Comparative Study of Static Load Balancing Algorithms in Distributed System Using CloudSim

Supriya P Belkar and Vidya Handur, School of Computer Science & Engineering, KLE Technological University, Hubballi, *supriyabelkar27@gmail.com,vidya_handur@bvb.edu* .

***Abstract*— Distributed system is an important field both in research oriented environments and in industrial organization. A distributed system is a collection of independent nodes that appears to its users as a single system. Load imbalance is one of the main issue in distributed system which can degrade the performance of the system. Load balancing is used to improve scalability, and overall system performance in distributed systems. The performance of the system is improved by dividing the work load effectively among the participating nodes. This paper presents the comparative study of three static load balancing algorithms: Round Robin, Randomized and Threshold using CloudSim Framework which is a novel tool. The response time, processing time, resource utilization and overall data transfer cost are the main performance parameters considered to simulate. The goal of this paper is to determine the best performance of an algorithm based on the simulation result**.

***Keywords*— Distributed System, Static Load Balancing Algorithms, Data Center, User Base, VM Load Balancer.**

I. INTRODUCTION

Distributed System is a collection of large number of autonomous computer nodes. These nodes are interconnected by SANs, LANs, or WANs in a hierarchical manner. Based on IP address these nodes are uniquely identified in a network. Distributed system provides the control for sharing and aggregation of different resources such as computers, storage systems and other devices.

Distributed systems provide functional capabilities based on multiple processes, inter process communication, disjoint address space and a collective goal [5]. Different organizations or agents own these resources which are distributed. User goals, objectives, strategies and their behaviour are very difficult to characterize in a distributed system. It is a very difficult task to manage resources and applications in such systems [2].

Distributed system can be considered as a collection of nodes and communication resources shared by active users. Information and resource sharing, Clusters, Network of workstations, Distributed manufacturing system, Portable devices, Mobile computing, infrastructure and application services are some of the examples of distributed systems. A distributed system has several issues among them load balancing is an important issue [6][9].

The distribution of loads to the processing elements is generally called the load balancing problem. The load balancing problem is important when the demand for computing power increases. The purpose of load balancing is to improve the performance of a distributed system through an appropriate distribution of the application load [12]. Processing speed of a system is always highly intended. From the initial stage the development of computer it is always focused on improving the system performance [3].

This paper is organized consisting of these sections: Section II discusses the literature survey in this area. Section III discusses the proposed work done. Section IV discusses the Results and Discussions. Section V summarizes the conclusion and lastly, the references used in writing this paper.

II. LITERATURE SURVEY

Load balancing for distributed system is a problem that has been deeply studied for a long time. Load balancing means removing tasks from overloaded VMs and assigning them to under loaded VMs. Casavant and Kuhl [1] have characterized the structure and behavior of decision-making policies, in particular referring to the load sharing considering performance and efficiency. Kremien and Kramer [11] have presented an quantitative analysis for distributed system that provides both performance and efficiency measures considering load and the delay characteristic of the environment.

The probability of load imbalance in heterogeneous distributed computer system have been studied by Keqin Li [12] with a method to minimize the probability of load imbalance in the system. As suggested in on approximation algorithm by Motwani [14] it is usually hard to tell the exact difference between an optimal solution and a near-optimal solution. In a load balancing approach, tasks submitted by the users are distributed among the nodes of the system so as to distribute the workload equally among the nodes at any point of time [8]. Fengyun et al.,[15] presented a simple and effective approach of load balancing for MMORPGs and many more online games that use clusters of server for high scalability. Behavioral approach to load balancing is considered as compared to geographical approach as it suffers from the issue of crowding. The method showed high rate of scalability but there is need to conduct much research in this area.

Dimple & Atul [16] proposed an ant based framework to balance the load. In their work the author proposed an active ant at client side and at the server side both. The client ant is responsible for the request of service whereas the server ant is responsible for replying the request service. The authors improved the server performance in their work. B. Santosh Kumar [4] discussed "An Implementation of Load Balancing Policy for Virtual Machines Associated with a Data Centre".

