VOL. 15, NO. 4, FEBRUARY 2020 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

SELECTION OF MIGRATION VMS AND DESTINATION PMS USING AN OPTIMIZATION ALGORITHM IN PCA-TA-IRIAL APPROACH FOR GREEN AND LOAD BALANCED CLOUD COMPUTING

V. Radhamani and G. Dalin
Department of Computer Science, Hindustan College of Arts and Science, Coimbatore, India
E-Mail: radhamaniphd2018@gmail.com

ABSTRACT

Cloud computing is a popular technology where all applications and files are hosted on a cloud. One of the most challenging issues in cloud computing is load balancing which needs to be investigated for its perfect realization. Resource Intensity Aware Load Balancing (RIAL) was proposed for load balancing in cloud computing. Based on the dynamic weight assignment to resources, the RIAL selected the Virtual Machines (VMs) from heavily loaded PMs to migrate out and placed those VMs in lightly loaded destination PMs. An Improved RIAL was proposed to consider both the lightly loaded and heavily loaded PMs for selection of destination PMs. However, some important measures such as power consumption, temperature, and traffic were not considered in IRIAL while the selection of migration VMs and destination PMs. So, Power Consumption Aware- Traffic Aware- IRIAL (PCA-TA-IRIAL) method was proposed which considered power consumption, temperature and traffic measures to select the migration VMs and destination PMs. For an optimal selection of migration VMs and destination PMs, optimization algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Artificial Plant Optimization (APO) are introduced in this paper. Based on the crossover and mutation process, GA optimally selects the migration VMs and the destination PMs. PSO algorithm optimally selects the migration VMs to the destination PMs by updating the position and velocity of each particle in the population based on the cost value. APO algorithm is inspired by a tree's growing process. Based on the light intensity and photosynthesis, each branch of the tree in APO optimally selects the migration VMs and destination PMs. Thus the optimization algorithms optimally map the migration VMs and the destination PMs effectively for load balancing.

Keywords: cloud computing, load balancing, genetic algorithm, particle swarm optimization, artificial plant optimization.

1. INTRODUCTION

In the computer field, Cloud computing [2] is a modern technology which provides services to clients at any time. Cloud computing has faced many challenges including efficient load balancing, security, data center energy consumption, resource scheduling, scaling, performance monitoring, and Quality of Service (QoS) management. Because of increasing the usage of cloud, load balancing in cloud environment is more difficult. Load balancing [9] is the process of assigning and reassigning the load among various resources in order to minimize energy, cost and response time and to maximize the throughput, performance and resource utilization. Service Level Agreement (SLA) and user satisfaction could be provided by excellent load balancing techniques. Therefore providing an efficient load balancing algorithm is a key to the success of cloud computing environments.

Currently, load balancing is performed by migrating Virtual Machines (VMs) from heavily loaded Physical Machines (PMs) to lightly loaded PMs. The conventional load balancing techniques combined the utilization of different resources in selecting VMs to migrate and finding the most suitable destination PMs. For efficient load balancing, a Resource Intensity Aware Load balancing (RIAL) [15] method was proposed which migrate the VMs in heavily loaded PMs to lightly loaded PMs based on the usage intensity of resources. However in this method, only the lightly loaded PMs were considered for selection of destination PMs. So, in Improved RIAL (IRIAL) [4] both the heavily loaded PMs and lightly loaded PMs were considered for load balancing. Because it is possible that the heavily loaded PMs nearer to the PM of selected migration VM might have required resources to balance the load.

The performance of IRIAL was improved by Power Consumption Aware- Traffic Aware- IRIAL (PCA-TA-IRIAL) [3] method where different measures such as traffic, power consumption and temperature were considered while the selection of migration VMs and destination PMs. Here, the PCA-TA-IRIAL is named as Power consumption Traffic aware-Improved Resource Intensity Aware Load Balancing (PT-IRIAL). In order to optimally selects the migration VMs and the destination PMs in PT-IRIAL, optimization algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Artificial Plant Optimization (APO) algorithms are introduced in this paper. Based on the Euclidean distance, power consumption, temperature, and traffic flow measures, both the migration VMs and destination PMs are selected by GA, PSO and APO for load balancing. It optimally maps the selected migration VMs to the destination PMs effectively.

