Load Balancing during Task Scheduling in Cloud Computing

Sukhman Kaur,Rachhpal Singh
Punjabi University Regional centre for Information Technology and Management
Punjab;India

Abstract—In this technology era cloud computing plays a vital role. For efficient cloud computing it is important that the load on the machine in the network should be closely monitored and balanced in order to produce efficient performance. There are various techniques for load balancing during scheduling. The main focus is on the increase in performance that can be evaluated by efficiency and downtime. The increase in efficiency and decrease in downtime depicts efficient working of cloud network.

Keywords— load balancing, downtime,VM.

I. INTRODUCTION

Cloud computing takes virtual infrastructure and builds upon research in distributed computing, grid computing, utility computing, autonomic computing, networking, web services and software services. It has shown tremendous potential to empowerment, agility, multi-tenancy, reliability, scalability, availability, performance, security and maintenance. Through Cloud environment Email, Instant messaging, business software, and web content management can be offered. It incorporates many existing technologies such as information and infrastructure consisting of pools of computers, networks, distributed services application, information and storage resources.

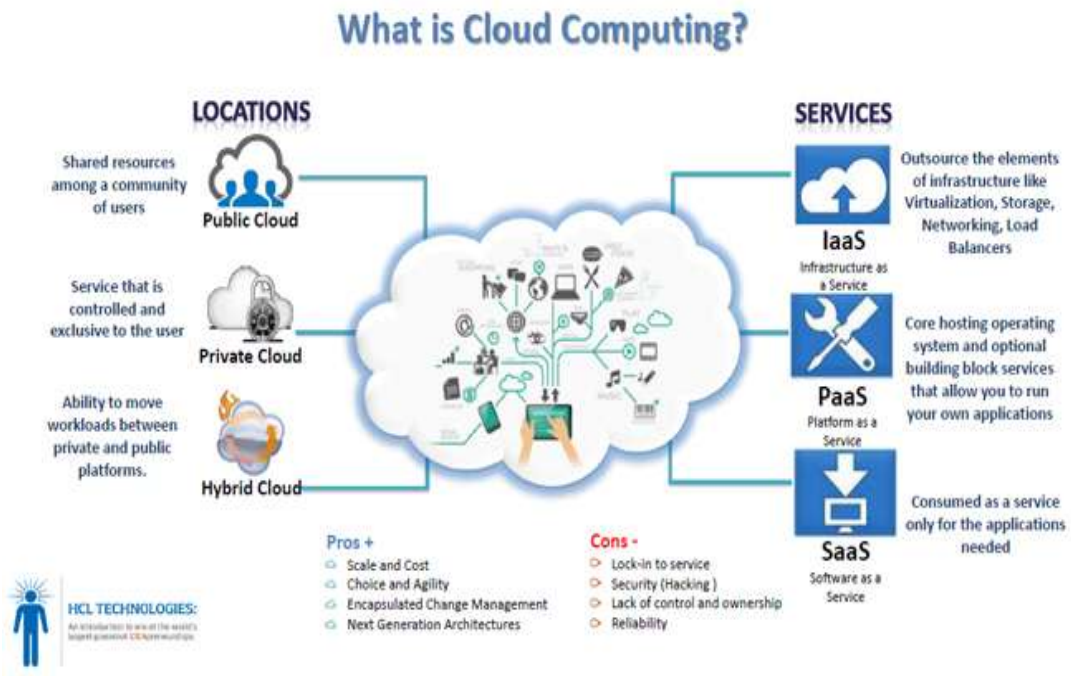

Figure 1: Cloud Computing

SPA is an approach that takes into account the availability of the whole network resource, while guaranteeing load-balancing and SLAs objectives. To achieve this, migrating Virtual DataCenter Networks (VDNs) should clearly specify its detailed resource requirements (i.e. the resource vector) to the hosting physical network. This can provide for optimal placements and satisfying services. In this context, requirements may vary from a virtual network to another, depending on the considered topologies and the provided services. However, among all the network components, the challenge for the hosting CDNs (i.e. the physical ones) mainly lies in the switching capabilities of its network, more precisely, its path processing capacities. Indeed, where for a packet to get processed through a switching device, certain resources are required. In this context, let us define the physical switch as a set of virtual switches, where each virtual switch operates a set of virtual switching paths. Mainly, a virtual switching path to operate requires a set of:

- packet processing resources (network processor cycles, search caches, memories);
- ports;
- bandwidth over the ports.

Typically, for a packet processing task to operate, this requires:

- processors (for parsing and analysis);
- memories (for the lookup tables) that can be either internal or external (e.g. TCAMs, SRAMs);
- queues (for packets' scheduling and storage, and for the process of shaping priorities);
- bandwidth over the busses that interconnect the aforementioned internal components.

