hyser, C., McKee, B., Gardner, R., & Watson, B. J. (2007). Autonomic virtual machine placement in the data center. *Hewlett Packard Laboratories*.

16. Jadhav, R., & Somani, P. (2015). *U.S. Patent No. 9,176,788*. Washington, DC: U.S. Patent and Trademark Office.

17. Jayasinghe, D., Pu, C., Eilam, T., Steinder, M., Whally, I., & Snible, E. (2011, July). Improving performance and availability of services hosted on iaas clouds with structural constraint-aware virtual machine placement. In *2011 IEEE International Conference on Services Computing (SCC)* (pp. 72-79). IEEE. doi:10.1109/SCC.2011.28

18. Khosravi, A., Garg, S. K., & Buyya, R. (2013, August). Energy and carbon-efficient placement of virtual machines in distributed cloud data centers. In *European Conference on Parallel Processing* (pp. 317-328). Springer. doi:10.1007/978-3-642-40047-6_33

19. Li, B., Li, J., Huai, J., Wo, T., Li, Q., & Zhong, L. (2009, September). Enacloud: An energy-saving application live placement approach for cloud computing environments. In *IEEE International Conference on Cloud Computing CLOUD'09* (pp. 17-24). IEEE.

20. Li, X., Qian, Z., Lu, S., & Wu, J. (2013). Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center. *Mathematical and Computer Modelling*, *58*(5-6), 1222–1235. doi:10.1016/j.mcm.2013.02.003

21. Liu, C. Y., Shie, M. R., Lee, Y. F., Lin, Y. C., & Lai, K. C. (2014, May). Vertical/horizontal resource scaling mechanism for federated clouds. In *2014 International Conference on Information Science and Applications (ICISA)* (pp. 1-4). IEEE.

22. Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing—The business perspective. *Decision Support Systems*, *51*(1), 176–189. doi:10.1016/j.dss.2010.12.006

23. Mastroianni, C., Meo, M., & Papuzzo, G. (2013). Probabilistic consolidation of virtual machines in self-organizing cloud data centers. *IEEE Transactions on Cloud Computing*, *1*(2), 215–228. doi:10.1109/TCC.2013.17

24. Meng, X., Pappas, V., & Zhang, L. (2010, March). Improving the scalability of data center networks with traffic-aware virtual machine placement. In Proceedings IEEE INFOCOM 2010 (pp. 1-9). IEEE. doi:10.1109/INFCOM.2010.5461930

25. Mills, M. P. (2013). The cloud begins with coal: Big data, big networks, big infrastructure, and big power. *Digital Power Group*. Retrieved from http://eduscol.education.fr/sti/sites/eduscol.education.fr.sti/files/resources/techniques/1751/1751-cloud-begins-with-coal.pdf

26. Nguyen Van, H., Dang Tran, F., & Menaud, J. M. (2009, May). Autonomic virtual resource management for service hosting platforms. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing* (pp. 1-8). IEEE Computer Society. doi:10.1109/CLOUD.2009.5071526

27. PlanetLab. (n.d.). A global research network. Retrieved from *https://*www.planet-lab.org/

28. Roytman, A., Kansal, A., Govindan, S., Liu, J., & Nath, S. (2013). Algorithm design for performance aware VM consolidation. *Microsoft Research*.

29. Sharma, O., & Saini, H. (2017). SLA and Performance Efficient Heuristics for Virtual

Machines Placement in Cloud Data Centers. *International Journal of Grid and High Performance Computing, 9*(3), 17–33. doi:10.4018/ IJGHPC.2017070102

30. Shelar, M., Sane, S., Kharat, V., & Jadhav, R. (2014). Efficient Virtual Machine Placement with Energy Saving in Cloud Data Center. *International Journal of Cloud-Computing and Super-Computing*, *1*(1), 15–26.

31. Shelar, M., Sane, S., Kharat, V., & Jadhav, R. (2017, April). Autonomic and energy-aware resource allocation for efficient management of cloud data centre. In 2017 Innovations in Power and Advanced Computing Technologies (i-PACT) (pp. 1-8). IEEE. doi:10.1109/IPACT.2017.8244944

32. Shen, Z., Subbiah, S., Gu, X., & Wilkes, J. (2011, October). Cloudscale: elastic resource scaling for multi- tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing* (p. 5). ACM. doi:10.1145/2038916.2038921

33. SPECpower benchmark. (n.d.). Retrieved from https://www.spec.org/power_ssj2008/results/

34. Teyeb, H., Balma, A., Alouane, N. B. H., & Tata, S. (2014, June). Optimal virtual machine placement in large-scale cloud systems. In *2014 IEEE 7th International Conference on Cloud Computing (CLOUD)*, (pp. 424-431). IEEE.

35. Vu, H. T., & Hwang, S. (2014). A traffic and power-aware algorithm for virtual machine placement in cloud data center. *International Journal of Grid and Distributed Computing, 7*(1), 350–355. doi:10.14257/ijgdc.2014.7.1.03

36. Wang, S., Zhou, A., Hsu, C. H., Xiao, X., & Yang, F. (2016). Provision of data-intensive services through energy- and QoS-aware virtual machine placement in national cloud data centers. *IEEE Transactions on Emerging Topics in Computing*, *4*(2), 290–300. doi:10.1109/TETC.2015.2508383

37. Wang, W., Chen, H., & Chen, X. (2012, September). An availability-aware virtual machine placement approach for dynamic scaling of cloud applications. In *2012 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC)* (pp. 509-516). IEEE. doi:10.1109/UIC-ATC.2012.31

38. Zhuang, Z., & Guo, C. (2013, December). Ocpa: An algorithm for fast and effective virtual machine placement and assignment in large scale cloud environments. In *2013 International Conference on Cloud Computing and Big Data (CloudCom-Asia)* (pp. 254-259). IEEE.