2. LITERATURE SURVEY

A soft computing based load balancing approach [10] was proposed for load balancing in cloud computing. A local optimization approach called stochastic hill climbing was used to allocate the incoming jobs to the virtual machines. This algorithm was simply a loop that continuously moved the virtual machine in the direction of

uphill (cost value). This loop continued until the virtual machines reached a peak (low-cost value) where no neighbor had a low-cost value. This variant selected at random from among the uphill moves and the probability of selection was varying with the steepness of the uphill move. However, the stochastic hill climbing algorithm gets into the local maxima problem.

An algorithm called Honey Bee Behavior inspired Load Balancing (HBB-LB) [6] was proposed to balance loads across virtual machines in the cloud with maximum throughput. HBB-LB dynamically balancing loads by modeling the foraging behavior of honey bees. The honey bees in HBB-LB algorithm adopted to find and reap food. It improved the overall throughput of processing and a priority based load balancing focused on reducing the time of a task has to wait on a queue of the virtual machine. However, the population of honey bee behavior algorithm increases the computational cost.

Load and thermal-aware VM scheduling [8] mechanism was proposed for load balancing in the cloud. This mechanism was focused on both temperature balancing and load balancing with minimum energy consumption in the cloud center. By using this mechanism, it was make sure that none of physical machines suffered from overutilization or temperature. This mechanism mapped a VM to a PM with the consideration of temperature and load of the hosts.

A State-Based Load Balancing (SBLB) algorithm [11] was proposed for load balancing in cloud computing. This algorithm dynamically assigned tasks to available hosts from a task queue. This technique prevented the host to become heavily loaded and a task should not wait for a long time in the queue. The SBLB algorithm retained two different tables based on virtual machine states. The virtual machine was placed in the busy state only if it reaches its usage threshold. Otherwise, the virtual machine was flagged as in the available state. However, this algorithm is not simulated in real-world cloud brokering environment.

An energy-aware hybrid fruit fly optimization [7] was proposed for load balancing in the cloud environment. It was based on the foraging behavior of fruit fly. It has two stages such as smell and vision-based search. When the virtual machines in the cloud are overloaded, the task of the overloaded virtual machines was removed and placed in the different virtual machine. The searching stages were utilized to find an appropriate virtual machine which was suitable for the removed to be allocated. The appropriate virtual machines were found based on multiple objectives are energy consumption, cost of the data center and response time. However, network traffic is a more important issue in cloud computing which consumes more energy and cost of the data center that is considered in energy-aware hybrid fruitfly optimization.

A heuristic based approach called dynamic cost-load aware service broker load balancing [12] was proposed in the virtualization environment for load balancing. Initially, this approach sorted data centers in region wise and then the virtual machines were arranged in increasing order of its processing speed. Then the service broker policy checked whether the virtual machine with higher million instructions per second and it satisfied the threshold condition. If it so, then the user request was allocated to that virtual machine otherwise it was assigned with next virtual machine with next highest million instructions per second. This approach reduced overall processing time and response time by assigning user request. However, this approach was implemented in the simulation environment.

3. PROPOSED METHODOLOGY

In this section, the optimal selection of migration VMs and destination PMs based on Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Artificial Plant Optimization (APO) algorithms are described in detail. These algorithms help to migrate the optimal VMs to the optimal destination PMs for load balancing in cloud based on different measures such as Euclidean distance, power consumption, temperature, and traffic flow.

3.1 Genetic algorithm based selection of migration of VMs and destination PMs

Genetic Algorithm (GA) [14] is a branch of an evolutionary algorithm which encodes a potential solution to a specific problem in a simple chromosome-like data structure. This algorithm applies recombination operators so as to preserve critical information. The fitness value of individuals of any population tends to reproduce and survive to the next generation, thus improving the fitness value of successive generations. The GA process is started with the initialization of population size. Each chromosome in the population randomly selects the migration VMs or destination PMs. Then calculate the fitness of each chromosome based on the following cost functions [3].