Accordingly, such physical resources will be virtually partitioned among the different virtual data paths that are allocated (reserved) to satisfy the requirements of the VDNs topologies. Hence, for efficient allocations and optimal placement decisions, such resource vectors need to be clear in order to check for resource availability at the hosting physical network.

Therefore, to simplify the presentation, the resource requirements of the migrating networks (i.e. the VDNs) will be represented by the virtual data path capacity. At the hosting side (i.e. the hosting CDNs topology), this will be translated to path processing resources, being a parameter for the placement process. Thus, having the resource victor of the migrating VDN, the hosting network administrator can break this down to:

- ports;
- processing engine capabilities;
- memories;
- internal bandwidth; all constrained by certain speed/delay limits for QoS assurance.

Besides, the administrator need to specify the external bandwidth requirements (bandwidth over the links that interconnects the different switching devices), this could be defined by the Network Interface Cards (NICs) capacities.

II. LOAD BALANCING

Load Balancing is to achieve optimal resource utilization, maximize throughput, minimize response time and avoid overload. Dynamic algorithm is self-adaptive algorithm which is better than the static algorithm. This algorithm makes changes to the distribution of load among virtual machines of the cloud at run time. The algorithm always uses recent load information while making distribution decisions. It includes two processes monitoring the load states of the virtual machine. Due to these results dynamic load balancing algorithm provide a significant improvement in performance over static algorithms [1]. Load balancing is a method to strength network data processing capability, increase the flexibility and availability of the network. Using multiple components with load balancing, instead of a single component, to provide external services can solve the problem of insufficient processing capacity of a single host, and can also solve the challenges posed by concurrent access due to the importance of load balancing scheduling, which is the key issue as this a scheduling algorithm based on dynamic policy. Although dynamic load balancing exerts immense stress on a system and each node needs to interchange status information periodically yet it is more advantageous when most of nodes work in individual manner with partial interaction with others.

Load balancing algorithm in a cloud can be classified into two categories. These are:

1. Static Load Balancing.
2. Dynamic Load Balancing.

Static load balancing algorithm is simple in terms of implementation and overheads because there is no need to monitor virtual machines (VMs) for performance statistics. This algorithm takes less time but it does not refer to the states of the servers but there is very serious disadvantage that static algorithm only works well when there is not much variation in the load of VMs. It means where load of virtual machines significantly varies and the cloud resources are heterogeneous then this algorithm fails. Other shortcomings are that this does not depend on the current state of the system. Prior knowledge of system is one of the pre-requirement. It cannot make change to the distribution of work among virtual machines at run time.

Dynamic algorithm is self-adaptive algorithm which is better than the static algorithm. This algorithm makes changes to the distribution of load among virtual machines of the cloud at run time. The algorithm always uses recent load information while making distribution decisions. It includes two processes monitoring the load states of the virtual machine. Due to these results dynamic load balancing algorithm provide a significant improvement in performance over static algorithms [1]. Although dynamic load balancing exerts immense stress on a system and each node needs to interchange status information periodically but it is more advantageous when most of nodes work in individual manner with very few interaction with others. But dynamic load balancing technique is more useful in comparison to static load balancing. Among the entire techniques dynamic load balancing with round robin approach is the best for load balancing.

III. RELATED WORK

AnkushP.Deshmukh and Prof. Kumarswamy Pamu[1] have discussed/described different load balancing strategies, algorithms and methods and by studying pros and cons of different techniques used for load balancing, authors are specifically giving priority to Dynamic load balancing method rather than Static load balancing. They investigate that comparative behavior of load balancing with different parameters; dynamic load balancing is more reliable and after that they conclude that efficient load balancing can clearly provide major performance benefit.

Alam, B. et al.[2] states that in Round Robin Scheduling, the time quantum is fixed and then processes are scheduled such that no processes get CPU time more than one time quantum in one go. If time quantum is too large, the response time of the processes is too much which may not be tolerated in interactive environment. If time quantum is too small, it causes unnecessarily frequent context switch leading to more overheads resulting in less throughput. In this paper, a method using fuzzy logic has been proposed that decides a value that is neither too large nor too small such that every process has got reasonable response time and the throughput of the system is not decreased due to unnecessarily context switches.

