**International Journal of Intelligent Information Technologies**
Volume 15 • Issue 1 • January-March 2019

# Energy Efficient Load Balancing in Cloud Data Center Using Clustering Technique

N. Thilagavathi, Anna University, Chennai, India

D. Divya Dharani, Anna University, Chennai, India

R. Sasilekha, Anna University, Chennai, India
Vasundhara Suruliandi, Anna University, Chennai, India

V. Rhymend Uthariaraj, Anna University, Chennai, India

## ABSTRACT

Cloud computing has seen tremendous growth in recent days. As a result of this, there has been a great increase in the growth of data centers all over the world. These data centers consume a lot of energy, resulting in high operating costs. The imbalance in load distribution among the servers in the data center results in increased energy consumption. Server consolidation can be handled by migrating all virtual machines in those underutilized servers. Migration causes performance degradation of the job, based on the migration time and number of migrations. Considering these aspects, the proposed clustering agent-based model improves energy saving by efficient allocation of the VMs to the hosting servers, which reduces the response time for initial allocation. Middle VM migration (MVM) strategy for server consolidation minimizes the number of VM migrations. Further, randomization of extra resource requirement done to cater to real-time scenarios needs more resource requirements than the initial requirement. Simulation results show that the proposed approach reduces the number of migrations and response time for user request and improves energy saving in the cloud environment.

### KEYWORDS

Agent-Based Model, Cloud Computing, Clustering, Energy Consumption, Energy Efficiency, Load Balancing, Migration, Server Consolidation

## 1. INTRODUCTION

Cloud computing is a paradigm which provides online services and on-demand provisioning of scalable internet services. There are three types of cloud computing services: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Cloud computing has three deployment models: public cloud, private cloud, and hybrid cloud. Cloud computing refers to sharing computing resources to offer different services through the internet. Cloud computing services are offered by various cloud service providers globally, by deploying a data center. These cloud data centers are generally built on heterogeneous servers enabled with virtualization technologies, which provide the ability to handle varying workloads with limited resources (cloud computing infrastructure) to offer reliable services. Such cloud computing infrastructures in data centers are networked systems and data storage used for storing and managing data. Since the IT infrastructure depends on the stability of the business, there is a need to establish and maintain efficient data centers which can provide availability, manageability and scalability.

DOI: 10.4018/IJIIT.2019010104

Since high performance cloud infrastructure consumes a large amount of energy, and dissipates heat and carbon emission to the surrounding, green cloud initiatives contemplate efficient usage of the cloud computing infrastructure and minimization of energy consumption. The survey papers discuss (Nasrin & Mohamed, 2016; Thilagavathi & Uthariaraj, 2016) various issues and factors affecting energy and thermal efficiency of the computing infrastructure. The major part of power usage in a data centre arises from the computing process, disk access, networking and cooling requirements. Increasing energy consumption in data centers not only contributes to high operational cost but also has a severe environmental impact. Hence, energy efficient techniques have become essential due to their economic and environmental benefits.

Cloud service provider manages workloads by implementing mechanisms which will provision efficient usage of computing resources on demand. Thus, efficient resource management is a very essential component for offering cloud-based solutions, which improve reliability and reduce the operating cost of the cloud service provider.

The requests arriving from the users to the data center are distributed among the servers. There are chances that the load may be unevenly distributed. This uneven distribution of load can cause underutilization of resources. Load balancing is done to distribute the load evenly among the processing nodes to enhance the overall performance of the system. This fair allocation of computing resources helps to achieve better resource utilization.

Agent based systems are widely used software paradigms that bring concepts from the theories of artificial intelligence into the mainstream of distributed systems. This system essentially models an application as a collection of components called agents that are characterized by features such as, interaction among themselves, and with other objects, autonomy, proactivity and the ability to communicate. Being autonomous, they can independently carry out complex tasks. Being proactive, they can take the initiative to perform the given task even without an explicit request from the user.

The difference between traditional distributed systems and current agent based systems is that, the computing entities are intelligent (Singh, 1994). However, these multi-agent systems are distributed computing systems interacting with each other. In addition to this, it is simple to consider heterogeneity in the behavioral and random changes in the workload pattern. *Agent-based modeling* is an efficient simulation technique that has great influence in real-world problems in recent years. The behaviors and interactions of the agents represent complex applications more (Helbing &Balietti, 2012) precisely. This makes the modeling approach much more flexible to address critical aspects of real life scenarios.

In this context, the proposed work uses an agent-based model to handle the following issues in cloud data centers to enhance the energy efficiency aspect:

1. Address the open problem (Gutierrez & Ramirez, 2015) of initial allocation of Virtual Machines (VMs) to hosts using the clustering technique, to avoid unnecessary VM migrations. This helps in reducing the response time of the user request.
2. Consider load balancing to distribute the load and lodge extra resource requirement, as there are fluctuations in user requirements.
3. Server consolidation with VM migration. This reduces the energy consumption by turning off the servers that are underutilized.

The functionality of the proposed method based on the above aspects is verified in a JADE development environment. The performance is analyzed by simulation study.

The sections in the paper are organized as follows. Section 2 includes a study of the related literature in the area of the proposed work. The agent based architecture for cloud data centers is outlined in section 3. Section 4 elucidates the simulation setup and analysis. Section 5 discusses the performance evaluation of the proposed work. Section 6 summarizes the work performed and future research direction.

## 2. RELATED WORK

Energy efficiency in large scale data centers has attracted considerable attention of researchers for the past two decades. Various research methodologies have been carried out at the architectural, data center, cluster, system and processor levels to save energy in a cloud environment.

Heuristics based dynamic VM reallocation approaches (Beloglazov& Buyya, 2010; Beloglazov Abawajy, & Buyya, 2012) have been used for energy efficient cloud computing. The modified best fit decreasing algorithm maps VMs to suitable cloud resources; in addition, dynamic consolidation of VM has been done to reduce energy consumption. Performance of their algorithm was evaluated using the CloudSim toolkit. This work performs better when compared to static resource allocation techniques but there is no strong thrust on open challenges identified to enhance the energy-efficient management of cloud computing environments.

A typical low-end server system consumes less than half its actual peak power when it is idle. Variation in load and statistical effects are the main dynamic sources of inefficiency in power deployment. Depending on their activity, high-end server systems consume variable levels of power. Fan et al. (2007) analyzed the power usage pattern of thousands of servers, and how it gets affected by workloads at a large-scale data center level.

