IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-6, June 2019 E-ISSN: 2347-2693

Improving QoS Parameters for Cloud Data Centers Using Dynamic Particle Swarm Optimization Load Balancing Algorithm

Sanjay G. Patel[1*], S.D. Panchal[2]

[1]Computer Engineering Department, LDRP-ITR, Gandhinagar, Gujarat
[2]Information Technology Department, VGEC Chandkheda, Gujarat

DOI: https://doi.org/10.26438/ijcse/v7i6.337342 | Available online at: www.ijcseonline.org

Accepted: 10/Jun/2019, Published: 30/Jun/2019

***Abstract*-** Due to popularity of Cloud computing environment, the cloud computing users are increasing day by day and that has become one of the important challenge for the cloud providers in terms of load balancing. Load balancing distributes the traffic evenly over multiple paths. In this research work, we have proposed the Dynamic Improved PSO Load balancing algorithm and implement it over CloudSim toolkit. This toolkit assisted the modeling and generation of virtual machines in a simulated manner such that datacenters, jobs and their mapping to VMs can be done on a same node whereas provide the desirable result. Therefore, the results are compared with the existing load balancing algorithms namely Modified Throttled, FCFS and Particle Swam Optimization based on their performance using CloudSim Simulator. Simulation outcomes are recorded in terms of the Response time and datacenter processing time of these algorithms along with its performance and cost details.

***Keywords*:-** Cloud Computing, Load Balancing, Virtual Machine, Scheduling, Particle Swarm Optimization, Modified Throttled, FCFS, CloudSim, Response time, Data Center Processing Time, Cost

I. INTRODUCTION

The Cloud computing popularly termed as the computing system which offers internet based services on demand [1] in parallel and distributed environment. Requests from different users are distributed to different processors randomly which imbalances the load assignment and this is considered as one of the biggest disadvantage of cloud computing. Thus, loads needs to be managed. Load balancing is the strategy of dividing the workload between many computers or datacenters equally in order to enhance the performance and makes the process faster. Many algorithms are proposed through which load can be distributed equally and with minimum Response time. Load Balancer is use to balance the load. Load balancing concept can be further classified into two category i.e. static and dynamic load balancing algorithm [2]. Static Load Balancing algorithm checks the current state of the node algorithm and distributes the requests on a fixed set of rules depending on the input requests. Dynamic Load Balancing algorithm checks the previous state as well as the current node and adjusts traffic distribution evenly in real time.

The overall paper is arranged in a planned way as follows: Section 2 provides the literature survey. Section 3 gives introduction about PSO algorithm. Section 4 includes proposed system architecture and algorithm. Section 5 gives the simulation details and result analysis. Sections 6 conclude the paper.

II. LITERATURE SURVEY

In accordance with the two environments i.e. static and dynamic, many algorithms [3] have been proposed. In this section we have discussed the most known contributions in the literature for load balancing in cloud computing. Some of the load balancing algorithms discussed below:

2.1 Round Robin Scheduling Algorithm

Round Robin algorithm [4] is considered as one of the simple and static load balancing algorithm in which time sharing of jobs is equal and requests are assigned in a circular queue. The data centre controller then assigns the request to any randomly choosed VMs. All the nodes, servers are grouped together according to their processing times. Once the VMs are assigned to a job then it moves to the edge of the list. The main concern of this allotment is that the loads are not distributed uniformly and providing that the VM is not empty then the new incoming jobs have to stand in a queue. This issues would lead to low efficiency, more Response time and improper resource management. Round Robin algorithm carries the feature of low throughput and mainly removes starvation.

2.4 Modified Throttled Load Balancing Algorithm

Weighted-Round Robin scheduling algorithm [5] was propounded to expound the above issues found in Round Robin Scheduling such as starvation and priority scheduling. In Weighted- Round Robin scheduling algorithm each node

is assigned with a specific weight, so requests are received as per the assigned weight. Here uniform distribution of load takes place which would lead to high efficiency and proper resource management.