Hongchao Hu et al. [7] presented that the limitations in complexities and extensibilities of CICQ switchespsila scheduling policies are first analysed. Then, based on this analysis, the guidelines for designing high extensible scheduling policies and the concept of virtual channel are proposed. Based on the guidelines and virtual channel, it comes up with a dynamic round robin scheduling algorithm-FDR (fair service and dynamic round robin), which is simple, high efficiency and fair service. FDR is based on round robin mechanism and its complexity is of order one(O(1)). It allots the scheduling share for each virtual channel according to its current states. Thus, FDR has good

dynamic and real-time performance, and it can adapt to unbalanced traffic load network environment. Simulation results under SPES show that FDR performs good delay, throughput and anti-burst performance, which can be applied in high performance routing and switching devices.

Jadav, D.Choudhary, A.N. ; Berra, P.B. [12] have discussed for policies for request assignment that with High-performance servers and high-speed networks will form the backbone of the infrastructure required for distributed multimedia information systems. A server for an interactive distributed multimedia system may require thousands of gigabytes of storage space and a high I/O bandwidth. In order to maximize the system utilization, and thus minimize the cost, it is essential that the load be balanced among each of the server's components, viz. the disks, the interconnection network and the scheduler. Many algorithms for maximizing retrieval capacity from the storage system have been proposed in the literature. This paper presents techniques for improving the server capacity by assigning media requests to the nodes of a server so as to balance the load on the interconnection network and the scheduling nodes. Five policies for request assignment-round-robin (RR), minimum link allocation (MLA), minimum contention allocation (MCA), weighted minimum link allocation (WMLA) and weighted minimum contention allocation (WMCA)-are developed. The performance of these policies on a server model developed by the authors (1995) is presented. Authors also consider the issue of file replication, and develop two schemes for storing the replicas: the parent group-based round-robin placement (PGBRRP) scheme, and the group-wide round-robin placement (GWRRP) scheme.

Jaspreet Kaur [11] has discussed an algorithm called active VM load balancer algorithm to find the suitable VM in a short time period. She has stressed to count the maximum length of VM for the allocation of new request. If the length of the VM is not sufficient then a new VM would be added.

Jingnan et al. [13] have presented that with the advent of powerful network processors (NPs) in the market, many computation-intensive tasks such as routing table lookup, classification, IP sec, and multimedia transcoding can now be accomplished more easily in a router. An NP consists of a number of on-chip processors to carry out packet level parallel processing operations, ensuring good load balancing among the processors increases throughput. However, such type of multi-processing also gives rise to increased out-of-order departure of processed packets. This work first proposes an Ordered Round Robin (ORR) scheme to schedule packets in a heterogeneous network processor assuming that the workload is perfectly divisible. The processed loads from the processors are ordered perfectly. This finding analyzes the throughput and derives expressions for the batch size, scheduling time and maximum number of schedulable processors.

Jinhua Hu , JianhuaGu; Guofei Sun ; TianhaiZhao [14] sates that the current virtual machine(VM) resources scheduling in cloud computing environment mainly considers the current state of the system but seldom considers system variation and historical data, which always leads to load imbalance of the system. In view of the load balancing problem in VM resources scheduling, this paper presents a scheduling strategy on load balancing of VM resources based on genetic algorithm. According to historical data and current state of the system and through genetic algorithm, this strategy computes ahead the influence it will have on the system after the deployment of the needed VM resources and then chooses the least-affective solution, through which it achieves the best load balancing and reduces or avoids dynamic migration. This strategy solves the problem of load imbalance and high migration cost by traditional algorithms after scheduling. Experimental results prove that this method is able to realizeload balancing and reasonable resources utilization both when system load is stable and variant.

Milan E. Sokile [16] in this paper, author has studied that the comparison are made between various techniques but static load balancing algorithm are more stable and it is also easy to predict their behavior, but at a same time dynamic distributed algorithm are always considered better than static algorithm. Experimental results of performance modeling show that dynamic load balancing is better than static load balancing in a dynamic environment, which manifest in frequent clients' object creation requests and in short objects' lifetimes.

IV. RESULTS

STEP 1: Virtual machines in idle state. It describes about the main screen that is showing the machines that are in the idle state that is no load is assigned to them. In the proposed scheme the load migrated is done on the basis of parameters like utilization, speed, memory and power where VM is the virtual machine.

STEP 2: Load the machines. It describes about the machines when load is assigned to all the machines. The assigned values described about the load on various machines. When the load is allocated to the various machines continuously and it reaches the threshold value, the load will be migrated from that loaded machine to the other under loaded machine.

STEP 3: Overloaded Machine. It describes about the overloaded machine that is in the red mark. The assumed threshold for the overload condition to occur is above 80%.When the threshold is crossed, the load in the machine is migrated to the optimal destination having less load on it.