For the selection of migration VMs, a cost matrix of each candidate VM_{xy} in $P_x$is constructed as,

$$Cost_{xy} = Min\left(l_{xy}, Pow_{xy}, Temp_{xy}, STR_{xy}\right) \quad (1)$$

For the selection of destination PMs, a cost matrix of each candidate PM PM_p from the ideal PM is constructed as,

$$Cost_{p,xy} = Min\left(l_{p,xy}, Pow_{p,xy}, Temp_{p,xy}, STR_{p,xy}\right) \quad (2)$$

The chromosomes which have low mining cost are chosen as parent chromosomes. Then crossover process is carried out by creating a new population from two existing chromosomes. The mutation process occurs occasionally which allows the specific child to obtain the features that are not possessed by either parent. The replacement process of GA is used to decide which migration VMs or destination PMs stay or get replaced in a population. This process is repeated until the maximum number of generations. After the selection of migration VMs and destination PMs, migrate the selected VMs to the selected PMs to balance the load.

Genetic algorithm based selection of migration VMs and destination PMs

Input: Population of n chromosomes, VMs, PMs

Output: optimal selection of migration VMs and optimal selection of destination PMs

1. Each chromosome randomly selects the migration VMs and destination PMs.
2. Evaluate the fitness of each chromosome in the population using (1) and (2).
3. Select some parent chromosome in the population according to the fitness values.
4. Perform crossover and mutation process to create a new population.
5. Replace the old population of chromosomes with the new population.
6. If the maximum number of generation is reached, then stop and return the best migration VMs and destination PMs.
7. Else, Go to step 2.
8. Migrate the selected VMs to the selected PMs.

3.2 Particle swarm optimization based selection of migration VMs and destination PMs

Particle Swarm Optimization (PSO) [13] algorithm is a stochastic, population-based optimization technique which is derived from the behavioral research on bird predation. Initially, the number of particles is initialized and each particle in D-dimensional space is denoted as $P_i = (p_{i1}, p_{i2}, \dots p_{id})$, where $i = 1,2, \dots d$ is the particle number and d denotes the dimension number of parameters defining the solutions. The particles fly over search space with adjusted velocities. Each particle of PSO keeps two values in their memory are $pbest$, i.e., its own best experience and $gbest$, i.e., the experience of the whole particles. After the initialization, each particle randomly selects the migration VMs and destination PMs. Then the cost matrix of each particle is calculated and based on its value $pbest$ and $gbest$ of each particle are selected. At each iteration in PSO, a new velocity for each particle is calculated. The velocity for each dimension is denoted as$V_i = (v_{i1}, v_{i2}, \dots, v_{id})$. The new velocity is utilized to calculate the next position of the particle in the search space. This process is repeated until a user-specified maximum iteration is achieved. It returns the optimal migration VMs and the destination PMs. Finally, the selected VMs are migrated to the selected PMs.

PSO algorithm based selection of migration VMs and destination PMs

Input: Population of n chromosomes, VMs, PMs, $gbest$=0, $pbest = 0$, $maxiter$, c_1, c_2, r_1, r_2, num_p

Output: optimal selection of migration VMs and optimal selection of destination PMs

1. Each particle randomly selects the migration VMs and destination PMs.
2. do
3. for $i = 1$ to num_p
4. Evaluate the fitness value of each particle using (1) and (2).
5. if the current fitness value is better than $pbest$
6. Set current value as the new $pbest$
7. End if
8. End for
9. Choose the particle which has best fitness value as $gbest$
10. For $i = 1$ to num_p
11. Calculate the velocity of particle using $v_i^{t+1} = v_i^t + c_1 r_1 (pbest_i^t - x_i^t) + c_2 r_2 (gbest_i^t - x_i^t)$
12. Update the position of particle using $x_i^{t+1} = x_i^t + v_i^{t+1}$
13. End for
14. while $(maxiter)$
15. End while
16. Migrate the selected VMs to the selected PMs.

In the PSO algorithm, $gbest$ denotes the global best, $pbest$ denotes the particle best, $maxiter$ denotes the maximum iteration, $c_1$and c_2 are the learning factor, $r_1$and r_2 are the random number between 0 to 1, num_p is the number of particles in population, v_i^t is the i-th particle velocity at t-th iteration and the position of i-th particle velocity at t-th iteration is denoted as x_i^t.

3.3 Artificial plant optimization based selection of migration VMs and destination PMs

Artificial Plant Optimization (APO) algorithm [5] emulates the growing phenomenon of plant and how it enables photosynthesis for promoting the growth and creation of food. The process of APO is initialized by selecting a random number of branches as the migration VMs and destination PMs which act as candidate solutions. The fitness (cost matrix) of each branch is calculated subsequently. Subsequently, operators like photosynthesis and phototropism are used in this sample. Photosynthesis operator is used to measuring the efficiency of energy production of the branch. Phototropism is used to know the direction of growth towards the light source and is an indicator of the candidate solution moving towards the optimal solution.