2.3 Throttled Load Balancing Algorithm

Throttled Load Balancing Algorithm [4] is a dynamic algorithm which completely deploy on virtual machines. In this allocation, the client appeals the throttled load balancer to find the suitable VM to perform the operation. The VMs are grouped according to the requests they can manage. The moment the client sends the request, the load balancer immediately gets alert and searches for the required group which can manage easily. The issue of this allocation is that the load balancer has to search for the suitable VM, which would lead to delay in operation.

2.5 Modified Throttled Load Balancing Algorithm

This algorithm is bit modified version of Throttled Load Balancing Algorithm. It [3] maintains a set of virtual machines named as VM index table and stating the position of the VMs (i.e. Busy/Available). VM at first index is initially selected depending upon the state. If VM is available, then the request is assigned and if VM is not found then it returns to the Data Centre Controller. When the next request arrives, the VM next to the already allocated VM is choosed. This process is repeated continuously until the index table size is reached.

2.6 FCFS Algorithm

It is the simplest parallel task ordering dynamic load balancing algorithm [6]. With this scheme, the user request which comes first to the data centre controller would only be allocated with the VM for first execution. The implementation of FCFS policy is easily managed with FIFO queue. The data center controller searches for VM which is free or overloaded. Then the first request from the queue is removed and is passed to one of the VM through the VMLoadBalancer. The allocation of request takes place by two ways: Firstly the requests can be arranged in a queue manner and secondly by allocating heavy load node less work and low load node more work manner. Many function parameters can be taken into consideration in order to calculate the current real load weighing value and the complex load weighing value.

III. PARTICLE SWARM OPTIMIZATION

Particle Swam Optimization is a heuristic speculative escalation technique based on swarm intelligence. Particle Swam Optimization bids on the idea of social synergy of birds and fish flock behavior. Particle Swam Optimization idea was put forward by Dr .Kennedy and Eberhant in 1995 [1]. Particle swarm is mainly of 'n' particles and the position of each particle is spaced in N-dimensional region which keeps track of its coordinates in accordance to the fitness as far as achieved [8] (analysis of particle swarm algorithm in different areas). Particle Swam Optimization Algorithm main idea lies in modification of each particle in position. This modification mainly depends on the current position, current velocity vectors , (dist:current position → pbest) and (dist:current position → gbest) where pbest – best value procured so far by any particle in its own coordinate solution space, gbest - best value procured so far by any particle in the community of the particle concern. Initialisation of each particle carrying random position and velocity speed takes place. After each particle gets intialised, evaluation of particle is done resulting a fitness value(p). If the fitness(p) is greater than fitness(pbest) than p= pbest and the best value of pbest is assign as gbest with updating the particle position and velocity vectors. In the other way, if it does not agree with the condition then again the same process of initialization starts.

IV. PROPOSED WORK

There are various algorithms designed for balancing the load among different tasks. After completing the literature survey we are able to conclude that most of the load balancing algorithms proposed so far is complex, and not able to implement. PSO algorithm task will assign to the virtual machine in best fit manner [8]. i.e. task will check all the VM and assigns the task to proper VM which will have least memory wastage. User sends their task request to the cloud server that decides the VM to store the task. Cloud server will select the VM based on PSO algorithm. Our aim is to balance the load when there is an overload in VM. First step is to upload the file and cloud server will accept the request and it will transfer that request to VM. User control will initiate the process and give control to the VM Scheduler. Main function of VM Scheduler is performs the load balancing using PSO algorithm. Based on threshold value only we are finding the overloaded VM. After finding the overloaded VM next step is to migrate task from overloaded virtual machine to under loaded virtual machine.

