Analytical study of cloud data centers using load balancing algorithm

BHAVANI KANWAR, DASHRATH SINGH, SUMAN SINGH, NEETU KANWAR

1M. tech (CSE), SIET, Sikar (Rajasthan), (India)
2Lecturer (CSE), BTIT, Pilani (Rajasthan), (India)
3Assistant Professor (CSE), SIET, Sikar (Rajasthan), India
4B.tech (ECE), BKBIET, Pilani (Rajasthan), (India)

Email address of Corresponding Author: 1bhavaninirwan@gmail.com, 2dashrath btii@gmail.com, 3er.suman05@gmail.com, 4neetukanwar027@gmail.com

http://dx.doi.org/10.22147/jucit/090603

Abstract

Cloud computing is emerging technology which is a new standard of large scale distributed computing and parallel computing. It provides shared resources, information, software packages and other resources as per client requirements at specific time. As cloud computing is growing rapidly and more users are attracted towards utility computing, better and fast service needs to be provided. For better management of available good load balancing techniques are required. So that load balancing in cloud becoming more interested area of research. And through better load balancing in cloud, performance is increased and user gets better services.

CloudSim is an event-driven simulator implemented in Java. Because of its object-oriented programming feature, CloudSim allows extensions and definition of policies in all the components of the software stack, thereby making it a suitable research tool that can mimic the complexities arising from the environments.

Cloud provides data centers distributed across geographical channels. Each DC consists of hundreds of servers called VMs. When a user submits a task, that are collected to a cloudlet. These tasks are handled by the data center controller. The data center controller uses a VMloadbalancer to determine which VM should be assigned to the next request for processing. The VMloadbalancer uses various schemes to balance the load in complex cloud environment.

Key words: Cloud Computing, Cloud Simulators, Large Data Centers, Efficient load balancing.

Introduction

In a cloud system, the number of tasks are generated randomly with different sizes from any location. This leads to the problem of task scheduling within the available resources. The cost of tasks is different when they execute on different virtual machines even if the communication links are identical. Random generation of the task may create load imbalance. Load balancing algorithms solve the problem by transferring the load from heavily-loaded VM to under-loaded VM in a given DC. The schedulers are responsible for balancing the loads across

This is an open access article under the CC BY-NC-SA license (https://creativecommons.org/licenses/by-nc-sa/4.0)
the VMs\(^1\). A load balancing algorithm attempts to improve the response time of user request by ensuring maximal utilization of available resources. There are two kinds of load balancing techniques such as static and dynamic. Static algorithms work properly when nodes have a low variation in the load. Therefore, these algorithms are not suitable for cloud environments where the load pattern changes dynamically. Dynamic load balancing algorithms are advantageous over static algorithms. But to achieve this advantage, we need to consider the additional cost associated with collection and maintenance of the load information\(^2\).

Task scheduling helps in increasing the system throughput by optimizing the average waiting time. It helps in maximizing resource utilization while minimizing total task execution time. However, load balancing not only helps in the optimal use of the resources but also helps in reducing cost and making enterprises greener. Scalability is another important feature of cloud computing, which can be achieved by proper load balancing\(^3\). The primary goals of load balancing are to

(a) improve the performance substantially,
(b) maintain system stability and reliability,
(c) create a backup plan in case the system fails even partially, and
(d) increase scalability.

**Overview:**

Load balancing is one of the important issues in cloud computing. The load in this computing environment may be a memory, processor capacity, network. It is always required to share the load among various computing nodes of the distributed system to improve the resource utilization. Load balancing ensures the even distribution of load in cloud system.

In static load balancing algorithms, the decisions related to balancing of load are made at compile time. The advantage of this algorithm is the simplicity of both implementation and overhead. Static algorithms work properly when there is a low variation in the load. Therefore, these algorithms are not well suited for the computing environments where the load varies at various points of time. Dynamic load balancing algorithms are capable of managing the dynamic change in load pattern in cloud environment Cloud provides data centers distributed across geographical channels. Each DC consists of hundreds of servers called VMs. When a user submits a task, that are collected to a cloudlet. These tasks are handled by the data center controller. The data center controller uses a VMloadbalancer to determine which VM should be assigned to the next request for processing. The VMloadbalancer uses various schemes to balance the load in complex cloud environment.

