Optimal Load Distribution by Queue Partitioning Technique in Cloud Environment

Tejas T A
Department of Mechanical Engineering
B M S College of Engineering
(Affiliated to V.T.U)
Bangalore, India
tejasta@gmail.com

Vishnu Swaroop B T
Department of Mechanical Engineering
B M S College of Engineering
(Affiliated to V.T.U)
Bangalore, India
vishnu.swaroop63@gmail.com

Rajeshwari B S
Department of Computer Science Engineering
B M S College of Engineering
(Affiliated to V.T.U)
Bangalore, India
rajeshwari.cse@bmsce.ac.in

***Abstract*—The ambitious nature of the present digital era demands the IT industries to increase their return on investment and achieve better results. With this in mind, technologies have been developed and put into use. One such emerging technology that is making a big impact is cloud computing. Being on cloud is a benefactor to businesses and IT sector to save resources, time, investment and maintenance cost. This has led to adoption of cloud services by most of the organizations and the demand for cloud computing technology has observed steep increase with time. As a result, due to greater usage of cloud services there has been an increase in workload leading to an unequal grouping of work load among servers resulting in server being overloaded, affecting its performance. The challenge now is to assure Quality of Service (QoS) to customers by equal distribution of workload among servers. This issue is addressed in this paper by providing a solution to balance the load among the servers on the cloud and hence amplify the Quality of Service. This paper puts forward an algorithm - "Queue Partitioning Technique" where the incoming tasks queue are partitioned and tasks are distributed to virtual machines based on concept of quick sort technique. The suggested algorithm uses waiting time and load distribution as a QoS parameter and compared with two existing techniques. Proposed algorithm is implemented and evaluated using CloudSim simulator.**

Keywords—Quality of Service, Queue Partioning Technique, Quick Sort Technique, Waiting Time, Load Distribution, CloudSim Simulator.

I. INTRODUCTION

"The practice of using network of remote servers hosted on the internet to manage, store and process data, rather than a local server or a personal computers"[1]is cloud computing. Cloud computing involves IT provider offering servers, software and platforms for developing applications to use it on a 'pay-as-you-go' basis depending upon the needs of a customer. Cloud computing is offered as below three models:

- Software as a Service (SaaS) –In this service, software applications are made available to the customers over the internet on pay per usage basis.
- Infrastructures as a Service (IaaS) –Server, storage is entirely managed by an external cloud provider and provided as service to customers.
- Platform as a service (PaaS) –The entire set of resources required to build an application from the scratch is provided by the provider. Users can focus completely on developing their products instead of worrying about construction of infrastructure and maintenance.

Perks of using cloud computing

- Easy Integration – As customer demand increases, service provider can scale up the resources such as data storage, the number of servers.
- Saves money – Customers need not pay for the set-up of their infrastructure as it hosted by the provider. No license renewal issues as it is all managed by the provider.
- Ease of access – Service can be accessed by the user from any part of the world with an internet connection.
- Pay as per usage policy – Customers can pay for what they want and for how much of resources they want.
- Start-ups – Adoption of cloud will ensure a new innovation to entrepreneurs reducing cost on buying resources to start a venture.

Cloud technology has a huge potential for the future of IT industries and it plays a pivotal role in its development. However, it still faces some daunting issues like lack of security, lack of quality service, higher power consumption and distribution of workload among machines. Due to higher demand for cloud services among businesses, there is excessive task load which calls for the use of an intelligent load balancing method. Absence of an efficient load balancer would lead to un-even distribution of workload load on various servers causing the servers to crash and hampering the performance of cloud.

Proper utilization of resources, improved response time, avoiding request rejection can be achieved through an effectual load balancing technique thus demand for the need for smart load distribution strategies. The services provided by the server should be able to conform to the Service Level Agreement (SLA) agreed between the cloud service provider and the clients. SLA is a contract between service provider and end user which contains a description of the agreed service, parameters/criteria of the level of service, the assurance regarding the QoS and measures in case QoS is not satisfied by the provider under all circumstances. To comply with this agreement, incoming request will have to be properly scheduled on the server such that response time of the server is well within the time limit as mentioned in the SLA."The cloud providers agreed to the level of performance for the certain aspects of the services with the providers"[2]

Henceforth, satisfying terms and conditions mentioned in SLA offering both load balancing and quality service in the servers are essential. With this in mind, a framework is proposed in the paper that partitions the incoming task queue and tasks are distributed to virtual machines based on concept of quick sort technique which balances load among the servers, thus ensuring good response time as well as reduces processing time of tasks.