Sandeep et al.,[3] presented the performance analysis of static and dynamic load balancing algorithms. Comparison is done by the various parameters response time, processing time, accuracy and stability etc. The load balancing algorithm [7] is selected based on situation in which work load is assigned at run time or compile time. Static algorithms are proved to be more stable as compared to dynamic load balancing algorithms.

III. PROPOSED WORK

The proposed work carried out in this paper is the comparative study of Static Load Balancing Algorithms using CloudSim Framework.

A. *Static Load Balancing Algorithms*

In Static Load Balancing algorithms [1] the processes are assigned to the processors at the compile time according to the performance of the nodes. There will be no change at runtime once the processes are assigned to the processors. Number of jobs in each node is fixed in static load balancing algorithm.

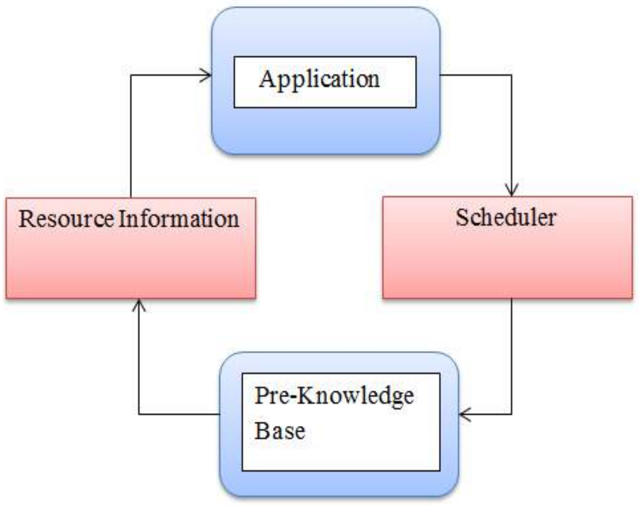

Fig. 1. Static Load Balancing

The static algorithm is easily carried into execution and takes less time, which will not refer to the states of the load nodes, but it can be only used in certain specific conditions. Although static load balancing algorithms has some drawbacks such as in its inflexibility but the speed, low delay, and easy to implement, become some of the special characteristics, that possibly make this algorithm very suitable for parallel system.

The load indices used for these static load balancing algorithms are memory, data size per request. The load also maintains a list of the jobs, their size and the resources requested.

The three algorithms have its own characteristics that will influence the algorithms performance. In this research, the three algorithms are simulated and their performance will be compared. In this paper the three static load balancing algorithms Round Robin, Randomized and Threshold are discussed below.

- Round Robin Algorithm : In this algorithm the load is distributed evenly to all the nodes. Round robin algorithm assigns tasks sequentially and evenly to all the nodes. In cloud data centers round robin algorithm [17] works by selecting a VM and then by assigning request to VMs in circular order. After allocation of request each assigned VM is moved to end of the list. The node will be back to the first node if the last node has been reached as there will be no priority. Each node maintains its load index independent of allocations from remote node.
- Randomized Algorithm : In this algorithm the node is selected on a random way, without having any information about the current or previous load on the node. Each node maintains its own load record. The computing nodes are selected randomly to balance the load. This algorithm [17] distributes the load randomly by first checking the size of the process and then transferring the load to a VM, which is lightly loaded.
- Threshold Algorithm : In Threshold Algorithm processes are assigned immediately upon creation to the computing nodes (processors). The load of a computing node can be characterized by one of the three levels which are: under loaded, medium and over loaded [10]. Two threshold values, t_upper and t_lower, that represent upper and lower threshold are considered.

Under loaded :	load < t_under
Medium :	t_under ≤ load ≤ t_upper
Overloaded :	load > t_upper

To balance the load we set some threshold values to the computing nodes. If we set threshold parameter 30% above average value then it will be highly loaded. Setting Threshold parameter 70% above average value will be lightly loaded. If the processor state is not overloaded then the process is allocated locally. VM Load Balancer will distribute some of its work to the VM if a VM is overloaded having least work so that every VM is equally loaded.

