Journal of Theoretical and Applied Information Technology
31st December 2016. Vol.94. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FCM-BPSO: ENERGY EFFICIENT TASK BASED LOAD BALANCING IN CLOUD COMPUTING

[1] GEETHA MEGHARAJ, [2]DR. K. G. MOHAN
[1] Associate Professor, CSE, Sri Krishna Institute of Technology, Bengaluru
[2] Professor & HOD CSE, Sai Vidya Institute of Technology, Bengaluru
E-mail: [1]geethagvit@yahoo.com, [2]mohan_kabadi_g@yahoo.com

ABSTRACT

Virtual machine (VM) migration is a methodology used for attaining the system load balancing in a cloud environment by transferring the one VM from one physical host to another host. In this paper, we plan to migrate the extra tasks from overloaded VM to suitable VM instead of migrating the entire overloaded VM. In order to select the host VMs, a FCM clustering algorithm has been used to group the similar kind of host VMs. Once the VMs identified as overloaded, then the corresponding candidate VMs are found using the FCM clustering algorithm. Binary Particle Swarm Optimization (BPSO) methodology has been used for selecting the host VMs from the set of candidate VMs based on multi-objective fitness function, which includes task transferring time, task execution time and energy consumption. By allocating the extra task from the overloaded VMs to the proper VMs, we achieved the load balancing in the cloud environment. The implementation of proposed methodology FCM-BPSO has been done using CloudSim tool and comparative analysis done to evaluate the FCM-BPSO method with a traditional load balancing algorithm in terms of energy consumption and time.

Keywords: *Load Balancing Algorithm, Task Scheduling, Particle Swarm Optimization, Fuzzy C Means, Clustering.*

1. INTRODUCTION

In the past years, it was little bit tough for smaller organizations to increase the number of essential computing resources like hardware, software or storage media (e.g. virtual desktop, virtual machines, development tools etc.), to increase their computation power and storage capacity. But nowadays, cloud computing delivers on demand resource allocation plan. These plans are coming for pay per use basis, in the form of services over the internet [1]. Cloud computing is an emerging technique and it is very successful due to its following characteristic like reliability, security, speed, fault tolerance and efficient communication etc. among different networks [2]. Because of the exponential growth of cloud computing, a number of clients and their demands for services are increasing quickly. This results in heavy workload over the servers and computing resources. An efficient load-balancing algorithm require for proper utilization of the resources to tackle workloads of the server. Load balancing is a mechanism for distribution of the workload uniformly across all the participating servers [3], [4]. So, to use the advantages of cloud computing entirely, an effective Load balancing technique is necessary. Several Load Balancing mechanisms had been proposed before. However, there are some challenges like: minimum migration time, minimum downtime, and minimum workload because of Virtual Machine Migration [5], [10]. The cloud computing platform for resource management achieves dynamic balance between the servers using virtualization technology. The online VM migration mechanism [6] can achieve remapping of VMs & physical resources dynamically to accomplish load balancing of the entire system [7]. Specifically, VM migration has been employed to carry out flexible resource allocation or reallocation, through transferring a VM from one physical machine to another for stronger computation power, larger memory, fast communication capability, and energy savings [8]. The major disadvantage of conventional procedure for accomplishment of the system load balancing in a cloud environment is that the majority's endeavor to migrate overloaded VMs [6], [7], [8], [9].

This VM migration strategy has a few downsides: (1) it prepares dirty memory that will increase after pre-copy in online VM migration, (2) it uses a lot of memory in both primary Physical Machine (PM) and new host PM, (3) it needs to pause the primary VM, so triggering VM downtime, (4) it carries the risk of losing recent customer activities in online VM migration, and (5) it is not cost effective and time-consuming [1].

2. LITERATURE REVIEW

Shahrzad Aslanzadeh et al. [11] presented a workflow load-balancing algorithm that could be used in an elastic cloud. STeM algorithm was suggested as the potential algorithm that could improve the load management by explaining the magnitude and direction of the fluctuation between dependent tasks. The model helped to find the level of the dependencies between each task by acknowledging the magnitude and direction of the load. Considering the behavior of the depended tasks, this approach explained a pattern for managing the load balancing more efficiently. The expected benefits of the proposed algorithm improved load balancing technique, which could result in better performance rate.