The following paper is as described below:

In Section II related work is discussed. Proposed queue algorithm is presented in section III. In section IV, experimental results of proposed queue algorithm are discussed. In section V results and conclusion is presented.

II. RELATED WORK

The main challenge faced by cloud providers is distributing the incoming requests from the customer equally among the servers. Failing to do so will cause server overloading and poor response time which in turn will decrease the effectiveness. The cloud providers negotiate with the customers and agree upon certain conditions which are provided in the Service Level Agreement. Therefore, it is crucial for the cloud providers to research upon various load balancing algorithms and keeps the time taken to process the queued requests in check with the limit specified in the SLA.

Shu Ching Wang et al., [4] presented two phase scheduling algorithms: Opportunistic Load Balancing (OLB) and Load Balance Min-Min (LBMM). The basic level is the OLB algorithm where the load or incoming requests from the customer is distributed randomly to the service manager regardless of its current load. The task assigned to the service manager is further split into subtasks. LBMM scheduling algorithm calculates execution time of each subtask on each service node and distributes subtasks to the service node that takes minimum execution time. A higher level is the Load Balance Min-Min Algorithm, where it takes into account the execution time of each subtasks at each node for the assignment of the incoming task. It assigns the incoming task to the node which has the least execution time. IvonaBrandic et al [12] discussed that in a Service Level Agreement, Quality of Service is the major parameter in the contract which is mutually agreed upon by the cloud provider and the client.

Another hierarchical cloud computing network as proposed by Shu-Ching Wang et al., [5] uses three scheduling algorithm which are Best Task Order (BTO), Enhanced Opportunistic Load Balancing (EOLB) and Enhanced Min-Min (EMM) algorithms. The BTO algorithm decides the best execution order for each task, EOLB algorithm schedules the task to a suitable service manager and the task is split into subtasks and lastly, EMM algorithm schedules the subtask to a suitable service node according to the execution time of the service node.

In Throttled Load Balancer which was suggested by Meenakshi Sharma et al., [6][7], when a task is received, the balancer checks if for a virtual machine that is idle and then it sends the idle virtual machine's ID to the data center controller which send the task to the virtual machine accordingly.

Round Robin algorithm assigns the initial set of tasks randomly to a set of virtual machines. After all the virtual machines are assigned, it schedules the next task to the next virtual machine in a circular fashion. The main fault in this type of scheduling is that sometimes servers get loaded heavily as Round robin algorithm does not take into consideration the execution time of the tasks. This algorithm was presented by Shanti Swaroop Moharana et al., [8].

Weighted Active Monitoring Load Balancing (WALB) studies the processing power of each virtual machine and assigns a weight for each machine. As the incoming tasks arrive, WALB calculates the processing time for the task and assigns a suitable virtual machine according to its weight and processing power. This type of load balancing algorithm was presented by Jasmin James et al., [16].

Another load balancing technique involves setting of priority for a node based on specified conditions of the task. It checks if the total number of requested nodes is less than the available nodes. If it is less, it schedules the task else it puts the task into queue. The task is rejected if the requested resource exceeds the limit. This type of resource allocation model was proposed by K C Gouda et al., [13].

Round robin also called as cyclic executive handles processes without priority in a circular pattern. Its implementation is easy and it allots certain time quanta to each process in the queue, if the process is not finished in allotted time it is moved back in the queue making way for the next process in line. Effective VM Load Balancing algorithm finds the expected response time of each VM. The algorithm functions in such a way that upon a request from the data center controller, the ID of the virtual machine with minimum response time is sent to the controller. The controller then allocates for the new request in the allocation table. The controller prompts the algorithm for VM deallocation when the VM finishes the processing of the request.

III. PROPOSED ALGORITHM

The tasks on the incoming queue are arranged in an increasing order on the basis of expected execution time to process them. Two pointers 'i' and 'j' point to first and last tasks in the task queue. Then incoming tasks queue are partitioned, where size of each partition is 'M' for M number of virtual machines and tasks are distributed to virtual machines based on concept of quick sort technique by incrementing pointer 'i' and decrementing pointer 'j'.