B. *CloudSim Framework*

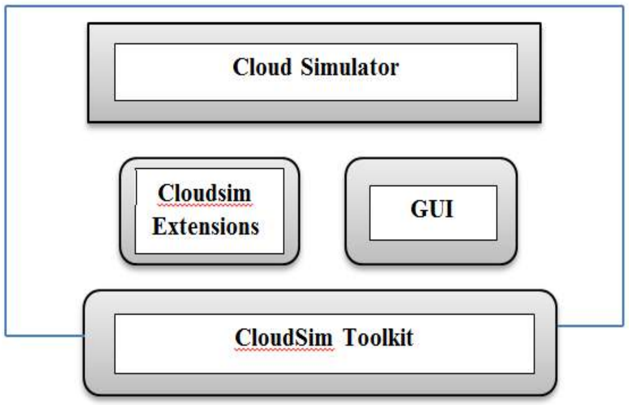

Fig. 2. CloudSim Toolkit

CloudAnalyst, [13] is a novel tool along with a new approach to simulate large-scaled applications on the Cloud. CloudSim can be used to model the data centers, service brokers, allocation policies and scheduling of a large scaled Cloud platform. It provides a virtualization engine with extensive features for modelling the creation and life cycle management of virtual engines in a data center. Such a study will provide valuable insights into designing Cloud infrastructure services in areas such as coordination between Data Centers, load balancing algorithms and possible value services such as Service Brokers to coordinate between Data Centers to optimize the application performance and cost to the owners.

C. *Methodology*

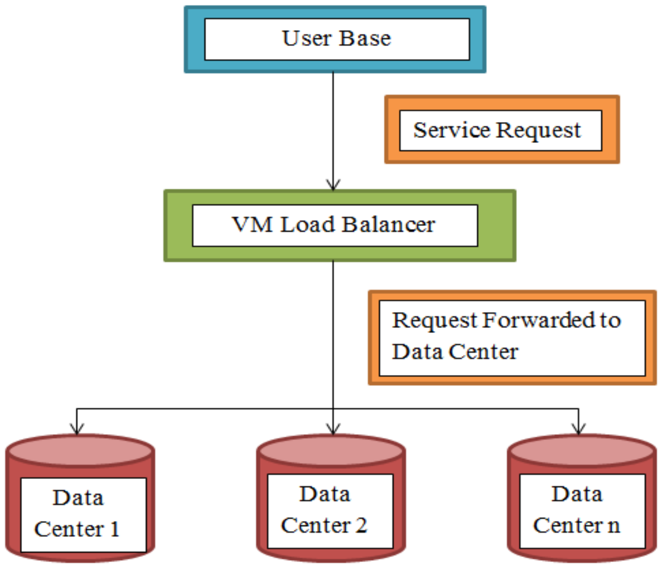

Fig. 3. Detailed Design

The detailed design carried out in this paper describes the use of VM Load Balancer to balance the load requested by the User Base to different Datacenters. It contains the following: User Base, DataCenters (DCs), VM Load Balancer.

- User Base : A User Base forms a group of users that is considered as a single unit in the simulation and its main responsibility is to generate traffic for the simulation.
- Data Center : The Data Center Controller is the most important entity in the CloudAnalyst. A single Data Center Controller is mapped to a single CloudSim. A User Base request will be forwarded to n Datacenters.
- VM Load Balancer : The Data Center Controller uses a VM (Virtual Machine) Load Balancer to determine which VM should be assigned to the next Cloudlet for processing. It balances the service requested by User Base to the Data center.

IV. RESULTS AND DISCUSSIONS

The proposed work illustrates results for simulating static load balancing algorithms.