Dhinesh Babu L. D. and P. Venkata Krishna [12] proposed a load balancing mechanism for cloud computing environments based on behavior of honey bee foraging strategy. They balanced the load and took into consideration the priorities of tasks that had been removed from heavily loaded Virtual Machines. The tasks eliminated from these VMs were treated as honeybees, which were the information updaters globally. Their algorithm also considered the priorities of the tasks. Honeybee behavior motivated load balancing enhanced the total throughput of processing & priority based balancing focused to reduce the amount of time a task had to wait on a queue of the VM. Their methodology reduced the response of time of VMs.

Zhi-Hui Zhan et al. [13] presented a LAGA, which was developed to solve task-scheduling problems in a cloud computing environment. As a better version of existing algorithms, it was suitable for complex problems and was capable of finding more load balancing solutions while maintaining good performance of the Makespan. Through the introduced time load balancing model to modify the fitness function, and the Min-Min, Max-min methods used for population initialization, the algorithm was proved efficient and useful.

Fahimeh Ramezani et al. [14] developed a Task Based System Load Balancing (TBSLB) approach to achieve system load balancing and confront with the lack of capacity for executing new task in one VM, by assigning the task to another homogeneous VM in a cloud environment. In addition, they proposed an algorithm to resolve the issue of migrating these tasks into a new VM host, which is a multi-objective problem in order to minimize cost, minimize execution time, and transfer time. To resolve this issue, they applied Multi-Objective Genetic Algorithm (MOGA).

Fahimeh Ramezani et al. [10] presented TBSLB-PSO method to achieve system load balancing in the cloud environment by migrating arrival tasks from an overloaded VM into a new homogeneous VM using PSO, without migrating the entire VM. This algorithm comprises of a task migration optimization model that reduces task execution time & task transfer time. In addition, they extended the CloudSim package and combined it with the Jswarm package to apply their PSO-based task-scheduling model as the task-scheduling algorithm in CloudSim and evaluated their proposed model.

3. FCM-BPSO TASK-SCHEDULING SYSTEM FOR ENERGY EFFICIENT LOAD BALANCING IN CLOUD COMPUTING

Rapidly increasing of number of clients and their demands of services also increasing due to reason of exponential development of cloud computing which leads to computing resources. These environments require a great efficient load-balancing algorithm intended for appropriate utilization of the resources. The virtualization method has enhanced utilization & system load balancing through permitting VM migration, and has delivered important advantages for cloud computing. VM migration specifically has been employed to flexible resource allocation or reallocation through transferring executing VMs from one physical machine to another machine for stronger computation power, larger memory, fast communication capability, or energy savings. The problems in traditional virtual machine (VMs) migration is presented in section 1.

To overcome those problems authors Fahimeh Ramezani et al. [10], [14] developed a Task Based System Load Balancing (TBSLB) that succeeds system load balancing only through transferring additional tasks from an overloaded VM without migrating the entire overloaded VM. They have used PSO algorithm for selecting the host VM in terms of minimizing the task execution time and transferring time. The drawback of previous work is they do not concentrate energy in the objective function for electing the host VMs and they have used constraints for choosing the candidate VM that leads to give more number of candidate VMs that makes PSO become more complex to return the optimized result.

Here, Fuzzy C Means clustering technique is utilized for grouping VM's having similar characteristics of the overloaded VMs such as number of CPU, capacity of CPU, memory and bandwidth. This leads to reduce the complexity of the PSO algorithm. The proposed load balancing methodology also considers the Energy function as objective function to select the host VM.

Once a virtual machine identified as overloaded VM_r, FCM clustering algorithm is used which helps to find candidate VM (VM_r^z) having the same capabilities as the overloaded VM (VM_r), such as number of CPU, capacity of CPU, memory and bandwidth. Then PSO algorithm selects set of target VMs based on the multiple objective fitness function from the set of host VMs (i.e. VM_r^Z) to schedule the overloaded tasks from VM_r.