STEP 4: According to the priority with respect to the parameters like utilization, memory, speed and power of the virtual machines, the optimal destination is chosen. It describes about the selection of the candidate Virtual Machine that to which the load is to be transferred according to the priority table. In the research a priority table is developed by the algorithm, for the calculation of the destination machine.

STEP 5: Selection of the Virtual Machine that to which the load is to be transferred according to the priority table. The optimal destination is chosen according to the priority with respect to the parameters like utilization, memory, speed and power of the virtual machines. The machine having less load on it and greater speed and better power, the load will be transferred to it. It describes about the selection of the candidate Virtual Machine that to which the load is to be transferred according to the priority table.

STEP 6: Downtime during the load sharing. Downtime is defined as the time at which the virtual machines stop executing. It includes transfer of the processor state. In the proposed approach, the downtime is decreased which results in better performance. The downtime can be calculated by the formula:

Total Downtime = Stop-and-copy + commitment + activation.

STEP 7: Efficiency. In the proposed approach, the efficiency of the system is improved.

Comparison graphs of the work performed:

[1]. Down Time: In the proposed approach, the downtime is decreased which results in better performance.

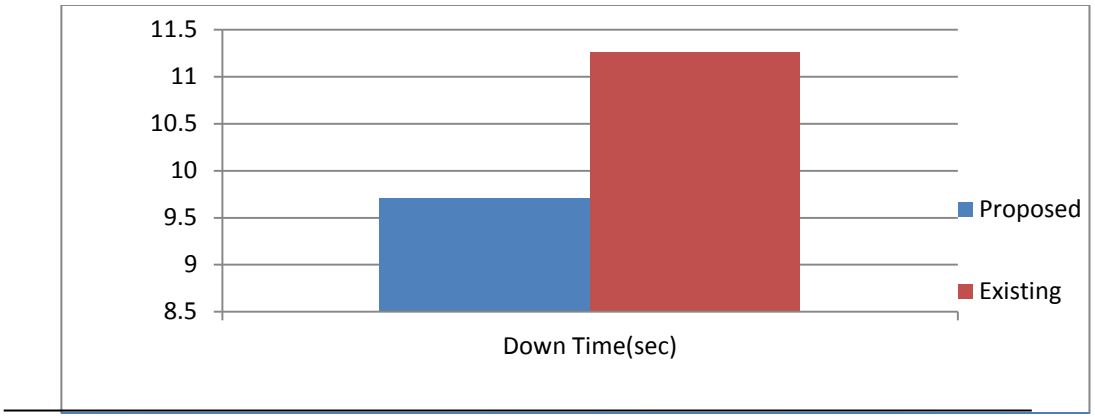

Figure 2: Down time

[2]. Efficiency: In the proposed approach, the efficiency of the system is improved.

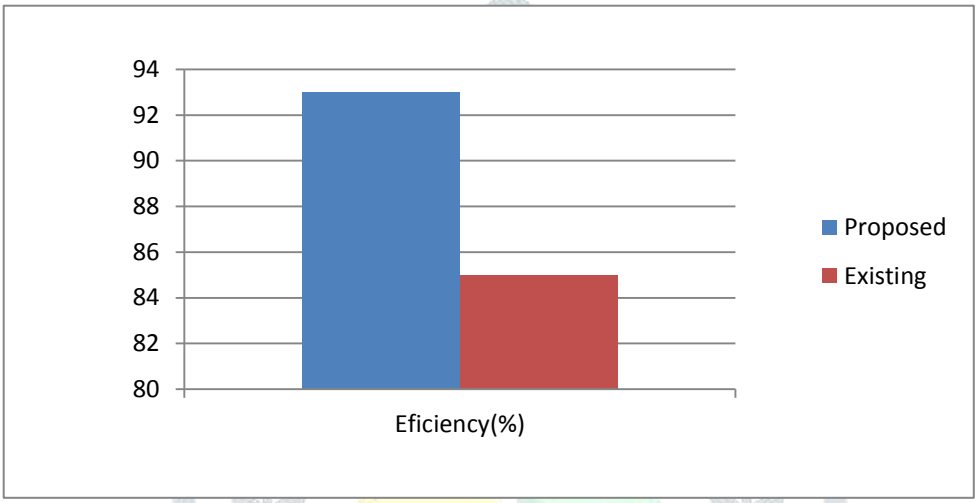

Figure 3: Efficiency

Comparison Table of work performed:

Here the comparison takes place between the base paper and the work performed. The results produced by the work are better than the previous work done.