APO based selection of migration VMs and selection PMs

Input:m branches, $max_{iter}, iter = 1$, α, P_{max}, R_d

Output: optimal selection of migration VMs and destination PMs

1. Each branch randomly choose the migration VMs and destination PMs
2. while $(iter < max_{iter})$ do
2. For each branch i to m
3. Calculate the fitness (cost matrix) value of each branch using (1) and (2)

4. Convert the fitness values between the range from 0 to 1 using

$$\mathrm{UCost_{xy,i}(t)} = \frac{\mathrm{WCost_{xy}(t) - Cost_{xy,i}(t)}}{\mathrm{WCost_{xy}(t) - BCost_{xy}(t)}}$$

$$\mathrm{UCost_{p,xy,i}(t)} = \frac{\mathrm{WCost_{p,xy}(t) - Cost_{p,xy,i}(t)}}{\mathrm{WCost_{p,xy}(t) - BCost_{p,xy}(t)}}$$

5. Calculate the photosynthetic rate using

$$\mathrm{p_i(t)} = \frac{\alpha \mathrm{UCost_{p,xy,i}(t) P_{max}}}{\alpha \mathrm{UCost_{p,xy,i}(t) + P_{max}}} - \mathrm{R_d}$$

6. End for
7. For each branch i to n
8. if $rand_2() < pm$

$$\mathrm{x_i(t+1)} = \mathrm{x_{min}} + (\mathrm{x_{max}} - \mathrm{x_{min}})\mathrm{rand_1()}$$

9. else
10. Control the direction of growth

$$\mathrm{coe} = \begin{cases} 1, & \text{if } \mathrm{p_i(t) > p_p(t)} \\ -1, & \text{if } \mathrm{p_i(t) < p_p(t)}, \text{ where i} = \\ 0, & \text{otherwise} \end{cases}$$

$1,2, \ldots \mathrm{m}, \mathrm{p} = 1,2, \ldots \mathrm{m}$ and $\mathrm{i} \neq \mathrm{p}$

11. Calculate the growing force which is guided by photosynthetic rate

$$\mathrm{F_i(t)} = \frac{\mathrm{F_i^{total}}}{\left\|\mathrm{x_i(t) - x_p(t)}\right\|}\left(\mathrm{x_i(t) - x_p(t)}\right)$$

$$\mathrm{F_i^{total}} = \sum_{\mathrm{i \neq p}} \mathrm{coe} \times \mathrm{e^{-dimP_i(t)}} - \mathrm{e^{-dimP_p(t)}}$$

12. Update the position of ith branch at time $t+1$ suing

$$\mathrm{x_i(t+1)} = \mathrm{x_i(t)} + \mathrm{G_p} \times \mathrm{F_i(t)} \times \mathrm{rand()}$$

13. End if
14. End for
15. End while
16. Migrate the selected VMs to the destination PMs

In the above algorithm, $WCost_{xy}(t)$ and $BCost_{xy}(t)$ are the best (low cost) and worst (high cost) light intensities at time t, $Cost_{xy,i}(t)$ denotes the light intensity of branch i,$p_i(t)$ is the photosynthetic rate of branch i at time t, α is the initial quantum efficiency, P_{max} is the maximum net photosynthesis rate, R_d is the dark respiratory rate, pm denotes a probability, $rand_1()$ and $rand_2()$ are two random number with uniform distribution, $x_i(t)$ denotes the position of ith branch at time t, $\|.\|$ denotes the Euclidean distance, G_p is a parameter reflecting the energy conversion rate and used to control the growing size per unit time and $rand()$ represents a random number sampled with uniformly distributed.

4. RESULT AND DISCUSSIONS

In this section, the performance of Power consumption Traffic aware-Improved Resource Intensity Aware Load Balancing (PT-IRIAL), PT-Genetic Algorithm- IRIAL (PT-GA-IRIAL), PT-Particle Swarm Optimization-IRIAL (PT-PSO-IRIAL) and PT-Artificial Plant Optimization-IRIAL (PT-APO-IRIAL) are evaluated in terms of communication cost reduction, number of migration and performance degradation. The experiment is carried out in CloudSim simulator [1].