3.1 Selection of Candidate Virtual Machines

With the intention of Migrating the extra task from an overloaded VMs to host VMs, initially, our proposed algorithm selects the set of candidate VMs for each overloaded VM. The set of candidate VMs should have similar properties of overloaded VMs. To find the similar properties, in this paper, we utilize FCM clustering algorithm for grouping the similar type of VMs with respect to the properties of overloaded VMs such as number and speed of the processors and memory, status of (idle or active) etc. These VMs properties are considered as features of FCM algorithm and it avoids the idle state VM for host VM because an idle PM must be turned on if tasks are transferred to it, and this action will increase energy consumption and cost.

3.1.1 Fuzzy C Means Clustering Algorithm

The FCM algorithm operates by allocating membership to every data point with respect to each cluster centroid based on distance between the cluster centroid & the data point. More the data is close to the cluster centroid, more is its membership headed for the particular cluster centroid. So, summation of membership of each data point should be equivalent to one. The proposed method also decreases energy consumption by avoiding the selection of idle PMs as the new host PMs. The clustering of VMs is based on minimization of the following objective function presented in equation (1).

$$J = \sum_{i=1}^{N} \sum_{j=1}^{C} \mu_{ij} \left\| x_i - c_j \right\| \qquad (1)$$

Where N indicates the total number of VMs in the cloud environment and C depicts the total number of clusters. The μ_{ij} represents the membership value i^{th} data point (virtual machine) with respect to j^{th} cluster. In this paper, the number of cluster defined based on the number of overloaded VMs presented in the following equation (2).

$$C = VM_R + 1 \qquad (2)$$

From the above equation 2, where C indicates the number of clusters and VM_R represents the number of overloaded VMs. Once the number of cluster has been defined, the next thing is to select the number of centroids randomly C and it can be updated using the following equation (3) at every iteration. At initial stage of clustering, the membership values between each data points with respect to every clusters can be defined randomly between 0 to 1 and with the constraints of summation of membership of each data point should be equal to one. At every successive iteration the membership values can be updated using the membership update function presented in following equation (4).

$$c_j = \frac{\sum_{i=1}^{N} \mu_{ij} \cdot x_i}{\sum_{i=1}^{N} \mu_{ij}} \quad (3)$$

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)} \quad (4)$$

From the above equation 3, where c_j indicates the updation of j^{th} centroid and x_i represents the data point and $\|x_i - c_j\|$ signifies the Euclidian distance between the i^{th} data point with respect to j^{th} centroid, and $\|x_i - c_j\|$ denotes the Euclidian distance between the i^{th} data point with respect to all other centroids except j^{th} centroid. With the aid of equation 3 and 4, the centroids and membership values are updated at each iteration. This process is repeated until minimum value of J stated in equation (1) is achieved.

3.1.2 Computation of Candidate Virtual Machines from Resultant Partitions

From the FCM clustering algorithm, we obtain C number of clusters, and set of centroids during the K number of iterations $C = \{c_j^k\}$ where $1 \le k \ge K$ and $1 \le j \ge C$. Each overloaded virtual machine VM_t^r are compared with set of centroids c_j^k to select the proper set of candidate VM. Consider the set of overloaded virtual machine $VM_R = \{VM_r\}$ $1 \le r \ge R$ for each overloaded virtual machine VM_r has set of candidate virtual machine VM_Z can be represented as $VM_r^Z = \{VM_r^z\}$ $1 \le z \ge Z$ which is obtain from the equation (5).

$$VM_r^Z = \{p_c | \arg\min \|x_r - c_j^k\|\} 1 \le k \ge K, 1 \le j \ge C \quad (5)$$

From the above equation 5, where p_c indicates the partition with respect the centroid which satisfies the condition of $\arg\min \|x_r - c_j^k\|$ and x_r represents the features of overloaded virtual machine. Likewise, the candidate virtual machines are obtained for each overloaded virtual machine and the total number of candidate VMs collected as presents in following equation (6).

$$CD(VM) = \left[\bigcup_{r,z=1}^{R,Z} (VM_r^Z)\right] \quad (6)$$

The host VMs are selected from the set the candidate VMs, the selection process can be taken by the BPSO algorithm.