Survey and analysis of different load balancing algorithms describe efficient load balancing and task scheduling (K Al Nuaimi, Mohamed, M Al Nuaimi, & AI-Jaroodi, 2012) in a cloud computing environment. Comparison of various algorithms, their advantages and disadvantages are discussed to get an overview of the latest trends in the field. An evaluation of three distributed load balancing algorithms like nature inspired Honeybee Foraging Behavior, Biased Random Sampling and Active Clustering algorithms for cloud environments has been carried out by Martin et al. (2010). The above mentioned survey papers throw light on centralized and distributed load balancing approaches for cloud computing

To minimize the energy cost Luo et al. (2015) proposed an architecture which reflects both the geographic and temporal load balancing for a distributed internet data center. Spatio-temporal load balancing shows better performance and low computational complexity than spatial load balancing and temporal load balancing.

For a large data center Nahir et al. (2016) used a replication-based load balancing technique to reduce performance degradation. This approach involves duplicating the tasks and sending replicas to different servers which will reduce the queuing time, even with a small number of replicas and in particular in high loads. By using this method high communication overhead is reduced but the system is affected by Signal propagation delay.

A novel VM-assign load balance algorithm has been proposed (Damanal&Reddy, 2014) by which all available virtual machines are allocated in an efficient manner to improve resource utilization. Further, the performance is analyzed using the CloudSim simulator.

A heuristic clustering based task deployment for load balancing for cloud data center using the Bayes theorem (Ding, Hu, & Xu, 2016) framed a task deployment approach in which the Bayes theorem and clustering are combined to obtain an optimal set of physical hosts. This approach reduces unnecessary computation complexity and increases deployment efficiency while fulfilling the performance. But this proposed approach applies only to LAN but not to WAN. The system was evaluated using CloudSim and the improvement in throughput and performance was proved.

Inefficiency in energy in data centers is due to server underutilization. The application level consolidation of workloads for cloud data centers is formulated as a modified bin packing problem by Srikantaiah et al. (2008). This study reveals the relationship between energy consumption, resource utilization and performance. Hence, energy aware consolidation improves the sever utilization. Further, energy aware load balancing in clustered cloud models has been studied (Paya & Marinescu, 2014) to improve server utilization and energy optimization. A recent study reveals the features, challenges and practices of energy aware cloud computing models, macro and micro level energy management

strategies, energy efficient assets, and cooling and power distributing units (Vafamehr & Khodayar, 2018).

A chapter in a research handbook (Iyer, Jyosthna,& Jonnalagadda, 2016) discusses various existing techniques to reduce the operating cost and carbon foot print with the cosideration of QOS. Challenges and limitations in cloud energy mangement are discussed to give an overview of the field of study.

An agent based (Gutierrez & Ramirez, 2015) model is proposed for distributed load management in cloud data centers. Heuristics for VM migration and host selection for load balancing were provided. The Collaborative load management Protocol (cProtocol) used for load balancing is based on an agent based architecture. Mechanisms for energy aware server consolidation were proposed with the energy-aware server consolidation protocol (eProtocol). The work considers only appropriate migration heuristics without considering proper initial allocation, which will increase unnecessary migrations and this work holds good only for data centers belonging to the same organization.

Studying various algorithms and frameworks for load balancing, server consolidation and energy management for cloud computing, provides an overview of the existing trends and open challenges in load balancing, server consolidation and shows possible areas of enhancement and exploration to overcome the problems.

## 3. PROPOSED WORK

The proposed work uses mechanisms such as initial allocation of VM, load balancing and server consolidation to handle the issue of energy efficiency. This approach uses an agent based framework to handle these mechanisms. In this proposal, the initial allocation of VMs, an issue highlighted by Gutierrez and Ramirez (2015) has been addressed to reduce the number of migrations. Load balancing handles overloading and extra resource requirement. Server consolidation with migration used to switch off the underutilized servers by periodically monitoring their status.

### 3.1. Proposed Model

This section describes the model used for an agent based framework. This consists of various agents like, User Agent (UA), Front-end Agent (FA), Clustering Agent (CA), Server Manager Agent (SMA) and Virtual Machine Agent (VMA). This work uses the model used by Gutierrez & Ramirez (2015) for distributed load management. Figure 1 represents the agent based architecture for a cloud data centre. It shows, the arrival of a request from UA, the information flow between FA and CA, between FA and SMA and between SMA and VMA.

The functioning of the agents used in the proposed model is discussed here. UA receives the requests from the users. It forwards the request to FA. Further, FA makes communication between UA and other agents. Each server in the data centre has a SMA and virtual machines. SMA manages the messages passed between FA and VMA. Each VM has a VMA. VMA replies to SMA. The user request for VM allocation consists of two components, the number of virtual cores and the memory capacity.

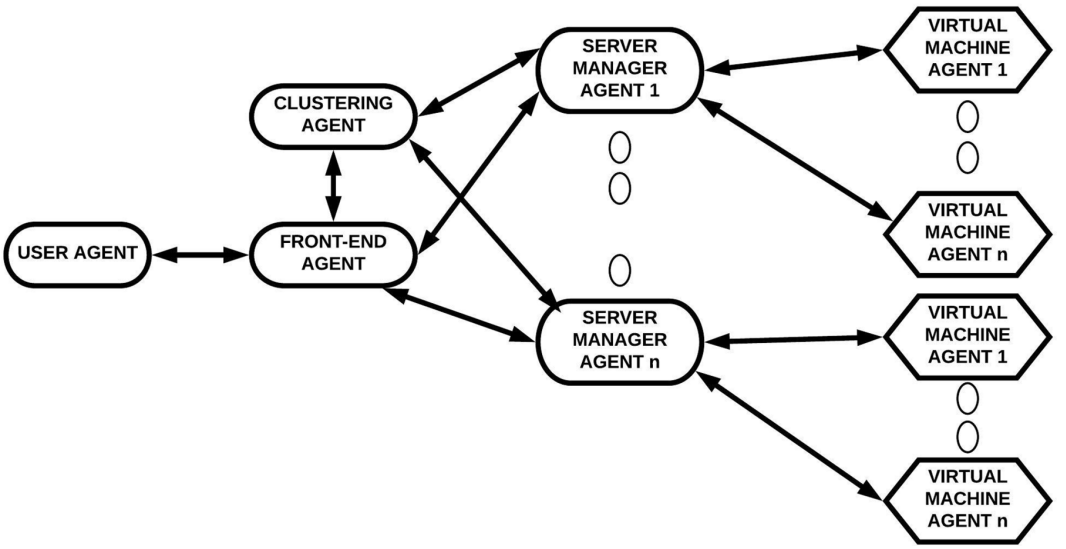
CA, the novel agent used in this work, uses the k-means clustering algorithm (Kanungo et al., 2002) for the grouping of virtual machines. Clustering is done periodically because virtual machines become free after completing their jobs and so these virtual machines should also be included in the clusters for further allocation which ensures that the present cluster does not contain the virtual machines which are allocated in the previous clustering.

