Performance Study of Some Dynamic Load Balancing Algorithms in Cloud Computing Environment

Priyadarashini Adyasha Pattanaik
School of Computer Engineering,
KIIT University, Bhubaneswar, India
priyadarshinikiit@gmail.com

Sharmistha Roy
School of Computer Engineering,
KIIT University, Bhubaneswar, India
sroyfcs@kiit.ac.in

Prasant Kumar Pattnaik
School of Computer Engineering,
KIIT University, Bhubaneswar, India
patnaikprasantfcs@kiit.ac.in

***Abstract*—In Cloud computing environment the utilization of computing resources may be available through the service that redernal by cloud consumers to service providers over the internet. Due to popularity of Cloud computing environment, the cloud computing users are increasing day by day and that has become one of the important challenges for the cloud providers in terms of load balancing. The top most challenge of cloud service providers is load balancing which is mostly used for effective operation in cloud computing environment. However, load balancing distributes the traffic evenly over multiple paths by minimizing the Response time, optimizing the resources, maximizing the throughput and by avoiding single resource overloading. In this present work, we have focused on dynamic load balancing algorithms namely modified throttled, FCFS and Particle Swam Optimization based on their performance using Cloud Analyst Simulator. Simulation outcomes are recorded in terms of the Response time and datacenter processing time of the three algorithms along with its performance and arrival cost details.**

Keywords—Cloud computing environment; Modified Throttled Load Balancing Algorithm; FCFS Algorithm; Particle Swam Optimization Algorithm; Response time

I. INTRODUCTION

Cloud computing popularly termed as the computing system which offers internet based services on demand [12]in parallel and distributed environment. It is considered as one of the emerging IT technology which relies on distributed sharing of resources over different geographical locations to deliver services efficiently to users upon their request. 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 and it can be done in two different methods i.e. load balancing and load sharing. 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 i.e. for example load is distributed to available computers or servers but in a uniform manner equally. Whereas Load sharing can be directed as the distribution of loads to different computers or data centers but not uniformly without balancing.

Load Balancing concept can be further classified into two category i.e. static load balancing algorithm and dynamic load balancing algorithm [2]. Static Load Balancing algorithm is an algorithm that checks the current state of the node algorithm and distributes the requests on a fixed set of rules depending on the input requests. The common static load balancing algorithms are Round-Robin, Weighted RR, and Shortest Job First Scheduling Algorithm. Another type is the dynamic Load Balancing algorithm which is called as self-adaptive algorithm. This algorithm checks the previous state as well as the current node and adjusts traffic distribution evenly in real time. Throttled Algorithm, Modified Throttled Load Balancing Algorithm, FCFS Algorithm, Particle Swam Optimization Algorithm, Genetic Algorithm, Clustering Algorithm, etc are some of the common dynamic algorithms. Cloud computing scenario cannot rely on the prior knowledge of nodes capacity, processing power, memory, performances and users requirements, so dynamic load balancing algorithm fits better than static load balancing algorithm as the former one takes run time statistics for load balancing. And another greater advantage in dynamic environment lies in the flexibility of user requirements, which may change at run time.

The overall paper is arranged in a planned way as follows: Section 2 provides the related work. Section 3 proposes different load balancing algorithms. Section 4 gives the comparative study and the simulation results obtained after analysis through Cloud Analyst. Section 5 concludes the paper.

II. RELATED WORK

In accordance with the two environments i.e. static and dynamic, many algorithms [1] have been proposed. In this section we have discussed the most known contributions in the literature for load balancing in cloud computing. The proposed Modified Throttled Algorithm by Shridhar [1], describes the process of assigning incoming jobs uniformly to various virtual machines. The Response time of Modified Throttled Algorithm is compared with the existing Round Robin and Throttled algorithm. The paper results with an efficient approach to handle the load by considering the available VMs with the respective requests using Modified Throttled Algorithms. The proposed Modified Throttled

Algorithm by Shridhar gives a better Response time than Round Robin and Throttled. Wenzheng Li and Hongyan Shi [2] discussed a new kind of FCFS Algorithm which reduces the probability of front-end scheduler to turn as bottleneck. Thus results with less assess time of the scheduler. The paper results with better Response time of FCFS as compared with Round Robin. In this paper, we present a comparative study on different static and dynamic algorithms and have analysed the results of different dynamic algorithms using cloud Analyst tool. The Response time of serving different incoming jobs is computed and compared.