3.2 Particle Swarm Optimization

Initially, the population for the PSO is initiated randomly $\boldsymbol{POP^k}$ with particles X_i^k can be represent as $pop^k = [X_1^k, X_2^k, X_3^k, \ldots, X_n^k]$ where $\boldsymbol{POP^k}$ is the set of n particle solutions in the swarm at iteration k. From $\boldsymbol{POP^k}$, where X_i^k indicates the candidate solution i in swarm at iteration k. Each of the candidate solution i represented by a d-dimensional vector and can be defined as $X_i^k = [x_{i1}^k, x_{i2}^k, x_{i3}^k \ldots x_{iid}^k]$. Each particle has their own velocity which can be described as $V_i^k = [v_{i1}^k, v_{i2}^k, v_{i3}^k \ldots v_{iid}^k]$ where $v_{i\,id}^k$ is the velocity with respect to d^{th} dimension.

The table 1 represents the sample solution representation of proposed binary particle swarm optimization algorithm. Consider the overloaded virtual machines (VM_{r5}) and (VM_{r8}) from which there are three overloaded task $t_{R5} = \{t_1, t_2, t_3\}$ from (VM_{r5}) and four overloaded task $t_{R8} = \{t_5, t_6, t_7, t_8\}$ from (VM_{r8}). For migrating the overloaded tasks t_{R5} from (VM_{r8}), and t_{R8} from (VM_{r8}) set of candidate VMs are defined by FCM algorithm $VM_{r5}^Z = VM_2^z, VM_3^z, VM_6^z$ and

$VM_{r5}^{Z} = VM_7^z, VM_9^z, VM_{10}^z$ respectively. From the table 1, X_1, X_2 and X_3 represents the sample solutions representation and $F(X_i)$ indicates the multi-objective fitness for the solution X_i. The solutions are represented in terms of binary values such as '0' and '1'. The binary term '1' depicts that corresponding task t_i is allocated to the candidate virtual machine VM_z otherwise the value will be '0'.

Table 1 Sample Solution Representation Of Binary Particle Swarm Optimization

VM_R	Overloaded VM 5 (VM_{r5})									Overloaded VM 8 (VM_{r8})												Fitness $F(X_i)$
$CD(VM)$	VM 2			VM 3			VM 6			VM 7				VM 9				VM 10				
$\{t_R\}$	t1	t2	t3	t1	t2	t3	t1	t2	t3	t5	t6	t7	t8	t5	t6	t7	t8	t5	t6	t7	t8	
X_1	0	0	1	1	1	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	2.86
X_2	1	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	0	1	1	3.14
X_3	0	0	0	1	0	0	0	1	1	0	1	1	0	0	0	0	1	1	0	0	0	1.89

3.2.1 Multi Objective Fitness Function

In this paper, our ultimate aim to reduce the energy consumption in datacenter which can be attained by defining multi-objective fitness function expressed in terms of task execution time, task transfer time and energy consumption. The proposed multi-objective fitness function is given in the following equation.

$$\arg\min \left(F(X_i)\right) = Texe + Ttrans + EC \qquad (7)$$

In the above equation (7), $Texe$ represents the total execution time of migrated task, second parameter represents task transferring time during the migration process and EC represents energy consumption for the execution of migrated task. Our aim is to minimize the multi-objective fitness based on the task execution time, task transferring time and energy consumption.

3.2.1.1 Task Execution Time

The task execution time [10] in the host virtual machine can defined using the following equations.

$$Texe_y = \sum_{i=1}^{n} \left(\frac{DE_{iy}}{VM_{my} \times VM_{cy}} \right) \qquad (8)$$

$$Texe = \sum_{y=1}^{m} Texe_y \qquad (9)$$

From the above equation 8, $Texe_y$ represents task execution time at Y_{th} Virtual Machine. DE_{iy} denotes the amount of data that the task i assigns to VM_y and VMm_y indicates the amount of memory of VM_y and VMc_y indicates the number of CPU's on VM_y. Overall task execution time can be obtained using the equation 9.