Approach	Downtime(sec)	Efficieny(%)
Existing	**11.3**	**85**
Proposed	**9.7**	**93**

Table 1: Comparison table of work performed

Vector dot algorithm is used for load balancing in virtual environment. But in this approach, the downtime is very high with lesser efficiency and it presumes the utilization factor only. Downtime is defined as the time at which the virtual machines stop executing. It includes transfer of the processor state. So, we proposed an algorithm that is modified vector dot algorithm in which we assume four parameters to solve the problem of overloaded and underutilized machines by taking different parameters.

Downtime is defined as the time at which the virtual machines stop executing. It includes transfer of the processor state. In the proposed approach, the downtime is decreased which results in better performance.

V. CONCLUSION

This implementation aims towards the establishment of performance qualitative analysis on load sharing in VM to VM and then implemented in CloudSim with Java language. Here major stress is given on the study of load balancing algorithm with heterogeneous resources of the cloud, followed by comparative survey of other algorithms in cloud computing with respect to scalability, homogeneity or

heterogeneity and process migration. A previous study also indicates change of MIPS will affect the response time and increase in MIPS versus VM decreases the response time. When image size of VM is implemented against the VM bandwidth then no significant effect is found on response time and it remains constant for which these parameters are investigated. But in case of Cloudlet long length versus Host bandwidth a pattern is observed in which response time increases in proportionate manner. Using the modified approach the reduction in the down time of the various processes are achieved as shown in results.

VI. REFERENCES

[1] Ankush P. Deshmukh and Prof. Kumarswamy Pamu "Applying Load Balancing: A Dynamic Approach" International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), vol. 2, issue 6, June 2012.

[2] Alam, B. Doja, M.N.; Biswas, R. "Finding Time Quantum of Round Robin CPU Scheduling Algorithm Using Fuzzy Logic", ICCEE 2008, International Conference, December 20-22, 2008.

[3] Ajay Gulati, Ranjeev.K.Chopra "Dynamic Round Robin for Load Balancing in a Cloud Computing" International Journal of Computer Science and Mobile Computing (IJCSMC), vol. 2, issue. 6, June 2013.

[4] BelabbasYagoubi and YahyaSlimani "Dynamic Load Balancing Strategy for Grid Computing" World Academy of Science, Engineering and Technology 19, 2006.

[5] CloudSim: A Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services, The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University of Melbourne, (2011) available from: http://www.cloudbus.org/cloudsim/

[6] Fang Liu, Jin Tong, Dr.Jian Mao, Knowcean Consulting Inc. "NIST Cloud Computing Reference Architecture" version 1, March 30, 2011.

[7] Hongchao Hu , Peng Yi , YunfeiGuo,Hui Li, "A Fair Service and Dynamic Round Robin scheduling scheme for CICQ switches" Telecommunications, ICT 2008, International Conference, 16-19 June 2008.

[8] http://en.wikipedia.org/wiki/File:Cloud_computing.svg

[9] IBM Cloud computing, http://www-07.ibm/in/cloud-computing/.html.

[10] Java software version 1.7 downloaded from: http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html, September 2012.

[11] Jaspreetkaur "Comparison of Load balancing algorithms in a Cloud" International Journal of Engineering Research and Applications (IJERA), vol. 2, issue 3, May-June 2012

[12] Jadav, D. Choudhary, A.N. ; Berra, P.B., "Techniques for increasing the stream capacity of a high-performance multimedia server" Knowledge and Data Engineering, IEEE transactions, vol. 11, issue: 2 Mar/Apr 1999.

[13] Jingnan Yao JianiGuo ;Bhuyan, L.N. , "Ordered Round-Robin: An Efficient Sequence Preserving Packet Scheduler" IEEE transactions, vol. 57, issue 12, 30 May, 2008.

[14] Jinhua Hu; JianhuaGu; Guofei Sun; Tianhai Zhao, "A Scheduling Strategy on LoadBalancing of Virtual Machine Resources in Cloud Computing Environment" Parallel Architectures, Algorithms and Programming (PAAP), 2010 Third International Symposium, 8-20 Dec. 2010.

[15] Meenakshi Sharma, Pankaj Sharma, Dr.Sandeep Sharma, "Efficient Load Balancing Algorithm in VM Cloud Environment" International Journal of Computer Science and Technology (IJCST), vol. 3, issue 1, Jan-March 2012.

[16] Milan E. Soklic "Simulation of Load balancing algorithms" ACM - SIGCSE Bulletin, December, 2002.

[17] Rangarajan, S.; Garcia-Luna-Aceves, J.J. "Load-Balanced Routing in Ad hoc Networks" Computer Communications and Networks, 2007. ICCCN 2007, Proc. of Sixteenth International Conference, 13-16 Aug. 2007.