III. LOAD BALANCING ALGORITHMS

A. Static Load Balancing Algorithms

Some of the static load balancing algorithms is discussed below:

1) Round Robin Scheduling Algorithm: Round Robin algorithm [3] 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 is 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 (some nodes are heavily loaded and some are not) 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) Weighted- Round Robin Scheduling Algorithm: Weighted-Round Robin scheduling algorithm[10] was propounded to expound the above issues found in Round Robin Scheduling such as starvation and priority scheduling. Moreover it is referred as pre-emptive algorithm. In Weighted- Round Robin Scheduling algorithm [2][1], 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.

B. Dynamic Load Balancing Algorithms

Some of the dynamic loads balancing algorithms are discussed below:

1) Throttled Load Balancing Algorithm: Throttled Load Balancing Algorithm [3][1] 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) Modified Throttled Load Balancing Algorithm: This algorithm is bit modified version of Throttled Load Balancing Algorithm. Modified Throttled Load Balancing Algorithm [1] 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 (-1) 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, which is depicted in Fig. 1 below.

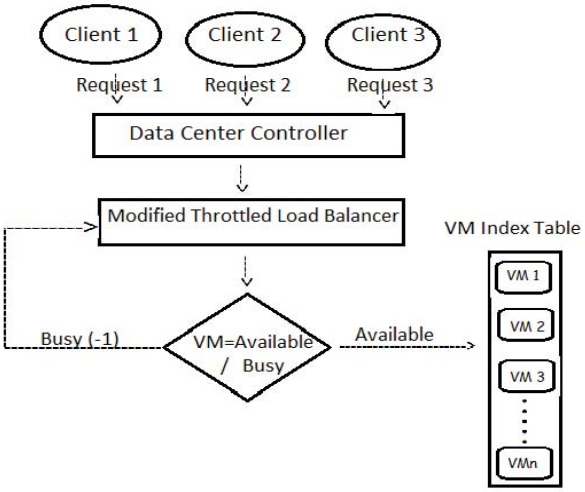

Fig. 1. Flowchart of Modified Throttled Load Balancing Algorithm

3) FCFS Algorithm : It is the simplest parallel task ordering dynamic load balancing algorithm[10]. The processing takes place by choosing the right order of jobs. 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[2] is easily managed with FIFO queue. The datacentercontroller searches for virtual machine which is in free or overloaded. Then the 1strequest 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. The whole mechanism of the algorithm is depicted in the below Fig. 2.

The ongoing real load can be calculated as follows:

$$L(Ni) = K_1 * T_{CPU}(Ni) + K_2 * BIOS_{usage} + K_3 * R_{BW} + K_4 * C_{usage} \quad (1)$$

The complex load weighing value

$$C(Ni) = A * D(Ni) + B*(D(Ni) - L(Ni))^{1/3} \quad (2)$$

where
L(Ni) - Ongoing real load on nodes
Ni - No. of nodes
T_{CPU} - CPU Tenancy Rate
C_{usage} - Cache Usage
$BIOS_{usage}$ - Basic Input and Output System usage.
R_{BW} - Network Bandwidth Tenancy Rate
K_i - set of coefficients of dynamic parameters
D(Ni) - Default weighed value of static parameters

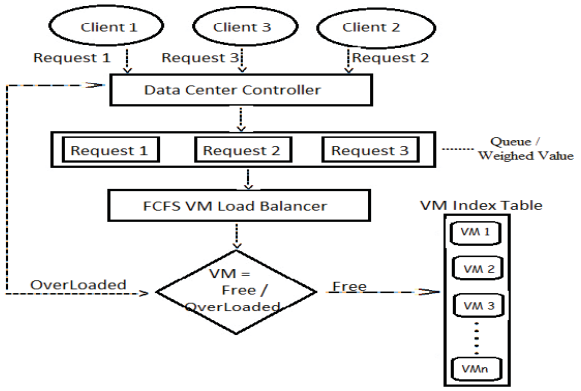

Fig. 2. Flowchart of FCFS Algorithm

4) Particle Swarm Optimization Algorithm: 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 behaviour. Particle Swam Optimization idea was put forward by Dr.Kennedy and Eberhant in 1995[12]. 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 [11] (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. The flow of the algorithm is depicted in Fig. 3. below. Here 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 doesnt agree with the condition then again the same process of initialisation starts.
Modified particle position equation:

$$Vel_i^{k+1} = WVel_i^k + w_1\ random_1(\ldots)\ x\ (pbest_i - s_i^k) + w_2\ random_2(\ldots)\ x\ (gbest - s_i^k) \quad (1)$$