3.2.1.2 Task Transferring Time

While migrating the task from one virtual machine to host virtual machine, task-transferring time is one of important factor for an optimized task-scheduling problem. The task transferring time [10] is depending on the bandwidth between the overloaded virtual machine and the host virtual machine. The task transferring time can be computed using the following equation (10).

$$Ttrans = \sum_{i=1}^{n}\sum_{y=1}^{m}\sum_{z=1}^{m} u_{iyz} \times \left(\frac{DT_{yz}}{BW_{yz}} \right) \qquad (10)$$

From the above equation 10, u_{iyz} become 1 when the i^{th} task is assigned from Y^{th} virtual machine to z^{th} virtual machine and DT_{yz} represents the volume of data transfer between the Y^{th} virtual machine to z^{th} virtual machine and BW_{yz}

bandwidth capacity between the Y^{th} virtual machine to z^{th} virtual machine.

3.2.1.3 Energy Consumption Model

The proposed approach assigns tasks to the VM's so as to save energy by efficiently utilizing the resources and improve the performance by minimizing the execution time of the jobs.

The energy consumption model for the virtual machine [15] presented in this section.

Energy consumption of a Virtual Machine can be defined as the power consumption of that Virtual Machine during t units of time. Energy consumption of a virtual machine is depends of task execution time $\boldsymbol{Texe_y}$ and power consumption PC_y for execution of the task.

Therefore, EC_y energy consumption at the Y_{th} virtual machine is given by using the following equation.

$$EC_y = PC_y \times Texe_y \tag{11}$$

Power consumption PC_y for execution of the task which can be compute using the following equation (12).

$$PC_y = (RPU_y) \times (RMU_y) \tag{12}$$

At any given time, for Y_{th} Virtual Machine, the CPU utilization RPU_y and memory utilization RMU_y can be given as

$$RPU_y = \sum_{i=1}^{n} rpu_i \tag{13}$$

$$RMU_y = \sum_{i=1}^{n} rmu_i \tag{14}$$

Where rpu_i and rmu_i are the CPU and memory utilizations of i tasks running in Y^{th} VMs.

The overall energy consumption is the summation of energy consumption of each host virtual machine.

The total energy consumption at all the m Virtual Machines given by,

$$EC = \sum_{y=1}^{m} EC_y \tag{15}$$

Once the fitness function is calculated for every particle, then the next process to find the Gbest and Pbest in which Pbest (particle's best) best value of each individual so far and Gbest (Global best) represents the best particle in overall swarm.

3.2.2 Updation of Velocity

Each particle adjusts its own position towards its previous experience and towards the best previous position obtained in the swarm. The particles are updated based on its velocity and the velocity is updated based on the Pbest and Gbest and exiting velocity. The updation of velocity is presented in equation (16).

$$V_i^{k+1} = \omega V_i^k + c_1 r_p (P_{best} - X_i) + c_2 r_g (G_{best} - X_i) \tag{16}$$

3.2.3 Updation of Particle

In this paper, binary particle swarm is applied to solve this problem and sigmoid function is used to update the particle with the aid updated velocity and the solution contains 0 or 1, so which presented in equation (17) and (18).

$$S(V_i^{k+1}) = \frac{1}{1+e^{V_i^{k+1}}} \tag{17}$$

$$X_i^{k+1} = \begin{cases} 1, & if \;\; R < S(V_i^{k+1}) \\ 0, & otherwise \end{cases} \tag{18}$$

In this study, searching process is terminated predetermined set of iterations.

4. EXPERIMENTAL RESULT AND DISCUSSION

In this section, the proposed methodology is evaluated by comparing with the existing TBSLB-PSO [10] in terms of task execution time, the number of candidate virtual machine and energy consumption. The proposed methodology is implemented using CloudSim toolkit [16]. The simulation was carried out for 80–100 VMs, 100-500 tasks of various sizes.