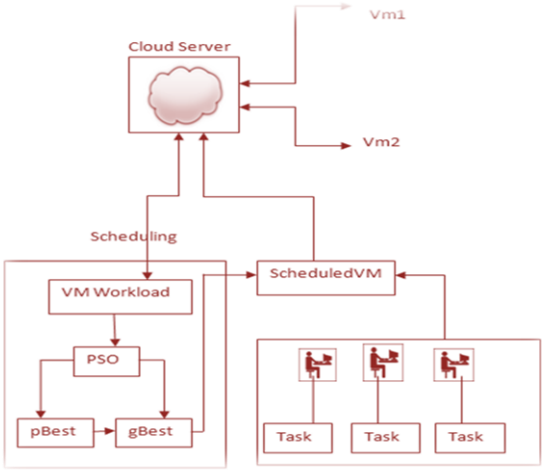

Fig.1. Proposed Architecture

The standard format of PSO algorithm is remaining same but the change is considered only in fitness function.
Improved PSO Algorithm:

1. Intialize pBest, gBest and p,S with 0s
2. Call intialize() [for particle generation]
3. Repeat while(false)
4. if pBest < gBest
5. gBest = pBest
6. do for i →0 to S
   particle.get (i)
   if testProblem (i) =Target
   set condition true for while
7. pBest= minimum() [minimum in the particle]
8. Particle.get(gBest)
9. If Target_testProblem(pBest)< Target_testProblem(p)
   p = pBest
10. a) getVelocity(gBest) [retrieve velocity of gBest particle]
    b) Updateparticle (gBest) [update particle with effect to gBest]
    c) S++ [make an increment in S's current value]
11. else set condition true for while
12. end of function

Modified particle position equation:

$$Vel_i^{k+1} = WV_e l_i^k + w_1\, random_i(\dots)\; x\; (pBest_i - s_i^k) + w_2\, random_2(\dots)\, x\, (gBest - s_i^k)$$

Weighting function equation:

$$W = w_{initial} - \left[(w_{initial} - w_{final})\, x\, I_{iter}\right]/M_{iter}$$

$$s_i^{k+1} = s_i^k + Vel_i^{k+1}$$

Where,
Vel_i^{k+1} = initial velocity of agent i at iteration k
W = weighting function
W_n = weighting factor
s_i^k = present position of agent I at iteration k
$pBest_i$ = pBest of agent i
$gBest$ = gBest of the group
$w_{initial}$ = initial weight
w_{final} = final weight
M_{iter} = maximum iteration
I_{iter} = initial iteration
s_i^{k+1} = initial searching point
Vel_i^{k+1} = modified velocity
P = random position fitness

V. SIMULATION, RESULTS AND ANALYSIS

In order to analysis the above all discussed algorithms we use the tool i.e. CloudSim for complete execution. The basic algorithm which we used to be executed in the cloudSim has simple concept without affecting the code of basic cloudSim's classes like DataCenter, VirtualMachine, and DataCenterScheduler etc where the parameter values are as under.

5.1 Experimental Setup

Table.1. Experimental Setup

Parameters	Values
Simulation Engine	CloudSim-3.0.3
Operating System	Windows 8
Front end	Eclipse
Virtual Machine Monitor	Xen
System Architecture	x86
Number of Data Centers	5
Number of Host	100
Number of VM	100
Number of Cloudlets	200
Number of Users	5
Types of workload	random
System Architecture	x86
Operating System	Windows 8
VMM	Xen

5.2 Result Analysis

Analyzed result shows that the Improved Particle Swarm Optimization Load Balancing algorithm consumes less time for overall response time and data center processing time. Here we compared the result of Improved Particle Swarm Optimization (IPSO) with few existing load balancing algorithms such as Modified Throttled load balancing (MTLB), First Come First Serve (FCFS), and Particle Swarm Optimization (PSO). Response Time of IPSO load balancing algorithm for five data centers is less as compared to other existing load balancing algorithms. We check the results for four different cases.