First come first serve (FCFCS) is one of the simplest task scheduling strategies. Data center controller receives the request and arranges the tasks in a queue based on their arrival time to the system. The first task is removed from the queue and assigned to available VM through VMloadbalancer. Similarly, the next tasks are assigned based on the availability of VM till the task queue is empty. So at any point of time, some VMs may be heavily loaded, and others remain idle.

Randomized scheme selects the task and randomly assigns to the available VM. The algorithm is very simple, but it does not take into consideration whether the VM is overloaded or under loaded. Hence, this may result in the selection of a VM under heavy load, and the task may require a long completion time.

Equally spread current execution load (ESCEL) algorithm works on equally spreading the execution load on different VMs. Load balancer queues up the tasks and assigns them to different VMs based on availability. The load balancer continuously scans the queue as well as the list of VMs to allocate the new requests. This offers better efficiency. However, it requires more computational overhead. In Throttled load balancer (THR) algorithm, the client first requests the load balancer to find a suitable VM to perform the required operation. If a match is found on the basis of the size and availability, then the load balancer accepts the request of the client and allocates that VM to the client. If available VM doesn’t match the criteria, then the load balancer returns (-1) and the request is queued. During allocation of a request the current load on the VM is not considered which increases the response time of a task.

Active monitoring load balancing (AM) algorithm maintains index table of each VM and their current load status. When a new request arrives, it identifies the Least.

VMloadbalancer returns the VM id to the data center controller. Data center controller sends the request to the VM identified by that id and notifies the VMloadbalancer for the new allocation. VM is allocated based on the current load. Its processing power is not taken into consideration. So the waiting time of some tasks increases violating the quality of service requirement. Min-Min scheduling algorithm starts with a set of tasks. It selects the VM with minimum completion time and the task with the minimum size. It assigns the task to the corresponding VM. After allocation the task is removed from the set and the same procedure is repeated by Min-Min algorithm until all tasks are assigned. The method is
simple, but it does not consider the existing load on a virtual machine before assigning a task. So proper load balance is not achieved in this scheme.

Max-Min algorithm works as the Min-Min algorithm. But it gives more priority to the larger tasks. The tasks that have large execution time or large completion time are executed first. The problem is that the smaller task has to wait for a long time.

PERFORMANCE EVALUATION THROUGH REGRESSION ANALYSIS:

In this section we first calculate the response time of different load balancing algorithms using the tool cloud analyst which is a cloud sim based GUI tool used for modelling and analysis of large scale cloud computing environment. Moreover, it enables the modeller to execute the simulation repeatedly with the modifications to the parameters quickly and easily. The following diagram shows the GUI interface of cloud analyst tool.

Simulation setup and analysis of results are carried out for a period of 60 hrs by taking different numbers of users, 3 data centers i.e. DC1, DC2, and DC3 having 75, 50 and 25 numbers of VMs respectively. The other parameters are fixed according to Table 1 as shown.

After performing six different experiments by cloud analyst successfully in two cases we get the overall response time of different load balancing algorithms as given in the Table 2 and Table 3.

CASE-I: VMs having Same Number of Processors:

In this case we consider all virtual machines having same number of processors i.e. quad core processors.

| Table 1. Setting of Parameters |
|--------------------------------|
| Parameter                   | Value Passed          |
| VM-image size              | 10000                 |
| VM-memory                  | 1024 MB               |
| VM-bandwidth               | 1000                  |
| Service broker policy      | Optimize response time|
| Data center architecture   | x86                   |
| Data center-OS             | Linux                 |
| Data center-VMM            | Xen                   |
| Data center-No of VMs      | DC1-75,DC2-30,DC3-50  |
| Data center-memory         | 2 GB                  |
| per machine                | 1 TB                  |
| Data center-storage        | per machine           |
| Data center-available      | 1000000               |
| bandwidth per machine      |                       |
| Data center-processor speed | 10000                |
| Data center-VM policy      | Time shared           |
| User grouping factor       | 1000                  |
| Request grouping factor    | 250                   |
| Executable Instruction length | 250                |