The initial work allocation is done by finding the proper eligible VMs for the arrived task based on the properties of VMs. Once the works are started by the eligible VMs, the additional task are needed to be allocated into other VMs. In this process our proposed methodology uses FCM-BPSO and the performance of our proposed methodology is compared with TBSLB-PSO [10]. The number of arriving works and the number of eligible VMs and the number of overloaded works are similar for the both methodologies. However, our proposed methodology performs differently for the selection of candidate VMs for transfer the overloaded tasks.

The selection of the number of candidate VMs for completing the overloaded task by TBLSB-PSO is more than our proposed algorithm. Since the calculation of transfer cost and the task execution cost of every candidate VMs should calculate for the selection of proper VMs. In our proposed method, initially we use FCM clustering algorithm for grouping the VMs based on the properties of eligible VMs. Since the number of candidate VMs get reduced which leads to increase the processing speed to select the proper candidate VMs. Based on the above process, we designed evaluation metrics based on the number of candidate VMs that should be less otherwise processing time for selection of proper VMs become high. The performance of selected VMs by our proposed methodology is evaluated based on the execution time and energy consumption.

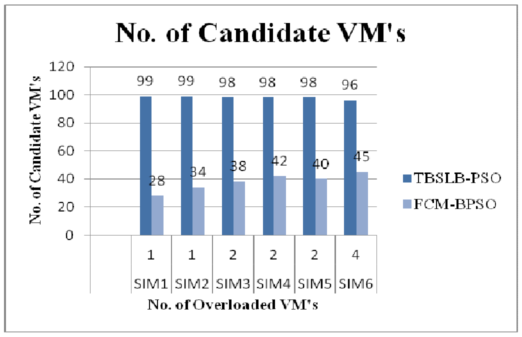

Figure 1: Comparative analysis based on the number of candidate VMs

Figure 1 represents the comparative analysis of a number of candidate VMs. The graph shows that the proposed methodology selects the less number of candidate VMs when compared with the existing methodology. The less number of candidate VMs helps the PSO algorithm to select the target VMs in a less number of iterations. So, the proposed methodology helps to reduce the selection timing of target VMs. Moreover, the selected candidate VMs have more relevant characteristics as that of the overloaded VMs which helps improvement in task execution time.

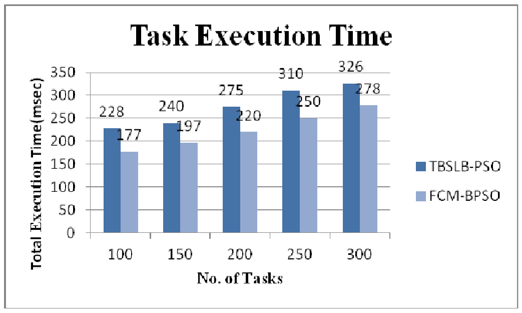

Figure 2: comparative analysis in terms of task execution time

The figure 2 depicts the comparative analysis in terms of task execution time and shows that the proposed methodology performs well in terms of total task execution time compared to TBSLB-PSO.

Figure 3 represents the comparative analysis of Energy Consumption and shows that the proposed task based load balancing algorithm has less energy consumption when compared to the existing load balancing algorithm.

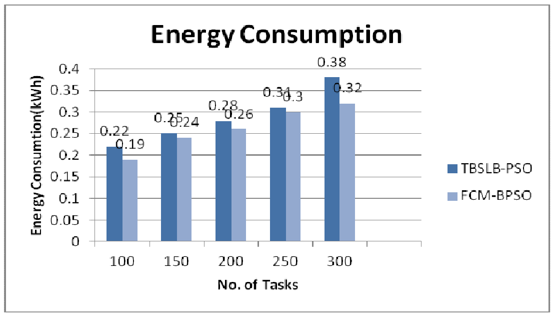

Figure 3: comparative analysis in terms of energy consumption

5. CONCLUSION

Here, we proposed FCM-BPSO task scheduling algorithm for attaining the efficient load balancing in cloud environment. Once the VMs are identified as overloaded, candidate VMs are identified from the FCM algorithm by grouping VM's with similar characteristics as that of the overloaded VMs. The BPSO algorithm based on multi-objective fitness function transfers the extra task from overloaded VMs to target VM's. The multi-objective fitness function aims to minimize the task execution time, task transferring time and energy consumption and also leads to better load balancing in cloud environment. The performance of the proposed methodology is evaluated by comparing the results with TBSLB-PSO load-balancing algorithm and simulation result shows that the proposed algorithm FCM-PSO has performed well in terms of energy consumption and task execution time.