4.1 Communication cost reduction

The communication cost reduction is the difference between the total communication cost observed at a certain time point from the initial total cost of all VMs. The following Table-1 shows the comparison of communication cost reduction between PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for different timings.

Table-1. Comparison of communication cost reduction.

Time (hrs)	PT-IRIAL	PT-GA-IRIAL	PT-PSO-IRIAL	PT-APO-IRIAL
8	180	200	215	234
16	210	250	270	287
24	240	276	299	322

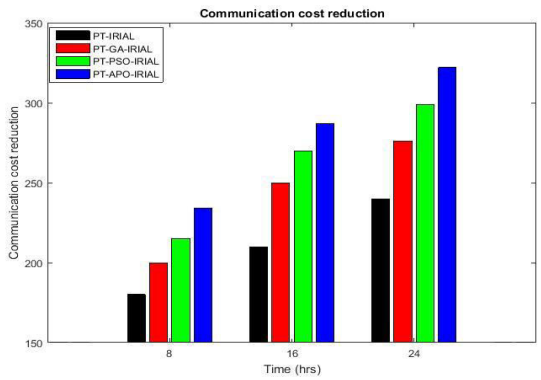

Figure-1. Comparison of communication cost reduction.

Figure-1 shows the comparison of communication cost reduction between PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for different timings. The time in hours is taken in X-axis and communication cost reduction is taken in Y-axis. At 24 hours timing, the communication cost reduction of PT-APO-IRIAL method is 34.17% greater than PT-IRIAL, 16.7% greater than PT-GA-IRIAL and 7.7% greater than PT-PSO-IRIAL. From this analysis, it is known that the PT-APO-IRIAL based load balancing has a better communication cost reduction than the other load balancing methods.

4.2 Number of migrations

The number of migrations denotes the migration of VMs from one PM to another PM. The following Table-2 shows the comparison of the number of migration by using PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for the different number of VMs.

Table-2. Comparison of number of migration.

No. of VMs	PT-IRIAL	PT-GA-IRIAL	PT-PSO-IRIAL	PT-APO-IRIAL
2500	0.67	0.51	0.47	0.42
3000	1.7	1.52	1.44	1.39
5000	2.1	1.9	1.72	1.64

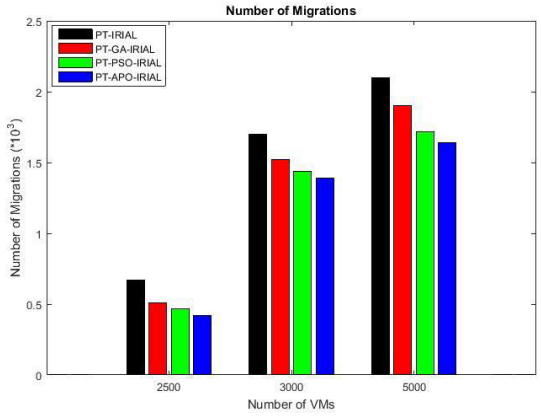

Figure-2. Comparison of number of migration.

Figure-2 shows the comparison of the number of migration using PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for the different number of VMs. The number of VMs is taken in X-axis and the number of migration is taken in Y-axis. When the number of VMs is 5000, the number of migration using PT-APO-IRIAL is 21.9% less than PT-IRIAL, 13.7% less than PT-GA-IRIAL, 4.7% less than PT-PSO-IRIAL. From this analysis, it is known that the PT-APO-IRIAL based load balancing has a better number of migrations than the other load balancing methods.

4.3 Performance degradation

When a VM is being migrated to another PM, its performance is degraded. Based on the migration of VMs performance degradation is calculated. The following Table-3 shows the comparison of performance degradation by using between PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for the different number of VMs.

Table-3. Comparison of performance degradation.

No. of VMs	PT-IRIAL	PT-GA-IRIAL	PT-PSO-IRIAL	PT-APO-IRIAL
2500	0.06	0.052	0.047	0.04
3000	0.68	0.61	0.56	0.5
5000	2.7	2.64	2.59	2.54

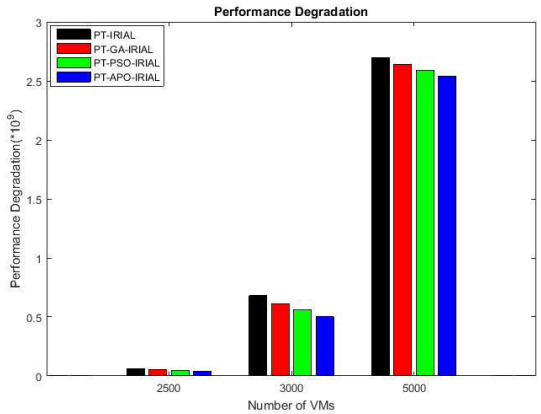

Figure-3. Comparison of performance degradation.