Based on this architecture of the proposed work, the three mechanisms, namely initial allocation of VMs, load balancing, server consolidation with migration are presented here.

#### *3.2. Initial Allocation of VM*

- If the requested CPU capacity is less than or equal to the available CPU capacity.

**Figure 1. Agent based architecture for cloud data center**




- If the requested memory capacity is less than or equal to the available memory capacity.

If these two conditions are satisfied, the VMA replies to SMA with an OK message. Otherwise, it sends a NO message. If the VMA sends OK, then the SMA checks if allocating the requested VM would cause the threshold level of the host to be exceeded. If it would exceed, the SMA sends a REJECT message to FA. Otherwise, it sends a REQUEST_OK message to FA to indicate that it is capable of handling the job and also, the threshold will not be exceeded upon allocating the requested VM. If the VMA sends NO, then the SMA sends a REJECT message to FA. The FA ignores the REJECT messages from the SMAs and it generates a list of VMs whose hosts SMA has responded with a REQUEST_OK message. Now, it has to choose one VM which has the least capacity among all the VMs. It has been done by the following steps:

• Arrange the VMs in increasing order of CPU capacity and assign weights (CPU weight) starting from the first VM in the order.

- Calculate the memory weight in the same way as above for each VM.
- Calculate the total weight.

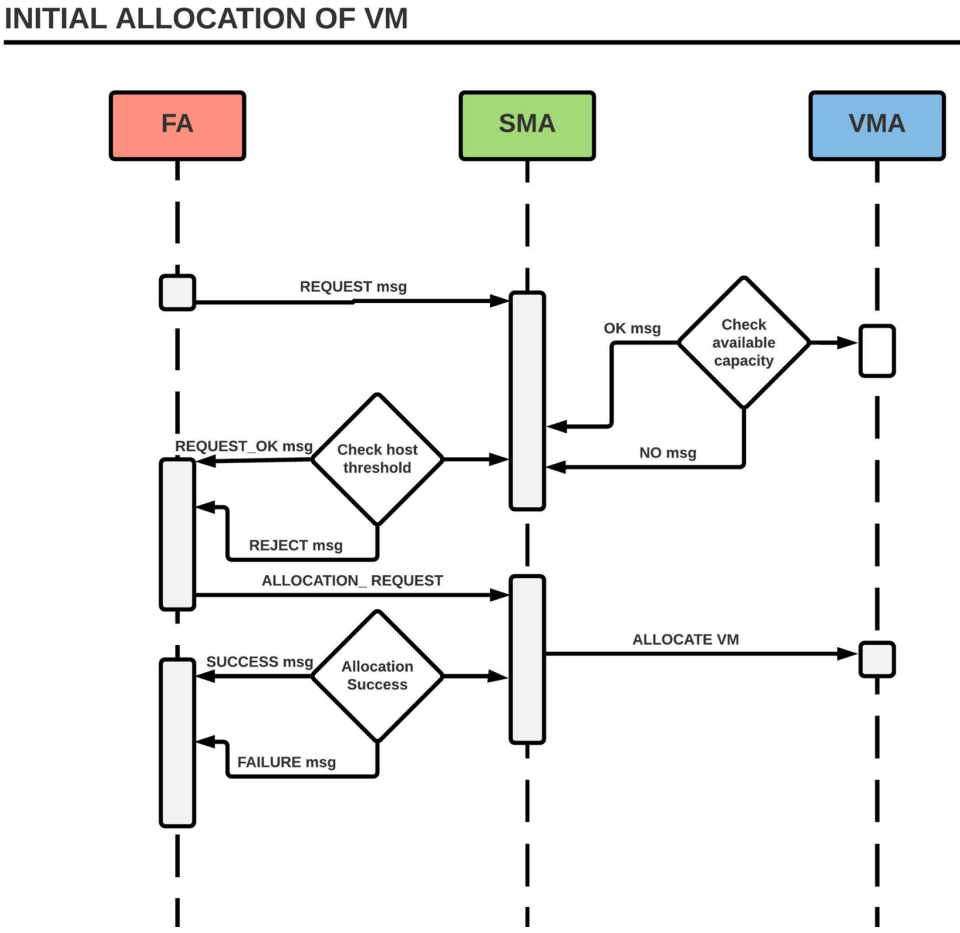
Total Weight = CPU weight + Memory Weight.

- Select the VM with the least weight.

Then, the FA sends an ALLOCATION_REQUEST to the SMA of the host of the chosen VM. If the allocation is done successfully, the SMA sends a SUCCESS message to FA, otherwise it sends a FAILURE message. Repeat all the steps to further initiate the clustering process, since some of the virtual machines are free after completion of their tasks. The above stated steps characterise the novel approach for the initial allocation process of VMs in the selected cluster. Figure 2 shows the sequence diagram of the initial allocation process.

CA uses the k-means clustering algorithm (Kanungo et al., 2002) for the grouping of virtual machines. Clustering is done periodically because virtual machines become free after completing their

**Figure 2. Sequence diagram for initial allocation Of VM**




jobs and so these virtual machines should also be included in the clusters for further classification. A brief of the clustering technique, functions at CA and functions at FA are illustrated below.

### *3.2.1. K-means Clustering Technique*

The K-means clustering technique intends to partition 'n' objects into 'k' groups, where k is predefined. Then randomly assign a k value for the initial cluster formation. Allocate objects to their nearest cluster center corresponding to the Euclidean distance function. The Cluster centroid is an average of all objects in each cluster. K-means clustering is not only efficient and faster than hierarchical clustering, but also produces tighter clusters. The Cluster centroid for each cluster is calculated using equation (1).

Centroid of cluster I,

$$C_{i=} (\sum x_j/n, \sum y_j/n) \quad (1)$$

Where j = 1, 2, 3…n

$x_j$ = j th x point in the cluster

$y_j$ = j th y point in the cluster
n = no. of points in the cluster

While applying this concept in this proposal, 'n' is considered as the number of servers, and 'k' is the number of clusters; here k is predefined and the VMs in the hosting servers are considered as point 'p'.

### *3.2.2. Functions of the Clustering Agent*

After waiting for time 't', the clusters are formed and the centroid of each cluster is calculated by the clustering agent.