Weighting function equation:

$$W = w_{initial} - [(w_{initial} - w_{final})\ x\ I_{iter}] / M_{iter} \quad (2)$$

$$s_i^{k+1} = s_i^k + Vel_i^{k+1} \quad (3)$$

where,
Vel_i^k = initial velocity of agent i at iteration k.
W = weighing function
w_n = weighing factor
s_i^k = present position of agent i at iteration k
$pbest_i$ = pbest of agent i
gbest = gbest of the group
w_{final} = final weight
$w_{initial}$ = initial 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

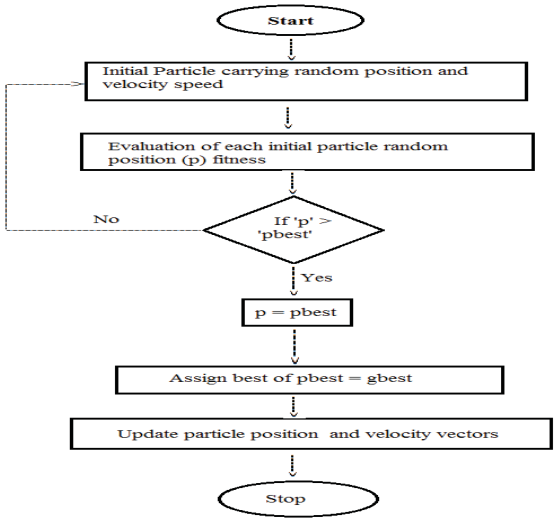

Fig. 3. Flowchart of Particle Swam Optimization Algorithm

The whole scenario of load balancing strategy can be viewed below in Fig. 4.The figure depicts the overall traffic flow management of inbound and outbound requests. Load balancing could even lead to better connectivity and provides security with proper allocation of resources as per

clients need. A number of clients get connected and send requests to get allocated with their desired resources. The whole scenario between the requests and resources gets adjusted by load balancers.

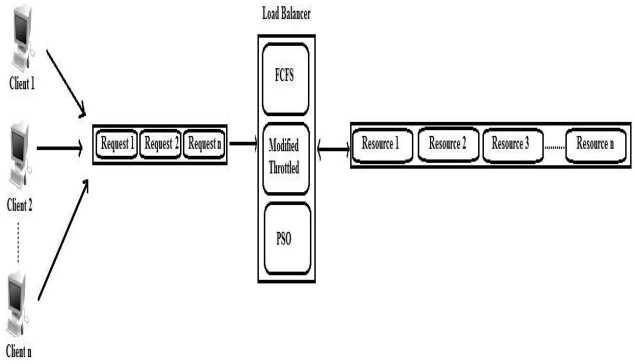

Fig. 4. Scenario of Load Balancing.

IV. SIMULATION, RESULTS AND ANALYSIS

In order to analysis the above all discussed algorithms we use the tool i.e. Cloud Analyst for complete execution. The Cloud Analyst [9] together is built on the top of CloudSim tool kit and on a comprehensive GUI which is used to configure the simulation at a high level of details. The GUI tool helps the users to build up and execute simulation experiments easily in order to get accurate results.

A hypothetical configuration has been organized using Cloud Analyst, where the whole world is divided into 6 parts or continents. i.e. with 5 user bases with peak and non peak users are taken 100 each. The user base configuration menus comprises of number of user or requests per hour, Request Size/Request, Start time and Finish Time. The Response time refers as the total arrival time and wait time together in a job queue. Individually each data center has the ability to arrange sets of virtual machines together having 4 GB of RAM,100 GB of cache space, each machine with 4 CPU, and a power of 10K MIPS. Each datacenter comprises of total number of resources, Memory size and Bandwidth.