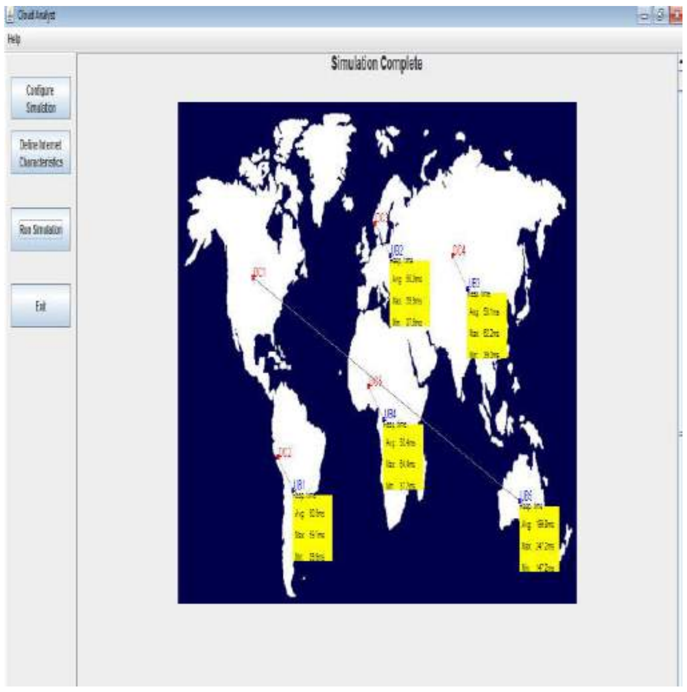

Fig. 4. Cloud Analyst Main Screen.

In this Simulation the User bases are requesting services to the nearest Data Centers located in different locations. An average response time is measured in terms of ms which balances the load among them.

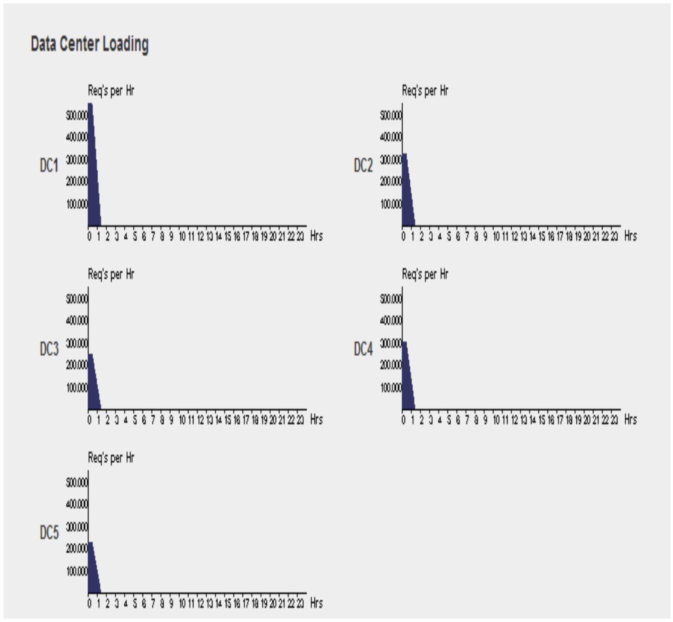

Fig. 5. Data Center Loading

Data Center Loading illustrates 5 Data Centers with 50 VMs each sharing load during peak hours clearly the heavy loads occur in the 5 Data Centers at different periods of time. If some of the load at any point in time is diverted from the most loaded data center to the lesser loaded data center in this scenario it can be analysed by using the optimized response time service broker policy in CloudAnalyst.

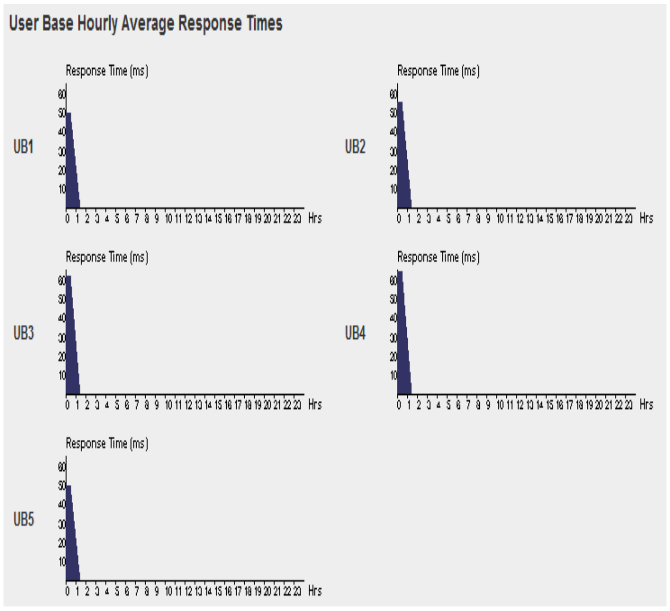

Fig. 6. Average Response Time

It illustrates the User Base Hourly Average Response Times in ms. Response time may vary to different User Bases. The performance is improved as response time is reduced.