Step 1. Pick 'k' points randomly and assign them as initial cluster centroids.
Step 2. Do for each point 'p':
  a. Calculate Euclidean distances from 'p' to all the cluster centroids.
  b. Put the point 'p' in a cluster 'c' such that p is closer to the centroid of the cluster 'c' than any other cluster centroids.
Step 3. Calculate a new cluster centroid using equation (1).
Step 4. Repeat steps until no more different clusters can be formed.

3.2.3. Functions of the Frontend Agent

Step 1. Choose a cluster where VMs with similar resource capacity are required using similarity measures.
Step 2. Send a REQUEST message to all the hosts in the cluster requesting for the corresponding VMs.
Step 3. Get the replies from SMAs.
Step 4. If the reply from an SMA is REJECT, Then Ignore
Step 5. Else put the corresponding VM in a list
Step 6. Sort the VMs in the list in the ascending order of CPU capacity.
Step 7. Assign CPU weight (1, 2, 3…) for each VM.
Step 8. CPU $weight_i = I$
  a. Where 'i' is the position of VM in the sorted list of VMs arranged in the ascending order of CPU capacity.
Step 9. Sort the VMs in the ascending order of memory capacity.
Step 10. Assign memory weight (1, 2, 3…) for each VM.
Step 11. Memory $weight_i = I$
  a. where 'i' is the position of VM in the sorted list of VMs arranged in the ascending order of memory capacity.
Step 12. Compute the total weight.
  a. Total Weight = CPU weight + Memory weight
  b. Select the VM with the least Total Weight.
  c. Send ALLOCATION_REQUEST to the selected VM's host SMA specifying the VM for allocation.
Step 13. Block until SMA replies.
Step 14. If the SMA replies SUCCESS, then terminate the thread.
Step 15. Else if SMA replies FAILURE then go back and repeat all the steps from the clustering process.

By performing the above mentioned functions of CA and FA, the initial allocation of VM is done efficiently. This reduces the number of request messages sent to the hosts in the selected cluster instead of all the hosts in the data center. This initial allocation reduces unnecessary migration, since the overloading condition is checked before VM allocation. Load balancing is used wherever necessary. The following section discusses the load balancing process used in this proposal.

### 3.3. Load Balancing

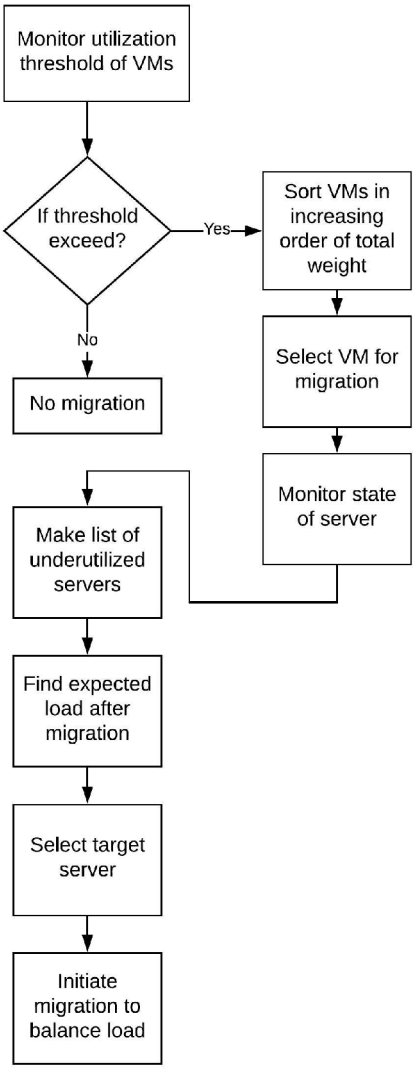
Load balancing is performed when a server is overloaded. A server may be overloaded due to two reasons. One, the VM in that server may require extra resource than what is allocated to it. The other, is the imbalance in load distribution. The CPU and memory requirements of a job are generally predicted using workload prediction methods (Vazquez, Krishnan, & John, 2015). This prediction is done before the initial allocation of a job to a VM. The workload prediction methods predict the requirements of the job based on previous workload patterns. Sometimes, the actual job may require more resource than the predicted requirement. When the resource usage of a job exceeds the allocated resource capacity of the VM, there is a need for migration. This work considers such real time scenarios involving more resource requirement than initial requirement specification, when randomization of extra resource requirement is done in the simulation. This randomized request is made only when the details of all the VMs are gathered by their respective SMAs.

The threshold management for load balancing can be done using the VM usage profile. SMA has to manage the VM resource usage information (utilization). While setting the threshold, there is a need to check that the threshold is neither too low nor too high. Setting up the threshold too low may cause or will lead to more number of migrations. More number of migrations will result in SLA violations. whereas, fixing the threshold too high will result in system overutilization, which will lead to performance degradation. So it is essential to set an optimal threshold level to trigger migration. In the proposed system, the threshold level is set to 75% for load management after performing 10 iterations, whereas the existing model uses a lesser threshold (Gutierrez & Ramirez, 2015), which leads to more number of migrations. An Efficient threshold level is set to the servers and constantly monitored to initiate migration. Monitoring the threshold periodically is essential to identify when to perform migration. This frequent monitoring of the threshold is handled by the SMA. The SMAs gather the load of each VM regularly. Based on the load of the VMs, the SMAs check the CPU and memory thresholds. When the upper threshold is exceeded, the SMA triggers the migration.

Migration may be triggered in two different situations. The first one is, when the upper threshold is exceeded migration is triggered, i.e. server is overloaded. The other one is, when the lower threshold is exceeded; i.e. the server is underutilized. To consider the first case, the VMs in that server are assigned weights based on the CPU and memory usages. Then the combined weights are sorted. If the VM with the lowest weight is migrated, the load on that server may exceed which may result in VM migration. And if the VM with the highest weight is migrated, the load on the server to which the VM is migrated may require further migration. Therefore, the VM that has neither a lower weight nor the highest weight is chosen for migration; i.e. the VM that is in the middle (MVM method) of the sorted array is chosen for migration. Figure 3 shows the flow diagram for load balancing.