The Configure simulation menu composes of number of users per hour, resource size with request, start and finish time respectively. Table 1 depicts the configuration below:

TABLE I. USER BASE CONFIGURATION

Name	Region	RUH (Request per user per Hr)	DS (Data size per Request) bytes	PHS (Peak Hours Start)	PHE (Peak Hours End)
U1	0	10	100	3	3
U2	1	12	1000	2	6
U3	2	15	2000	3	9
U4	3	17	3000	1	6
U5	4	20	4000	2	9

For analysis of various load balancing algorithms, proper configuration of various components of cloud analyst tool is to be done. In this paper, 5 user bases are considered with VM ranging from 20 - 100 are fixed in six different location/regions of the world. The 5 datacenters constitute of the following features including number of resources, Cache size and datacenter bandwidth respectively. The entire ordering is depicted in Table 2 below:

TABLE II. DATA CENTER CONFIGURATION

Datacenter	Number of Resources	Cache Size	Bandwidth
DC1	20	512	10000
DC2	30	1024	1000
DC3	40	512	100
DC4	25	1024	10000
DC5	15	1024	1000

Here we are comparing the three most popular dynamic load balancing algorithms namely Modified Throttled Load Balancing Algorithm, FCFS Algorithm and Particle Swam Optimization algorithm.Table1 below shows the Overall Response Time and Processing Time obtained. By comparing these algorithms we observe that the Response time of Particle Swarm Optimization Algorithm is lower than the other two algorithms i.e. Modified Throttled Load Balancing Algorithm and FCFS Algorithm. This signifies that Particle Swam Optimization Algorithm reduces the computing time of the scheduler and facilitates effective load balancing.

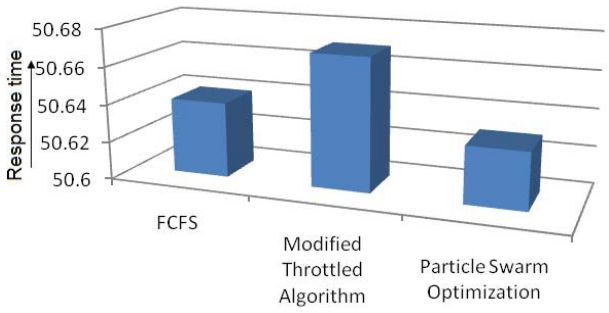

Fig. 5. Average Response Time

Fig. 5. shows that Particle Swam Optimization algorithm gives better average Response time compared to other algorithms.

With the help of cloud analyst, the average, minimum and maximum time taken by Modified Throttled Load Balancing, FCFS and Particle Swarm Optimization Algorithms on the five different user bases (UB1-UB5) are depicted in Fig. 6, Fig. 7 and Fig. 8 respectively below.

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.67	37.68	63.84
Data Center processing time:	0.50	0.01	1.06

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.17	39.05	59.30
UB2	50.83	42.04	59.57
UB3	50.05	37.68	60.64
UB4	50.51	41.00	60.68
UB5	51.42	41.17	63.84

Fig. 6. Overall Response time and Computing time of Modified Throttled Load Balancing Algorithm

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.64	40.02	60.94
Data Center processing time:	0.75	0.08	1.33

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.57	44.19	55.69
UB2	50.32	44.28	58.77
UB3	50.11	40.02	59.35
UB4	50.88	43.48	60.94
UB5	50.83	40.95	58.26

Fig. 7. Overall Response time and Computing time of FCFS Algorithm

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.63	37.67	62.05
Data Center processing time:	0.50	0.03	1.05

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.07	42.56	60.06
UB2	50.55	42.72	62.05
UB3	50.22	37.67	58.94
UB4	50.54	40.99	61.32
UB5	51.30	40.73	61.44

Fig. 8. Overall Response time and Computing time of Particle Swam Optimization Algorithm

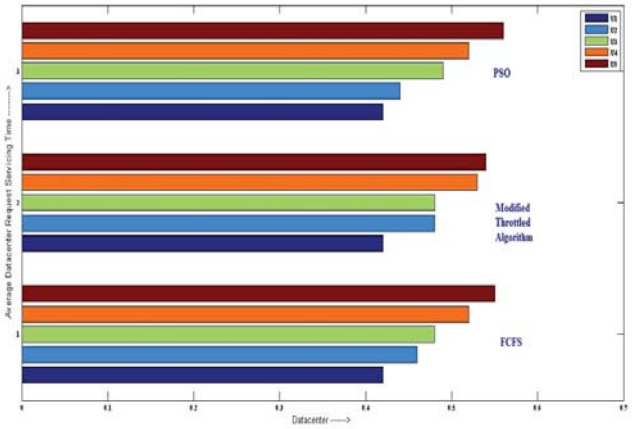

Fig. 9. Analysis of average data center servicing time of 3 different algorithms

Fig. 10, Fig. 11 and Fig. 12 depicts the overall analysis on the cost details of Modified Throttled, FCFS and Particle Swam Optimization algorithms respectively for five different data centers (DC1 to DC5). The total virtual machine cost and total data transfer cost together form the grand total cost in form of dollar. Each datacenter carries its own virtual machine details and data transfer cost. The whole analysis is obtained by using cloud analyst tool and it shows that reducing the VM cost will reduce the grand total cost which improves the efficiency of the algorithm and gives better customer satisfaction.