REFRENCES:

[1] P. Mell and T. Grance, "The NIST definition of cloud computing", *Communications of the ACM*, Vol. 53, no. 6, 2010.

[2] V. Venkatesa Kumar and K. Dinesh, "Job Scheduling Using Fuzzy Neural Network Algorithm in Cloud Environment", *Bonfring International Journal of Man Machine Interface*, Vol. 2, No. 1, 2012.

[3] Y. J. Lee, G. Y. Park, H. K. Song, and H. Y. Youn, "A Load Balancing Scheme for Distributed Simulation Based on Multi-Agent System", *Proceedings of IEEE 36th Annual Computer Software and Applications Conference Workshops (COMPSACW)*, 2012, pp. 613-618.

[4] Grover, J., and Katiyar, S."Agent Based Dynamic Load Balancing in Cloud Computing", In Proceedings of International Conference on Human Computer Interactions, pp. 1-6, Chennai, 2013.

[5] R. Palta and R. Jeet, "Load Balancing in the Cloud Computing Using Virtual Machine Migration: A Review", *International Journal of Application or Innovation in Engineering & Management (IJAIEM)*, Vol. 3, No. 5, 2014.

[6] C. Clark, K. Fraser, S. Hand, G. H. Jacob, "Live migration of virtual machines", *Proceedings of 2nd ACM/USENIX Symposium on Network Systems, Design and Implementation (NSDI)*, pp. 273-286, 2005

[7] C. Jun, and C. Xiaowei, "IPv6 virtual machine live migration framework for cloud computing", *Energy Procedia*, 13, pp. 5753-5757, 2011.

[8] H. Jin, W. Gao, S. Wu, X. Shi, X. Wu, and F. Zhou, "Optimizing the live migration of virtual machine by CPU scheduling", *Journal of Network and Computer Applications*, Vol. 34, No. 4, pp. 1088-1096, 2011.

[9] X. Liao, H. Jin, and H. Liu, "Towards a green cluster through dynamic remapping of virtual machines", *Future Generation Computer Systems*, Vol. 28, No. 2, pp. 469-477. 2012.

[10] F. Ramezani, J. Lu, F. K. Hussain, "Task-Based System Load Balancing in Cloud Computing Using Particle Swarm Optimization", *International Journal of Parallel Programming*, Vol. 42, No. 5, pp. 739-754, 2014.

[11] S. Aslanzadeh, Z. Chaczko, and C. Chiu, "Cloud Computing-Effect of Evolutionary Algorithm on Load Balancing", *Computational Intelligence and Efficiency in Engineering Systems*, vol. 595, pp 217-225, 2015.

[12] D. B. LD, and P. V. Krishna, "*Honey bee behavior inspired load balancing of tasks in cloud computing environments*", Applied Soft Computing, Vol. 13, No. 5, pp. 2292-2303, 2013.

[13] Z. H. Zhan, G. Y. Zhang, Y. J. Gong, and J. Zhang, "Load balance aware genetic algorithm for task scheduling in cloud computing", *Simulated Evolution and Learning*, pp. 644-655, 2014.

[14] F. Ramezani, J. Lu, and F. Hussain, "Task Based System Load Balancing Approach in Cloud Environments", *Knowledge Engineering and Management*, pp. 31-42, 2014.

[15] N. J. Kansal, and I. Chana, "Artificial bee colony based energy-aware resource utilization technique for cloud computing", *Journal of Concurrency and Computation: Practice & Experience*, Vol. 27, No. 5, pp. 1207-1225, 2015.

[16] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F.D. Rose, R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and Experience 41 (1) (2011) 23–50.