When a VM is selected for migration by the SMA, the server machine (host) to which the VM has to be migrated must be identified. For this, the concerned SMA communicates with the FA to trigger migration by providing the details of the selected VM. The FA collects the details of all the servers and finds the best server to migrate the VM. The cluster which is nearest (k-means clustering) to the selected VM is considered for searching the target server. This is done by comparing the difference between the VM requirements of CPU and memory capacities and each of the centroid values of clusters (expressed as CPU and memory capacities). This method of searching for the target server in only one cluster reduces the searching time and thus the waiting time for migration is reduced. When a cluster is selected, the free VMs in that cluster are identified. From these available VMs, the VMs that can satisfy the selected VM's CPU and memory requirements are chosen. Then, for each chosen VM, the corresponding server (host) to which the VM belongs is found. For each chosen server, the expected load when the selected VM is transferred to that server machine is calculated. The probability of migration is less when the expected load on that server machine is less. Therefore, the server that has the lowest expected load is selected for migration. This information is sent to the SMA which initiates the migration by FA. Figure 3 shows the flow diagram for load balancing.

**Figure 3. Flow diagram for load balancing**




### *3.3.1. VM Selection Process*

Step 1. Arrange the VMs of the overloaded server in the increasing order of CPU usage.
Step 2. Assign CPU weights 1, 2, 3...to the VMs in the arranged order respectively.
Step 3. Arrange the VMs in the increasing order of memory usage.
Step 4. Assign memory weights 1, 2, 3... to the VMs in the newly arranged order respectively.
Step 5. Compute
a. $\text{Total Weight}_i = \text{CPU Weight}_i + \text{Memory Weight}_i$.
Step 6. Sort the VMs in the increasing order of total weight.
Step 7. Pick the Middle VM (MVM method) for migration.

### *3.3.2. Target Server Selection Process*

Implementing the target server selection algorithm involves contribution from various agents for selecting the proper target server. The steps are as follows.

*Functions at SMA:* Send message (selected VM's ID, server's ID) to FA

*Functions at FA:* Send message (VM request (CPU capacity, Memory capacity)) to CA
*Functions at CA:* For all the clusters
Step 1. Dis[i] ← Euclidean distance between the cluster centroid and VM request.
Step 2. Find 'i' such that dis[i] is the smallest value
Step 3. Selected _Cluster ← Cluster 'i'
Step 4. Send Selected_Cluster to FA

#### *3.3.3. Functions at FA:*

For each VM 'i' in Selected_Cluster
Step 1. If Status(i) = FREE and $CPU\_Capacity_i$>= required CPU capacity and $Mem\_Capacity_i$>=required Mem capacity
Step 2. If Status($Server_i$)! = OVER_UTILIZED
Step 3. VM_array ← $VM_i$
Step 4. Server_array ← $Server_i$
Step 5. For each server 'j' in Server_array
Step 6. $Expected\_CPU\_load_j$← $CPU_j$ + required CPU
Step 7. $Expected\_Mem\_load_j$← $Mem_j$ + required Mem
Step 8. Find 'j' with lowest $Expected\_CPU\_load_j$ and $Expected\_Mem\_load_j$ values
Step 9. Selected_Server ← Server 'j'
Step 10. Send message (Selected_Server's ID, migrating VM's ID) to the SMA which initiates the migration process.

Once the VM to be migrated and the server machine to which the VM has to be migrated are selected, the selected server machine is monitored to check if it still can handle the requested job and allocate the corresponding VM at that instant of time. This is done to avoid situations where the selected server might be engaged with some other job in the free VM slot at that time, making the migration impossible to happen. Then, the FA sends a message to the selected server's SMA regarding migration so that the server can make suitable arrangements for VM allocation. Then the VM is migrated to the selected server with the help of the FA.

### 3.4. Server Consolidation

Server consolidation helps in minimizing the number of active servers in a data centre, which in turn, reduces the energy consumption considerably. The server consolidation is composed of 3 parts, identifying underutilized servers, selecting the leader to perform server consolidation and selecting the host server where the VM is to be migrated. Then migration is performed to turn off the underutilized server. Figure 4 shows the flow diagram for the server consolidation process.

#### *3.4.1. Identification of Underutilized Servers*

The utilization level of the servers is categorized into 3 states – underutilized, normally utilized and over utilized at FA, by checking the status of the server. The FA monitors the state of all the servers with the help of the SMAs of the servers. Whenever there is a change in the state of a server, its SMA sends this change of state information to the FA. This message contains the server ID and the changed status of the server.

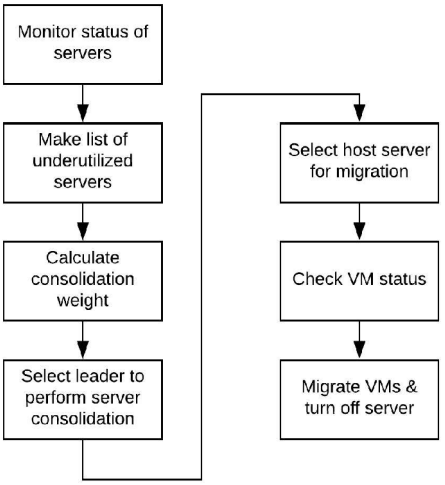
Change of State message: (Server ID, changed state)

The FA updates the states of the servers when it gets the Change of State message from its SMAs. While monitoring the states of the SMAs, the FA looks for the underutilized servers. Then, the FA makes a list of all the underutilized servers.

#### *3.4.2. Leader Selection Process*

At any point of time, the FA has a list of all the underutilized servers. The list often contains more than one underutilized server. In this case, if all the underutilized servers start migration simultaneously,

**Figure 4. Flow diagram for server consolidation**




the underutilized servers may end up migrating their VMs to one another which leads to a migration loop that will never allow the associated servers to be turned off. To avoid the above mentioned case, only one server must initiate the migration for performing server consolidation (Gutierrez & Ramirez, 2015). So, a leader is selected to initiate the migration. The Consolidation weight is calculated for each of the underutilized servers. The consolidation weight is given as follows.

$$\text{Consolidation weight } W_i = 1/ (C_i * \text{Active VM}_i) \quad (2)$$

where Active $VM_i$ is the number of active VMs in that server.

The underutilized server with the largest consolidation weight is selected as the leader. Then the leader's VMs are migrated to other servers. This method uses the fact that if two or more servers have the same CPU utilization percentage, then the server with the minimum number of active VMs will be chosen as the leader as this reduces the number of migrations as well as migration overhead and energy.