Case 1: In case 1 we take 5 Datacenters, 50 Cloudlets, 20 VM, 20 host and 10 number of iterations. The result is shown below:

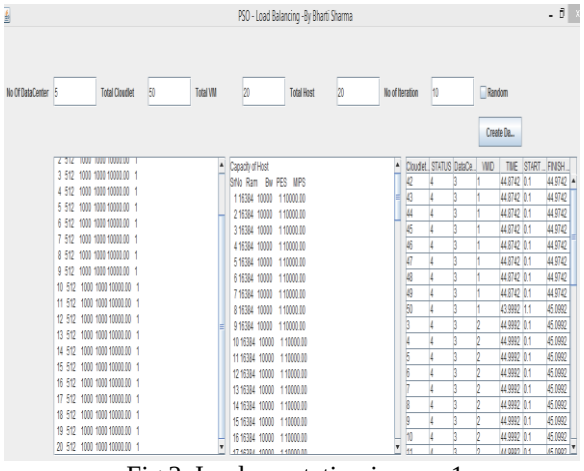

Fig.2. Implementation in case 1

Fig.3.Response time of algorithms in case 1

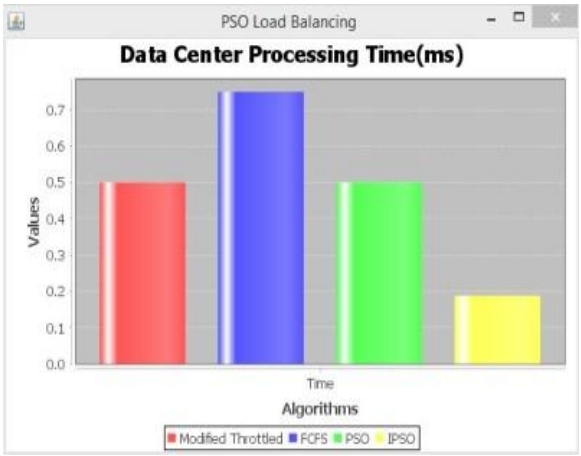

Fig.4. Data center processing time of algorithms in case 1

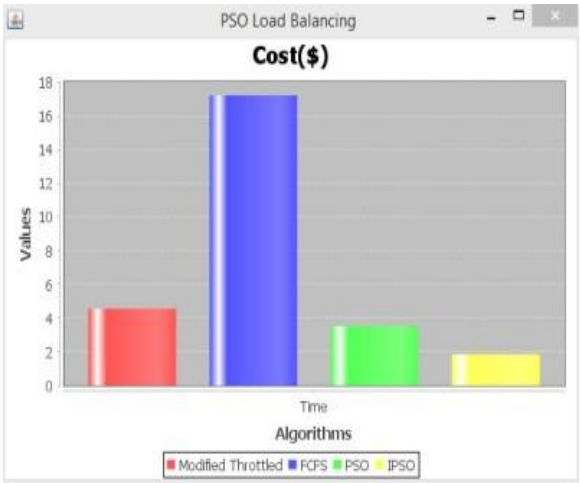

Fig.5. Cost of Algorithms in case 1

Case 2: In case 2 we take 5 Datacenters, 100 Cloudlets, 50 VM, 50 host and 20 number of iterations. The result is shown below:

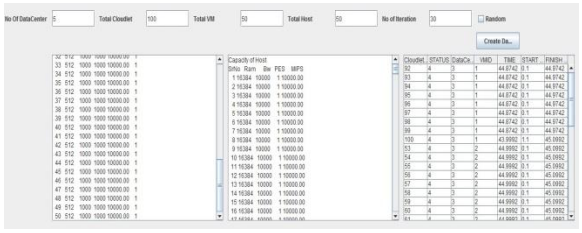

Fig.6. Implementation in Case 2

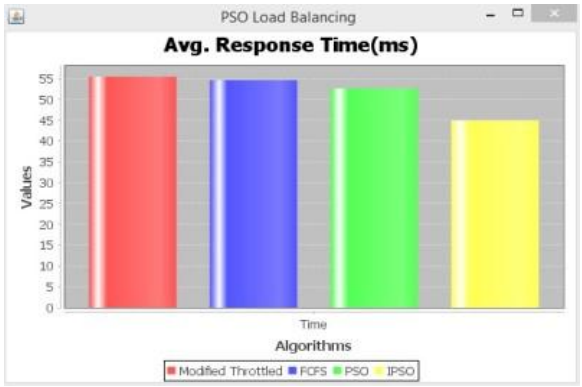

Fig.7.Response time of algorithms in case 2

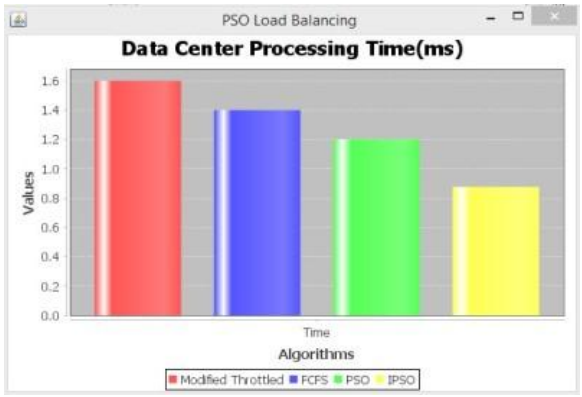

Fig.8. Data center processing time of algorithms in case 2

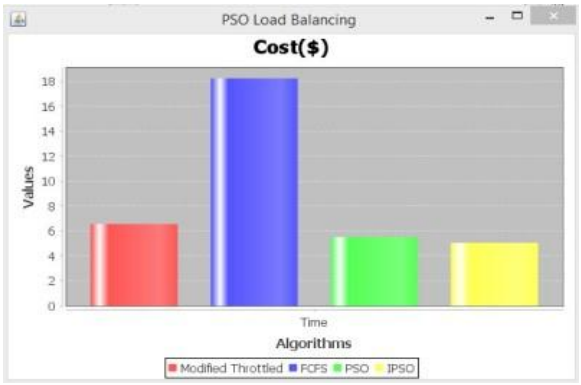

Fig.9. Cost of algorithms in case 2

Case 3: In case 3 we take 5 Datacenters, 150 Cloudlets, 100 VM, 100 host and 50 number of iteration. The result is shown below:

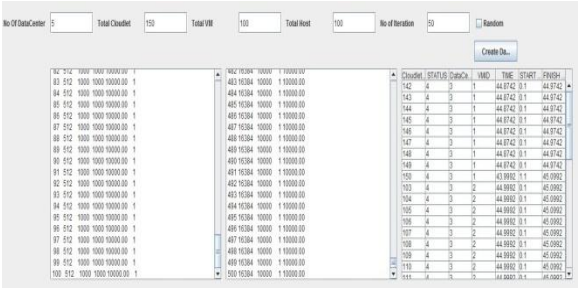

Fig.10. Implementation in case 3

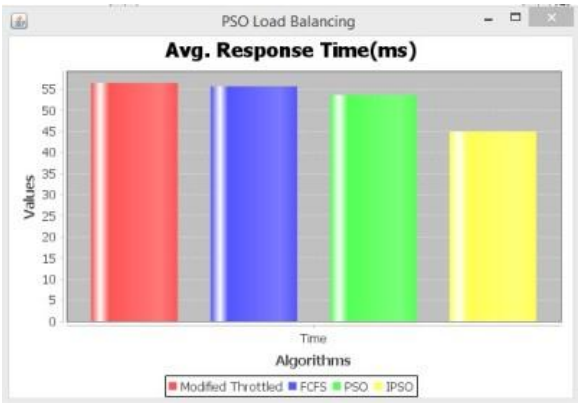

Fig.11.Response time of algorithms in case 3

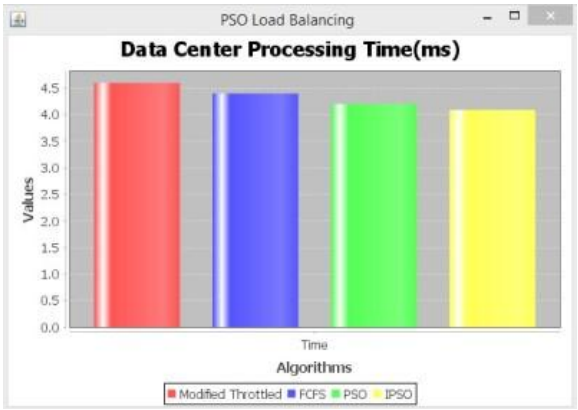

Fig.12. Data center processing time of algorithms in case 3

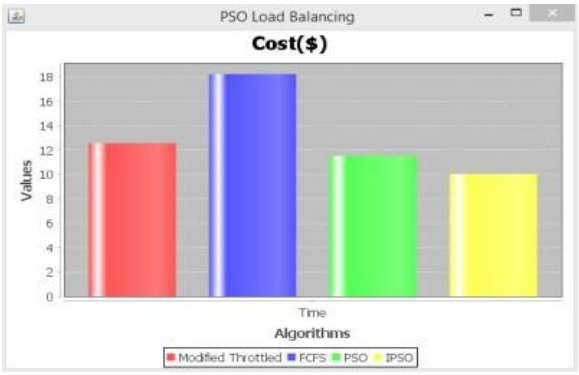

Fig.13. Cost of algorithms in case 3

Case 4: In case 4 we take 5 Datacenters, 200 Cloudlets, 100 VM, 100 host and 100 number of iteration as well as selection of VM and Host is made random. The result is shown below:

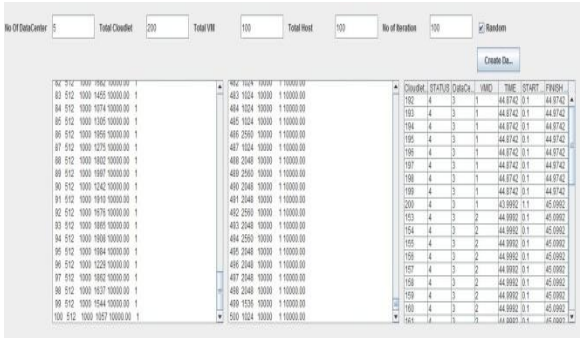

Fig.14. Implementation in case 4

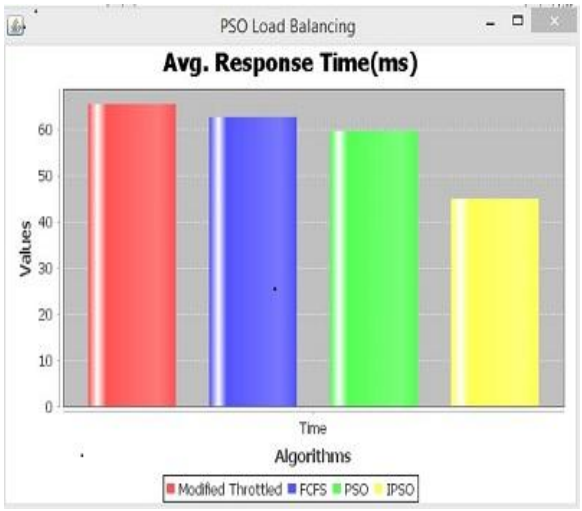

Fig.15.Response time of algorithms in case 4

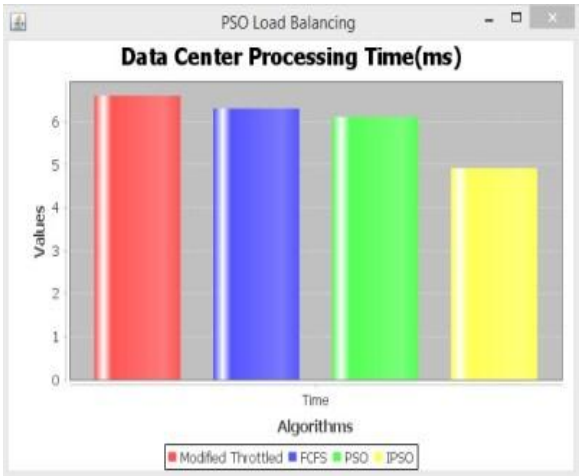

Fig.16. Data center processing time of algorithms in case 4

Fig.17. Cost of algorithms in case 4

VI. CONCLUSION AND FUTURE WORK

The greatest challenge in cloud computing world is to minimization of Response time and cost in order to balance the load and increase business performance with customer satisfaction. Considering these things in mind we have compared three major dynamic load balancing algorithms namely Modified Throttled Load Balancing Algorithm, FCFS Algorithm and Particle Swarm Optimization Algorithm. Setting the number of processors of each VMs, we found that the Response time of the Improved Particle Swarm Optimization Algorithm is efficient one as compared to other three algorithms. Also the average costs of datacenters for Improved Particle Swam Optimization is lower than others three algorithms. As cost plays a vital role in cloud environment, so reduction in cost would not only be efficient but also be top most priority for customer satisfaction. Huge amount of data transfer in a balanced manner with cheaper rate is a greater advantage in Cloud computing environment. We have taken Improved Particle Swarm Optimization Algorithm into account in order to find the optimal solution to our resource allocation which provides better distribution map. Future work can be focused on proposing new modified improvise algorithms and implementing those in real world.

In future work we will implement proposed algorithm (Improved Particle Swarm Optimization Dynamic Load Balancing Algorithm) in Open stack for the real environment implementations and then we will develop this algorithm using java (Eclipse as a run time environment).

REFERENCES

[1] P. J. Angeline, "Using selection to improve Particle Swarm Optimization," Proc. IEEE Int. Conf. Computational Intelligence, pp.84-89, 1998

[2] W. Li, H. Shi, "Dynamic Load Balancing Algorithm Based on FCFS," IEEE (ICICIC) Fourth International Conference on Innovative Computing, Information and Control, pp.1528 - 1531, December 2009

[3] S. G. Domanal, G. R. Mohana Reddy, "Load Balancing in cloud computing Using Modified Throttled Algorithm," IEEE International Conference on Cloud Computing in Emerging Markets (CCEM).

[4] S. Mohapatra, K. Smruti Rekha, S. Mohanty, "A comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing," IJCA Journal, vol. 68(6), pp. 34-38, April 2013.

[5] B. Mondal, K. Dasgupta, P. Dutta, "Load Balancing in Cloud computing using stochastic hill climbing-a soft computing approach," In Proceedings of 2nd International Conference on Computer, Communication, Control and Information Technology (C3IT), Elsevier, Procedia Technology, vol. 4, pp. 783 -789, February 2012.

[6] B. Mondal, K. Dasgupta, P. Dutta, "Load Balancing in Cloud computing using stochastic hill climbing-a soft computing Approach, "In Proceedings of 2nd International Conference on Computer, Communication, Control and Information Technology(C3IT), Elsevier, Procedia Technology, vol. 4, pp. 783 -789, February 2012.

[7] Q. Bai, "Analysis of Particle Swarm Optimization," Computer and Information Science (CCSE), IEEE, vol. 3(1), pp. 180-184, February 2010.

[8] Anju Baby, "Load Balancing In Cloud Computing Environment Using PSO Algorithm", International Journal for Research in Applied Science and Engineering Technology, Vol 2 Issue IV, April 2014.

[9] Vidhi Tiwari1*, Pratibha Adkar2, Implementation of IoT in Home Automation using android application, International Journal of Scientific Research in Computer Science and Engineering, Vol.7, Issue.2, pp.11-16, April (2019), E-ISSN: 2320-7639.

[10] Amogha A.K., Load Forecasting Algorithms with Simulation & Coding**,** International Journal of Scientific Research in Computer Science and Engineering, Vol.7 , Issue.2 , pp.16-21, Apr-2019, E-ISSN: 2320-7639.

Authors Profile

Sanjay G. Patel: I am Prof. Sanjay Patel. I am working as an Assistant Professor at LDRP-ITR. I completed Ph.D. in Computer Engineering from CSPIT, Charotar University of Science and Technology, Changa, India in 2017. I received Master degree in Computer Science and Engineering from Nirma Institute of Technology, Nirma University, Ahmedabad. My research interest is Cloud Computing, Fog Computing, Internet of Things, Machine Learning, Deep Learning and Distributed Computing. I am interested in research work that has great impact on society and nation developments.

S.D.Panchal: Associate professor Information technology. Total teaching exp 19 yrs. Graduation in EC 1999 GECM Post Graduate CSE from Nirma 2010 Phd CE from CHARUSAT 2017