| Table 2. Overall Response Time for Case-I |
|------------------------------------------|
| No of users | Overall Response Time (in ms) |
|-------------|-------------------------------|
|             | RR | AM | TH |
| 6000        | 189.40 | 187.51 | 187.47 |
| 12000       | 195.65 | 195.82 | 195.67 |
| 18000       | 198.19 | 198.38 | 198.34 |
| 24000       | 199.50 | 199.56 | 199.58 |
| 30000       | 200.26 | 200.30 | 200.27 |
| 36000       | 200.88 | 200.95 | 200.88 |
| 42000       | 201.04 | 201.11 | 201.10 |
| 48000       | 201.43 | 201.50 | 201.42 |
CASE-II VMs having Different Numbers of Processors:

In this case we consider all virtual machines having different numbers of processors i.e. DC1 having the mixture of dual core and quad core processors, whereas DC2 having only dual core processors and finally DC3 have dual core, quad core and hexa core processors.

| No of Users | Overall Response Time (in ms) |
|-------------|--------------------------------|
|             | Round Robin | Active Monitoring | Throttled |
| 6000        | 194.91       | 191.21             | 192.75    |
| 12000       | 201.99       | 197.27             | 197.29    |
| 18000       | 201.57       | 199.73             | 199.15    |
| 24000       | 202.69       | 199.91             | 199.92    |
| 30000       | 204.18       | 200.45             | 200.43    |
| 36000       | 203.54       | 200.82             | 200.83    |
| 42000       | 203.79       | 201.06             | 201.06    |
| 48000       | 201.96       | 201.96             | 201.27    |

Conclusion

In case-1 we find Active monitoring load balancing algorithm and Throttled load balancing algorithm both have same correlation co-efficient value having 0.11 whereas the Round robin has 0.10. Therefore all these three algorithms are efficient where each virtual machine has same number of processors.

| Model | Unstandardized Coefficients Std. | Standardized Coefficients | T | Sig. |
|-------|----------------------------------|----------------------------|----|------|
| B     | 194.58                           | 1.252                      | 155.4 | 0.000 |
| Users | 0.000                            | 0.855                      | 4.041 | 0.007 |

Acknowledgement

It gives me immense pleasure to express my sincere thanks towards my supervisor Mrs. Suman Singh, Assistant Professor Department of Computer Science, Shekhawati Institute of Engineering & Technology, Sikar for her constant support and guidance throughout the course of this work.

References

1. Sherif Sakr, Anna Liu, Daniel M. Batista, and Mohammad Alomari, “A survey of large scale data management approaches in cloud environments”, IEEE Communications Surveys & Tutorials, Vol. 13, No. 3, (2011).
2. Milani AS, Navimipour NJ. Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. J Network Comput Appl. 71, 86–98 (2016).
3. Armbrust, M., Above the clouds: A berkeley view of cloud computing. Tech. Rep. UCB/EECS-2009-28, EECS Department, U.C. Berkeley, Feb (2009).
4. Greenberg A, Jain N et al., VL2: a scalable and flexible data center network. In: Proc SIGCOMM (2009).
5. Duncan, Dexter & Chu, Xingchen & Vecchiola, Christian & Buyya, Rajkumar. The Structure of the New IT Frontier: Cloud Computing - Part I (2018).
6. Buyya, R., Yeo, C. S., & Venugopal, S. (2008, September). Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In High Performance Computing and Communications, 2008. HPCC’08. 10th IEEE International Conference on (pp. 5-13).
7. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I., Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation computer systems, 25(6), 599-616 (2009).
8. Habib, S. M., Ries, S., & Muhlhauser, M. (2011, November). Towards a trust management system for cloud computing. In Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on (pp. 933-939). IEEE.
9. Jøsang, A., Ismail, R., & Boyd, C., A survey of trust and reputation systems for online service provision. Decision support systems, 43(2), 618-644 (2007).
10. B.kanwar, D.singh,S.singh,K.arya (2018 December) A CloudSim-based Analysing for Cloud Computing Environments and Applications Journal of Computer & Information Technology JCIT Vol. 9(6), 70-74