### *3.4.3. Host Selection and Initiate Migration Process*

Once the leader for triggering server consolidation is selected by the FA, it sends a message to the selected leader's SMA requesting to trigger server consolidation. When the leader server's SMA gets the message, it starts to migrate all the VMs of the server and to turn it off. For migration to happen, the host servers must be selected. For selecting those servers, one VM is chosen at a time and a potential server is searched for it. This is repeated for all the VMs present in the leader server. The Selection of host servers for migrating the VMs of the selected leader for performing server consolidation is just the same as selecting the host server for migrating the VM. For each VM, after selecting the best host server for migration, the selected server's status is checked to ensure that it is still capable to undertake the migration. When all the necessary checks (server's load, and selected VM slot's status) for successful migration are made, the VMs are migrated to the corresponding host servers. After migration of all the VMs of the leader server, the concerned SMA turns the server off, to save energy consumption.

### 3.5. Simulation Setup

In the proposed work, we are dealing with three key aspects, namely, initial allocation of VMs, load balancing and server consolidation with migration. A simulation study has been carried out to validate the work. The experimental setup for agent based simulation was performed using the Java Agent DEvelopment (JADE) framework (Telecom Italia, 2000; Bellifemine, Caire, Poggi, & Rimassa, 2008; Caire, 2009). The Interface for each agent was created and communication was established among all the agents. The Behavior of each agent was specified in the JADE simulation environment. The simulation contains two groups of servers. Group 1 contains 6 servers and each of these servers contains 4 VMs. Group 2 contains 6 servers and each of these servers contains 8 VMs. Each server contains a SMA which makes a total of 12 SMAs. Group 1 contains 24 VMs and group 2 contains 48 VMs which makes a total of 72 VMAs. The JADE experimental setup also contains many UAs corresponding to the number of requests simulated, one CA and one FA. The number of VMAs depends on the number of virtual machines contained in each group of servers. The performance was measured with the response time for the allocation of VMs to user requests, the number of VM migrations and the amount of energy consumed. Response time denotes the time required for handling the requests and the number of migrations denotes the VM migrations as a result of load balancing and server consolidation.

Table 1 Describes the various agents used in the proposed agent based architecture for the data center. The system contains 50 UAs and one CA and one FA. The number of SMAs depends on the number of servers used in the system. There are 12 SMAs and 72 VMAs used for the simulation. The number of VMAs depends on the number of VMs contained in each group of servers. The total capacity of each server is determined by the number of virtual cores and the memory capacity.

### 3.6. Discussion

Experiments were performed to compare the interpretation of the proposed methods and baseline methods, namely, CPU based and memory based migration methods. The performance is estimated by the number of VM migrations, the response time for the allocation of VMs to user requests, and the amount of energy consumed. Response time denotes the time required for handling the requests and the number of migrations denotes the VM migrations as a result of load balancing and server consolidation.

The above Figure 5 represents the number of migrations that has been performed as a result of CPU based selection of VM, memory based selection of VM, combined CPU and memory based method from the existing work and the proposed Middle VM Method (MVM) for the selection of VMs for migration. The CPU based method results in more number of migrations when compared to other given methods. From the above graph it can be inferred that the proposed method results in less number of migrations when compared to the existing methods. Even for larger number of user requests, the middle VM method results in less number of migrations.

**Table 1. Agents used in the simulation**

| Agents used | Numbers(#) |
|---|---|
| User Agent | 50 |
| Frontend Agent | 1 |
| Clustering Agent | 1 |
| Server Manager Agent | 12 |
| Virtual Machine Agent | 72 |

**Figure 5. Comparison of number of migrations**

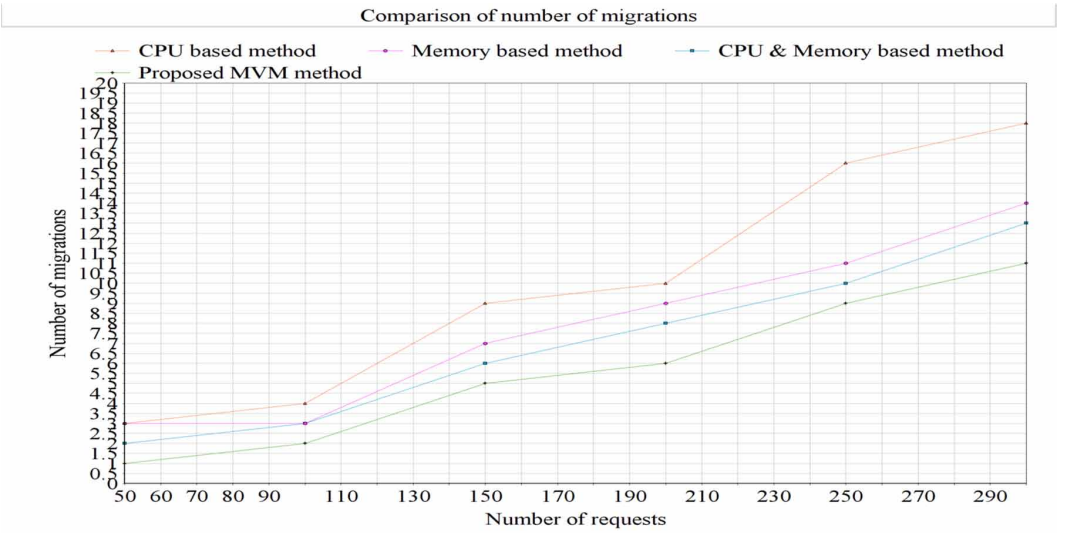



Figure 6 represents the response time with a different number of user requests. Here, the response time is represented in milliseconds. The experiment includes a comparison of the existing system (Gutierrez & Ramirez, 2015) and the proposed system to evaluate the response time with different numbers of requests in each system. The response time of the existing system which does not use the clustering agent is greater than the proposed system which uses its novel clustering agent. Initially for 50 requests, the difference in the response time is minimal (i.e. 10 ms), but when the number of requests increases to 150, 250 and so on, the difference in response time between both the systems increases gradually to 70 ms and 80 ms respectively. Finally, when the user request is 300, the response time for the system without the clustering agent is 205ms and the response time for the proposed system with the clustering agent is 125ms. Therefore, it can be observed that response time with the use of the clustering agent is lesser, and hence, shows better performance.

Figure 7 shows the comparison of the amount of energy saved by using the existing server consolidation method with that of the proposed server consolidation method. The power consumption of a server depends on its CPU utilization and the power drawn during the peak load state and idle state of the server. Power consumption is stated using equation (3) in terms of kilowatts per hour. Power consumption is estimated as follows:

$$P = (P_{max} - P_{idle}) \times \frac{n}{100} + P_{idle} \quad (3)$$

In the given equation (3) P refers to the estimated power consumption, $P_{max}$ denotes the power consumption at maximum performance, $P_{idle}$ is the power consumption of the system in the idle state and n refers to the processor utilization value. The values of the power consumption model were calculated using a quasi-linear relationship between CPU usage and power consumption for one of the Intel servers (Fan et al., 2007; Chen et al., 2008) based on which the proposed system was evaluated.