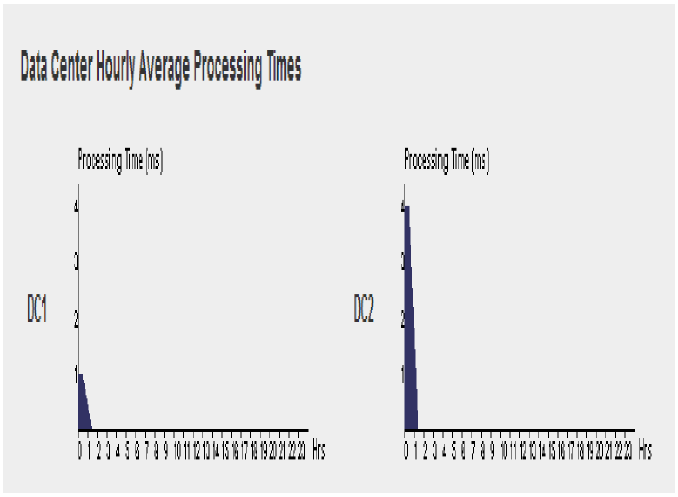

Fig. 7. Data Center Hourly Processing Time

The above graph illustrates during 24 hour period the average time taken by the data center to process a request and the number of requests processed as during the high load processing time gets dropped.

A. *Comparison*

The proposed work illustrates the comparison of Static load balancing Algorithms: Round Robin, Randomized and Threshold. The performance parameters considered here are response time, processing time and overall data transfer cost.

TABLE I. SIMULATION RESULTS OF STATIC LOAD BALANCING ALGORITHMS

Algorithms	**Number of DCs**	**Number of VMs**	**Response Time(ms)**	**Processing Time(ms)**	**Overall Data Transfer Cost ($)**
Round Robin	5	25	76.77	2.96	537.31
Randomized	5	25	75.97	2.17	537.31
Threshold	5	25	74.70	0.90	537.31
Round Robin	5	50	94.64	0.70	3652100.76
Randomized	5	50	95.88	2.09	3652100.76
Threshold	5	50	95.85	2.14	3652100.76

Here in Table I we are considering 5 DCs, 25 and 50 VMs to compare all the three algorithms. The overall data transfer cost is same but response time and processing time may vary.

TABLE II. COMPARISION OF STATIC LOAD BALANCING ALGORITHMS

Parameters	**Static Load Balancing Algorithms**		
	Round Robin	***Randomized***	***Threshold***
Response Time	Less	More compared to other algorithms	Less
Processing time	decreases	increases	decreases
Resource Utilization	Less	Less	More
Overall Data transfer cost	Remains same	Remains same	Remains same

In the above Table II illustrates comparison of the static load balancing algorithms by response time, processing time, resource utilization and data transfer cost as follows:

- Case 1: Round Robin

  Round robin is easy to implement and starvation free. There is no inter process communication and gives best performance. In Round robin algorithm the overall response time is reduced compared to other algorithms. There is minimum overhead. It is considered as best algorithm compared to other algorithms for load balancing.
- Case 2: Randomized

  It is best suited when the system has equal load on each node. Inter process communication is not required. But sometimes it may cause a single node overloaded while the other node is under loaded. It shows uneven load distribution. Response time is maximum.
- Case 3: Threshold

  It has low inter process communication. Performance is improved because of large number of local process allocations. It decreases the overhead of process allocations. It increases the execution time.