Figure-3 shows the comparison of performance degradation PT-IRIAL, PT-GA-IRIAL, PT-PSO-IRIAL and PT-APO-IRIAL methods for the different number of VMs. The number of VMs is taken in X-axis and the performance degradation is taken in Y-axis. When the number of VMs is 5000, the performance degradation by PT-APO-IRIAL is 5.9% less than PT-IRIAL, 3.8% less than PT-GA-IRIAL, 1.9% less than PT-PSO-IRIAL. From this analysis, it is known that the PT-APO-IRIAL based load balancing has better performance degradation than the other load balancing methods.

5. CONCLUSIONS

In this article, different evolutionary algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Artificial Plant Optimization (APO) are introduced for optimal selection of migration VMs and destination PMs for load balancing. Based on the cost matrix, each algorithm selects the VMs and PMs. The GA selects the VMs and PMs by the crossover and

mutation process. Each particle in PSO, updates the position and velocity based on the cost matrix value of VMs and PMs. The APO selects the optimal VMs and PMs based on the photosynthesis and phototropism process. Finally, the selected VMs are migrated to the selected PMs to balance the load in the cloud. The experimental results show that the proposed APO based load balancing method has better communication cost reduction, number of migration and performance degradation than the other methods.

REFERENCES

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

[2] Chiregi M and Navimipour N.J. 2017. Cloud computing and trust evaluation: A systematic literature review of the state-of-the-art mechanisms. Journal of Electrical Systems and Information Technology. 5(3): 608-622.

[3] Dalin G. and Radhamani V. 2019. PCA-TA-IRIAL: Power Consumption Aware- Traffic Aware- IRIAL a novel unified approach for Green and Load Balanced Computing in Cloud. IEEE 3rd International Conference on Engineering and Technology (ICETECH'18).

[4] Dalin G. and Radhaman V. 2018. IRIAL-an improved approach for VM migrations in cloud computing. International Journal of Advanced Technology and Engineering Exploration.5(44): 165-171.

[5] Kar A.K. 2016. Bio inspired computing–A review of algorithms and scope of applications. Expert Systems with Applications. 59: 20-32.

[6] Krishna, P.V. 2013. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing.13(5): 2292-2303.

[7] Lawanyashri M., Balusamy B and Subha S. 2017. Energy-aware hybrid fruitfly optimization for load balancing in cloud environments for EHR applications. Informatics in Medicine Unlocked.8: 42-50.

[8] Mhedheb Y., Jrad F., Tao J., Zhao J., Kołodziej J and Streit A. 2013. Load and thermal-aware VM scheduling on the cloud. In International Conference on Algorithms and Architectures for Parallel Processing Springer, Cham. 101-114.

[9] Mishra S. K., Sahoo B. & Parida P. P. 2018. Load balancing in cloud computing: A big picture. Journal of King Saud University-Computer and Information Sciences.

[10] Mondal B., Dasgupta K and Dutta P. 2012. Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. Procedia Technology.4: 783-789.

[11] Naha R. K and Othman, M. 2016. Cost-aware service brokering and performance sentient load balancing algorithms in the cloud. Journal of Network and Computer Applications.75: 47-57.

[12] Rekha P. M and Dakshayini, M. 2018. Dynamic Cost-Load Aware Service Broker Load Balancing in Virtualization Environment. Procedia Computer Science.132: 744-751.

[13] Rezaee Jordehi A and Jasni J. 2013. Parameter selection in particle swarm optimisation: a survey. Journal of Experimental & Theoretical Artificial Intelligence.25(4): 527-542.

[14] Ritwik K and Deb S. 2011. A genetic algorithm-based approach for optimization of scheduling in job shop environment. Journal of Advanced Manufacturing Systems.10(02): 223-240.

[15] Shen H. 2017. RIAL: Resource intensity aware load balancing in clouds. IEEE Transactions on Cloud Computing. (99): 1-14.