The experiments were conducted to achieve energy efficiency in a cloud environment. The performance measure is energy saved, measured in kilowatt-hour (kWh) for the cloud data centre. Based on the power model mentioned above, the amount of energy saved while using the existing

**Figure 6. Comparison of response time**

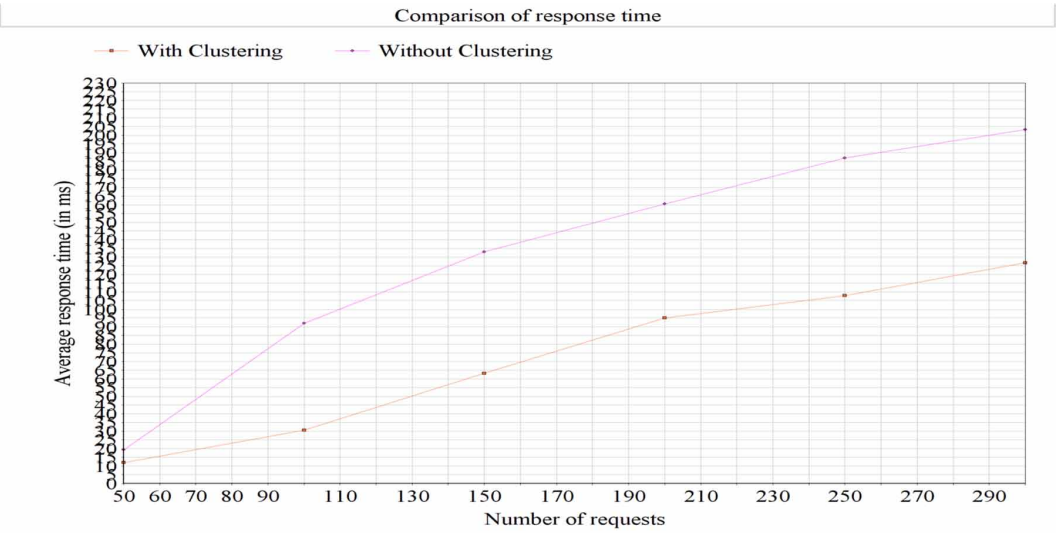
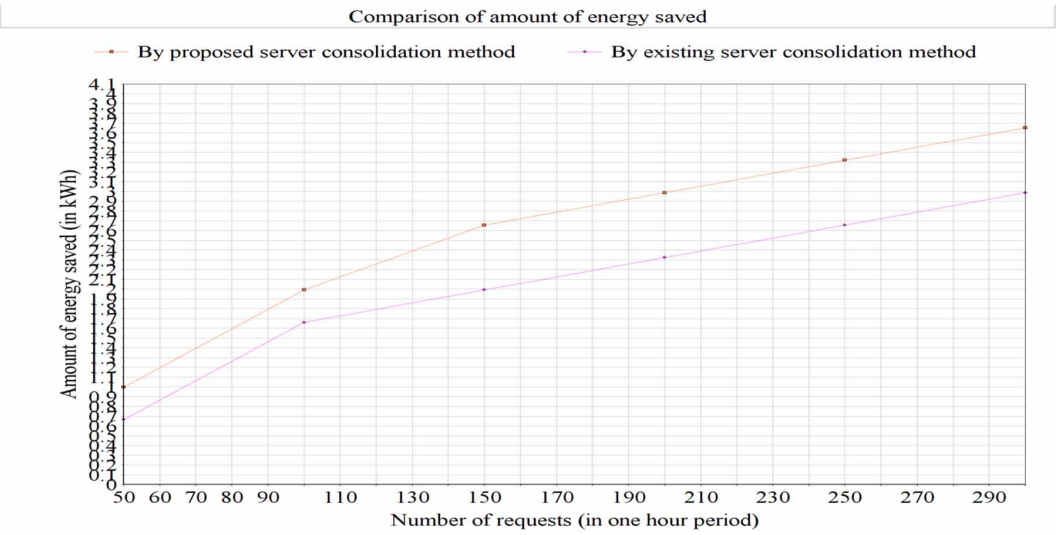



**Figure 7. Comparison of energy saved**




server consolidation and using the proposed server consolidation method for different number of requests in a one-hour period was analysed. From the experiment, it is inferred that the amount of energy saved while using the proposed server consolidation method is high. The proposed work endowed with the leader selection algorithm in server consolidation showed better kWh consumption for the data centre, which was lower than the kWh consumption attained while using the existing server consolidation. It is observed that there is an increase in the amount of energy saved with an increase in the number of requests. Adopting the proposed server consolidation method in a larger data centre will result in minimizing the energy consumption to a greater extent.

## 4. CONCLUSION AND FUTURE WORK

In this paper, the problem of energy efficiency in a cloud data centre is addressed through initial allocation of VMs, energy efficient load balancing and server consolidation mechanisms, by adopting an agent based model. The initial allocation of VMs is achieved through the k-means clustering approach. The novel clustering agent, plays a vital role both in reducing the energy consumption and the response time. The performance is measured with the response time for the allocation of VMs to incoming requests, the number of VM migrations and the amount of energy consumed. Clustering of the hosting servers is done using a clustering agent. It is observed that, efficient VM selection and target server selection processes help to achieve a load balance. Server consolidation is performed to reduce the number of active servers, thereby reducing energy consumption. The proposed method reduces the number of migration due to efferent initial allocation of VMs. It also shows improvements in the response time when the number of user requests increases.

The future directions for further research include dynamic threshold adjustment and live migration heuristics to improve energy efficiency in a cloud based environment. Another opening problem might be the interactions and coordination among their member agents. Also, agent based modelling attracted considerable interest recently; considering intelligent agents or self-organised agents is a promising area to explore.

## 5. ACKNOWLEDGMENT

The author is thankful to the research fellowship (Proc. No. 9500/PD2/2014) provided by the Government of India, Department of Science and Technology, DST PURSE II fellowship, Anna University. The author is indebted to Prof. Dr. S. Jayaprakash for his valuable suggestions. The authors are grateful to the anonymous reviewers for their feedback and critical comments to improve the quality of the paper.

## REFERENCES

Akhter, N., & Othman, M. (2016). Energy aware resource allocation of cloud data center: review and open issues. *Journal of Cluster Computing*, *19*(3), 1163-1182.