The algorithm is as follows

NOMENCLATURE:

VM:	Virtual Machine
$\mathbf{Task_{ExpET}}$**:**	Expected Execution Time for Tasks
$\mathbf{Q_{Task}}$**:**	Incoming Task Queue
$\mathbf{Q_{Sorted\text{-}TaskExpET}}$**:**	Sorted queue based on Expected Execution Time
M:	Number of Virtual Machines
N:	Number of Tasks

```
01.  $\mathbf{Q_{Task}}$ ← Incoming Tasks
02.  for each N tasks in $\mathbf{Q_{Task}}$ do
03.      Find Expected Execution Time of Task
         $\mathbf{Task_{ExpET}}$
04.  end for
05.      $\mathbf{Q_{Sorted\_TaskExpET}}$ ← Sort queue based on
         Expected Execution Time
06.      Partition the $\mathbf{Q_{Sorted\_TaskExpET}}$ into $\mathbf{M}$ sets
07.      Set the pointer
             i=0; j=N-1
08.              while(i<j)
09.                  for each virtual Machine $\mathbf{K}$ do
010.                     Schedule Task $\mathbf{t_i}$ to $\mathbf{VM_K}$
011.                     Schedule Task $\mathbf{t_j}$ to $\mathbf{VM_K}$
012.                     i++;
013.                     j--;
014.                 end for
015.             end while
016.         Repeat step 2 to 15 for next batch of N
             tasks
```

IV. RESULTS AND DISCUSSIONS

The performance of the presented strategy has been evaluated with various parameters. CloudSim 3.0.3 toolkit with NetbeansIDE8.0 has been used as simulation platform. The proposed strategy has been implemented in Java. The experimentation has been performed on a Windows 10 64-bit platform machine with Intel® Core™i7- 3770 and 4GB of DDR3 SD RAM. The efficiency of the presented technique is assessed with loads generated dynamically and the results are compared with an existing Round Robin algorithm and Effective VM Load Balancing algorithm.

Figure 1 demonstrates the distribution of loads among the virtual machines by three algorithms. Y axis denotes load distribution in percentage and X axis denotes load distribution among three virtual machines by three different algorithms. Figure shows that the proposed strategy has distributed load nearly equally among three virtual machines whereas an unequal load distribution by other two existing Round Robin algorithm and Effective VM load balancing algorithm.

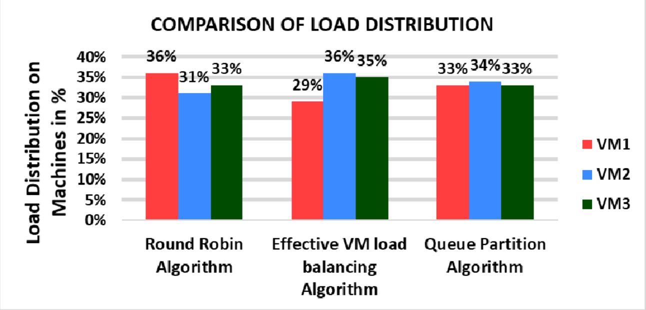

Fig. 1. Comparison on Load Distribution among Virtual Machines.

Figure 2 explains average waiting time for various scheduling algorithms. Here Round robin, Effective VM load balancing, Queue partitioning algorithm is plotted against average time taken in seconds. As we can infer from the graph, Round robin (2.3 seconds) and effective VM load balancing algorithm (2.2 seconds) considerably take more time to process compared to our proposed Queue partitioning algorithm which takes 1.6 seconds. This shows that the proposed queue partitioning algorithm responses to the incoming tasks instantly without delay.

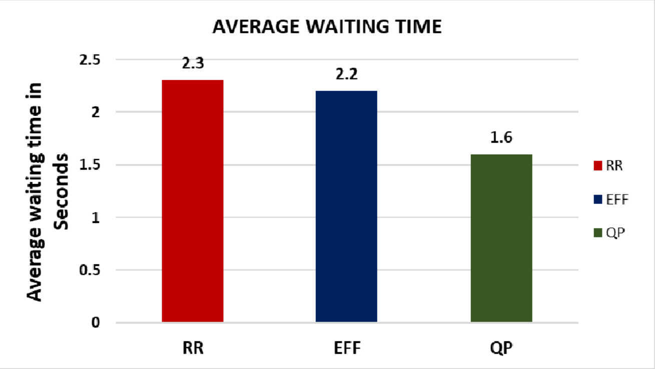

Fig. 2. Comparison on waiting times of 3 different algorithms.

V. CONCLUSION

The primary aim of this paper is to find an effective algorithm which balances load among the various servers. The algorithm used is called Queue partitioning algorithm. A detailed algorithm is presented in the paper. Also, the suggested algorithm is compared with the existing algorithms used by service providers and a comparison is made which highlights Queue partitioning algorithm as effective in terms of Load distribution as well reducing average waiting time for tasks on queue. To confirm the credibility of the algorithm graphs are plotted. The result shows that proposed queue partitioning algorithm has good response time for tasks on server and is effective in utilizing resources compared to round robin algorithm and effective VM load balancing algorithm.