Cost

Total Virtual Machine Cost ($):	2.50
Total Data Transfer Cost ($):	2.06
Grand Total: ($)	4.56

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC5	0.50	0.95	1.46
DC4	0.50	0.61	1.11
DC3	0.50	0.34	0.84
DC2	0.50	0.15	0.65
DC1	0.50	0.01	0.51

Fig. 10. Modified Throttled Load Balancing Algorithm Cost Details

Cost

Total Virtual Machine Cost ($):	16.52
Total Data Transfer Cost ($):	0.72
Grand Total: ($)	17.24

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC5	2.00	0.48	2.48
DC4	3.00	0.12	3.12
DC3	4.51	0.10	4.61
DC2	4.01	0.01	4.02
DC1	3.00	0.00	3.01

Fig. 11. FCFS Algorithm Cost Details

Cost

Total Virtual Machine Cost ($):	1.70
Total Data Transfer Cost ($):	1.81
Grand Total: ($)	3.52

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC5	0.40	0.86	1.27
DC4	0.30	0.52	0.83
DC3	0.40	0.29	0.69
DC2	0.30	0.12	0.42
DC1	0.30	0.01	0.31

Fig. 12. Particle Swarm Optimization Algorithm Cost Details

Fig. 13 below shows the total cost details of different algorithms. The graph signifies that Particle Swam Optimization Algorithm carries the lowest cost as compared to the other algorithms. From the above simulation results it can be stated that, Particle Swam Optimization Algorithm is more efficient as it has least Response time and cost. Particle Swam Optimization Algorithm is best way for efficient load balancing.

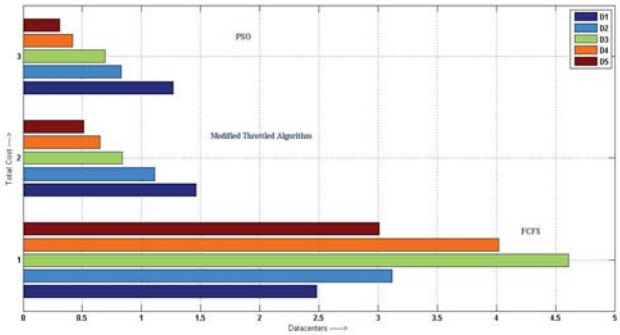

Fig. 13. Total cost details of different data centers with 3 different algorithms.

V. CONCLUSION

The greatest challenge in cloud computing world is 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 Particle Swarm Optimization Algorithm is efficient one as compared to other two algorithms. Also the average costs of datacenters for Particle Swam Optimization is lower than others two 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 Particle Swam 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.

References

[1] 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), pp.1-7, October 2013.

[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. 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.

[4] B. Wickremasingle, R.N. Calheiros, R. Buyya,"Cloudanalyst: A CloudSim-based visual modeller for analysing cloud computing," In Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA), pp. 446-452, April 2010.

[5] S. R. Jena, Z. Ahmad, "Response Time Minimization of Different Load Balancing Algorithms in Cloud Computing Environment," IJCA Journal, vol. 69(17), pp. 22-27, May2013.

[6] T. Hofmann, J. M. Buhmann, "Active Data Clustering," Centre for Biological and Computational Learning, MIT, Cambridge , pp.528 - 534, 1997.

[7] S. G. Domanal, G. R. Mohana Reddy, "Optimal Load Balancing in cloud computing By Efficient Utilization of Virtual Machines," 978-1-4799-3635-9/14, IEEE, 2014.

[8] G. Soni, M. Kalra, "A Novel approach for Load Balancing in cloud data Center," 978-1-4799-2572-8/14, IEEE, 2014.

[9] B. Wickremaisinghe, "CloudAnalyst: A cloudsim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments," MEDC Project, Cloud computing and Distributed Systems Laboratory, University of Melbourne, Australia, pp.1-44, June 2009.

[10] B. Mondal, K. Dasgupta, P. Dutta, "Load Balancing in cloudcomputing 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.

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

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