Al Nuaimi, K., Mohamed, N., & Al Nuaimi, M., & AI-Jaroodi, J.(2012). A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms. In *IEEE Computer Society, Second Symposium on Network Cloud Computing and Applications*. London: IEEE. doi:10.1109/NCCA.2012.29

Bellifemine, F., Caire, G., Poggi, A., & Rimassa, G. (2008). JADE – A software framework for developing multi-agent applications. Lessons learned. *Journal of Information and Software Technology, Elsevier*, *50*(1-2), 10–21. doi:10.1016/j.infsof.2007.10.008

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).

Beloglazov, A., & Buyya, R. (2010). Energy Efficient Allocation of Virtual Machines in Cloud Data Centers. *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. doi:10.1109/CCGRID.2010.45

Caire, G. (2009). JADE Tutorial. Jade programming for Beginners. *CSELT*. Retrieved from http://jade.tilab.com/doc/tutorials/JADEProgramming-Tutorial-for-beginners.pdf

Chen, G., He, W., Liu, J., Nath, S., Rigas, L., Xiao, L., & Zhao, F. (2008). Energy-aware server provisioning and load dispatching for connection-intensive Internet services. *Proc. 5th USENIX Symp. Netw. Syst. Des. Implementation*, 8, 337–350.

Dabbagh, M., Hamdaoui, B., Guizani, M., & Rayes, A. (2015). Energy-Efficient Resource Allocation and Provisioning Framework for Cloud Data Centers. *IEEE eTransactions on Network and Service Management*, *12*(3), 377–391. doi:10.1109/TNSM.2015.2436408

Damanal, S. G., & Ram Mahana Reddy, G. (2014). Optimal Load Balancing in Cloud Computing By Efficient Utilization of Virtual Machines. In *Proceedings of COMSNETS, Sixth International Conference on Communication Systems and Networks*. Bangalore, India: IEEE. doi:10.1109/COMSNETS.2014.6734930

Ding, Y., Hu, L., & Xu, G. (2016). A Heuristic Clustering-Based Task Deployment Approach for Load Balancing Using Bayes Theorem in Cloud Environment. *IEEE Transactions on Parallel and Distributed Systems*, *27*(2), 305–316. doi:10.1109/TPDS.2015.2402655

Fan, X., Weber, W. D., & Barroso, L. A. (2007). Power Provisioning for a Warehouse-sized Computer. In *Proceedings of the ACM International Symposium on Computer Architecture*. San Diego, CA: ACM.

Gutierrez, J. (2015). Collaborative Agents for Distributed Load Management in Cloud Data Centers Using Live Migration of Virtual Machines. *IEEE Transactions on Services Computing*, *8*(6), 916–929. doi:10.1109/TSC.2015.2491280

Helbing, D., & Balietti, S. (2012). Agent-Based Modeling. In Self-Organization: Agent-based simulations and experiments to study emergent social behaviour (pp. 25-70). Berlin: Springer-Verlag. doi:10.1007/978-3-642-24004-1_2

Italia, T. (2000). *JADE download JADE-all-4.4.0* [Software]. Available from http://jade.tilab.com/download/jade/

Iyer, G. N., Jyosthna, P. M., & Jonnalagadda, S. (2016). Energy management in cloud computing [Abstract]. IJIIT, 14, 293-309.

Kansal, N. J., & Chana, I. (2016). Energy-aware Virtual Machine Migration for Cloud Computing-A Firefly Optimization Approach. *Journal of Grid Computing*, *14*(2), 327–345. doi:10.1007/s10723-016-9364-0

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002). An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 881–892. doi:10.1109/TPAMI.2002.1017616

Luo, J., Rao, L., & Liu, X. (2015). Spatio-Temporal Load Balancing for Energy Cost Optimization in Distributed Internet Data Centers. *IEEE Transactions on Cloud Computing*, *3*(3), 387–397. doi:10.1109/TCC.2015.2415798

Nahir, A., Orda, A., & Raz, D. (2016). Replication Based Load Balancing. *IEEE Transactions on Parallel and Distributed Systems*, *27*(2), 494–507. doi:10.1109/TPDS.2015.2400456

Paya, A., & Marinescu, D. C. (2014). Energy-aware Load Balancing Policies for the Cloud Ecosystem. In *IEEE 28th International Parallel & Distributed Processing Symposium Workshops*, IEEE Computer Society.

Randles, M., Lamb, D., & Taleb-Bendiab, A. (2010). A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing. In *WAINA '10 Proceedings of IEEE 24th International Conference on Advanced Information Networking and Applications Workshops.* Perth: IEEE.

Rao, L., Liu, X., Xie, L., & Liu, W. (2010). Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment. In *Proc. IEEE INFOCOM*. San Diego, CA: IEEE. doi:10.1109/INFCOM.2010.5461933

Singh, M. P. (1994). Multiagent Systems. In G. Goos & J. Hartmanis (Eds.), A theoretical framework for Intentions, Know-How and Communications, LNCS (pp. 1-14). Berlin: Springer-Verlag.

Srikantaiah, S., Kansal, A., & Zhao, F. (2008). Energy Aware Consolidation for Cloud Computing. In *Proceedings of USENIX, Conference on power aware computing and systems, HotPower'08*. San Diego, CA: USENIX Association Berkeley.

Thilagavathi, N., & Uthariaraj, V. R. (2017). A Study on Energy and Thermal Management Factors of Green Computing. In *Proceedings of International conference on big data and cloud computing*. Coimbatore: Karunya University.

Vafamehr, A., & Khodayar, M. E. (2018). Energy-aware cloud computing. *The Electricity Journal*, *31*(2), 40–49. doi:10.1016/j.tej.2018.01.009

Vazquez, C., Krishnan, R., & John, E. (2015). Time Series Forecasting of Cloud Data Center Workloads for Dynamic Resource Provisioning. *Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications*, *6*(3), 87–110.

Xiao, Z., Song, W., & Chen, Q. (2013). Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment. *IEEE Transactions on Parallel and Distributed Systems*, *24*(6), 1107–1117. doi:10.1109/TPDS.2012.283

*Thilagavathi N. is a PhD student at Anna University, MIT Campus, Chennai, India.*

*Divya Dharani D. is a student, MIT Campus, Anna University.*

*R. Sasilekha is a Student, MIT Campus, Anna University.*

*Vasundhara Suruliandi is a Student, Anna University.*

*V. Rhymend Uthariaraj is a Prof. & Director, Ramanujan Computing Centre, Main Campus, Anna University, Chennai, India.*