V. Conclusion

Load balancing is necessary in distributed systems if efficient and maximum utilization of resources needs to be achieved. In this paper, we have discussed the existing static load balancing algorithms available for distributed systems. Load balancing is required to distribute the workload evenly across all the nodes to achieve high performance with minimum overheads. With proper load balancing waiting time can be kept to a minimum which will further minimize the response time. The comparison shows that the static load balancing algorithms are more stable. Simulation shows that proposed strategy works and response time is reduced, processing time may vary considerably and overall data transfer cost remains same. This analysis further can also help in designing new load balancing algorithms.

References

[1] Md. Firoj Ali and Rafiqul Zaman Khan. "The Study On Load Balancing Strategies in Distributed Computing System". International SJournal of Computer Science & Engineering Survey (IJCSES) Vol.3, No.2, April 2012.

[2] Penmasta, S. and Chronopoulus, A. T. (2007), Dynamic Multi-User Load Balancing in Distributed Systems, IEEE International Parallel and Distributed Processing Symposium, 122K. Elissa, "Title of paper if known," unpublished.

[3] Sharma S., Singh S., and Sharma M. (2008), Performance Analysis of Load Balancing Algorithms, Proceedings of World Academy of Science, Engineering, and Technology, 28, 269-272

[4] B. Santosh Kumar and Dr. Latha Parthiban2, "An Implementation of Load Balancing Policy for Virtual Machines Associated with a Data Centre", International Journal of Computer Science & Engineering Technology (IJCSET), volume 5 no. 03, March 2014, p253- 261.

[5] Nathana¨el Cheriere, Erik Saule "Considerations on Distributed Load Balancing for Fully Heterogeneous Machines: Two Particular Cases" 2015 IEEE International Parallel and Distributed Processing Symposium Workshop IEEE DOI 10.1109/IPDPSW.2015.366

[6] Suriya and Prashant, "Review of load balancing in cloud computing". International Journal of Computer Science, vol. 10 issue. 1, Jan 2013.

[7] N. S. Raghava and D. Singh, "Comparative Study on Load Balancing Techniques in Cloud Computing," vol.1, no. 1, pp.18-25,2014.

[8] L.K.Dey, D.Ghosh, Satya Bagchi "Efficient Load Balancing Algorithm Using Complete Graph J.Zhang (Ed): ICAIC 2011,Part V,CCIS 228 pp:643-646,2011.

[9] Deepika, Divya Wadhawa, Nitin Kumar, "Performance Analysis of Load Balancing algorithms in Distributed Systems" Advance in Electronic and Electric Engineering. ISSN 2231-1297, Volume 4, Number 1 (2014), pp. 59-66.

[10] Daniel Grousa, Anthony T., " Non-Cooperative load balancing in distributed systems". Journal of Parallel and Distributing Computing, 2005.

[11] Orly Kremien and Jeff Kramer. Methodical analysis of adaptive load sharing algorithms. Parallel and Distributed Systems, IEEE Transactions on, 3(6):747–760,1992.

[12] K. Li. Minimizing the probability of load imbalance in heterogeneous distributed computer systems. Mathematical and computer modelling, 36(9):1075–1084, 2002.

[13] Dr. Rajkumar Buyya, "CloudSim: a toolkit for modelling and simulation of cloud computing environments and evaluation of resource provisioning algorithm", published online august in Wiley Online Library 2010, pp. 23- 50

[14] Rajeev Motwani. Lecture notes on approximation algorithms: Volume i. Dept. Comput.Sci., Stanford Univ., Stanford, CA, Tech. Rep. CS-TR-92-1435, 1992.

[15] Fengun, Simon and Graham," Load balancing for massively multiplayer online games ", Netgames 2006, ,Singapore.

[16] Dimple Juneja and Atul Garg," Collective Intelligence based framework for load balancing of web servers",IJICIT , Vol 3 No.1, Jan 2012.

[17] Jayprakash Maltare, Balwant Prajapat " Dynamic Load Balancing in cloud computing framework for load balancing of web servers International Journal of Computer Applications, Volume 148 – No.5, August 2016