ACKNOWLEDGMENT

The work reported in this paper is supported by the college through the TECHNICAL EDUCATION QUALITY IMPROVEMENT PROGRAMME [TEQIP-III] of the MHRD, Government of India.

REFERENCES

[1] K.Soniya, Dr.A.Senthil Kumar, “A Basic study on Cloud Computing,” IOSR Journal of Computer Engineering (IOSR-JCE), e-ISSN: 2278-0661,p-ISSN: 2278-8727, pp 43-46, www.iosrjournals.org.

[2] Raj Kumar Buyya, J.Broberg, A.Goscinst, "Cloud Computing: Principles and paradigms", New Jersey: John Wiley & Sons, Inc, 2011.

[3] Rajeshwari B S, Dr. M Oakshayini, "COMPREHENSIVE STUDY ON LOAD BALANCING TECHNIQUES IN CLOUD", an International Journal of Advanced Computer Technology, Volume 3, Issue 6, June 2014, pp:900- 907, ISSN: 2320-0790

[4] Shu Ching Wang, Kuo Qin Yan, Wen Pin Liao, Shun Sheng Wang, "Towards a Load Balancing in a Three Level Cloud Computing Network", 3'd International Conference on Computer Science and

Infonnation Technology, Volume I, 9'h to Il'h July 2010, pp 108-113,00l:10.1109/ICCSIT 2010.5563889

[5] Shu Ching Wang, Kuo Qin Yan, Shun Sheng, Wang, Ching Wei, Chen, "A Three Phase Scheduling in a Hierarchical Cloud Computing Network", 3,d International Conference on Communication and Mobile Computing, Taiwan, 18th to 20'h April 201 I, pp 114-117, 001: 10.1109/CMC 2011.28.

[6] Meenakshi Shanna, Pankaj Shanna, "Perfonnance Evaluation of Adaptive Virtual Machine Load Balancing Algorithm", International Journal of Advanced Computer Science and Applications, Volume 3, Issue 2, pp 86-88, 2012, ISSN: 2156-5570

[7] Meenakshi Shanna, Pankaj Shanna, Sandeep Shanna, "Efficient Load Balancing Algorithm in VM Cloud Environment", International Journal of Advanced Computer Science and Applications", Volume 3, Issue I, pp 439- 441, 2012, ISSN:0976-8491 [online], ISSN:2229-433[Print]

[8] Shanti Swaroop Moharana, Rajadeepan 0 Ramesh, OigamberPowar, "Analysis of Load Balancers in Cloud Computing", International Journal of Computer Science and Engineering, Volume 2, issue 2,pp 101-108, May 2013, ISSN: 2278-9960

[9] Demystifying_the_cloud_ebook.pdf.

[10] Anthony j Velte, Toby J Velte, Robert Eisenpeter, "Cloud Computing: A Practical Approach", 1st Edition, 2009, Tata McGraw Hill Publishers, ISBN: 0071626948.

[11] https://csrc.nist.gov/publications/nistpubs/800-1

[12] IvonaBrandic, Vincent C. Emeakaroha, Michael Maurer, SchahramOustdar, Sandor Acs, Attila Kertesz, Gabor Kecskemeti,"LA YSI: A Layered Approach for SLA-Violation Propagation in Self-manageable Cloud Infrastructures",34th Annual IEEE Computer Software and Applications Conference, pp 365-360,2010,001 10.1109/COMPSACW.2010.70.

[13] K C Gouda, Radhika T V, Akshatha M, "Priority Based Resource Allocation Model for Cloud Computing", International Journal of Science, engineering and Technology Research, Volume 2, Issue I, pp 215-219, January 2013

[14] Bhavani B H, H S Guruprasad, "Resource Provisioning Techniques in Cloud Computing Environment: A Survey", International Journal of Research in Computer and Communication Technology, pp395-401, Volume 3, Issue 3, March 2015.

[15] John W Rittinghouse, James F Ransome, "Cloud Computing: Implementation, Management and Security", CRC Press, 2009

[16] Jasmin James, Bhupendra Venn a, "Efficient VM Load Balancing Algorithm for a Cloud Computing Environment", International Journal on Computer Science and Engineering, Volume 4,pp 1658-1663, September 2012, ISSN: 0975-3397