Dynamic Weighted Virtual Machine Live Migration Mechanism to Manages Load Balancing in Cloud Computing

Pradeep Kumar Tiwari
Dept. of Computer Science and Engineering
Manipal University Jaipur
Jaipur, India
pradeeptiwari.mca@gmail.com

Sandeep Joshi
Dept. of Computer Science and Engineering
Manipal University Jaipur
Jaipur, India
sandeep.joshi@jaipur.manipal.edu

***Abstract*— Load Balancing is a mechanism of efficient utilization of computing resources (i.e. CPU, Memory and Network). Cloud computing is the best option to reduce CAPEX and OPEX of the organizations. High Quality of Service (QoS) and reduction the Service Level Agreement (SLA) violation depend on efficient management of Load Imbalance problem. Cloud Service efficiency depends on proper management of physical and logical resources. Researchers did lots of work on the effective load management system. Researchers used many approaches (i.e. Ant colony, Honey Bee, Genetic, Static, and Dynamic) but still need improvement. Virtual Machines (VMs) migration system plays the vital role in load management. Our focus in this research is maximizing the utilization of VMs CPU capacity.**

Proposed research approach is based on Dynamic Weighted Live Migration (DWLM) to manage load imbalance problem. The proposed mechanism results outcomes compared with Migration time, Scalability, Throughput and Availability factor from Equally Spread Current Execution load balancing algorithm (ESCEL) and Push Pull algorithm. This paper also focuses on others load balancing strategies and future research scope in Load management mechanism.

Keywords—: Load Balancing, Virtual Machine, Cloud Computing, Load Mnagement, User Base, Data Center

I. INTRODUCTION

Load balancing enhances the workload distribution among the multiple physical resources. Hyper threading mechanism can split the single processor as multiple processors to maximize the utilization of CPU. Load balancing intends to minimize the resource utilization in minimum availability of resources. Our aim to maximize the throughput and minimizes the response time with fault tolerance [1].

Cloud computing is the ability of using various computing resources via the internet. These resources are divided in physical resource (i.e. CPU, Memory, Storage, Work Station) and logical resource (i.e. Energy, Network, Throughput, Load Balancing Mechanism). The shared pool of the resources is managed by resource manager. Resource manager have the accounting information of resources. Resource manger stores the availability status, utilization status of resources and management of resources among the VMs [2].

Our work has analysis of researchers proposed work. We developed and implemented the Dynamic Weighted Live Migration (DWLM) algorithm using Java Programming on Cloudanalyst. Proposed mechanism result shows the best result from others. Our research focuses on Migration time, Throughput, Scalability, and Fault tolerance. Graphical and tabular representation shows the efficiency of proposed work.
.

II. LETRATURE REVIEW AND WORK ANALYSIS

Genetic Approach-- CT Joseph et. al.[3] proposed a Genetic Approach for VM Allocations. This approach focused on minimization of energy consumption and number of VM migrations. The proposed algorithms are divided in three sub-algorithm, first algorithm is based on the Family Genetic algorithm, second algorithm work on the fitness value of each chromosome and the third algorithm takes as input an individual and returns a chromosome representing a feasible assignment.

Adaptive Approach. - G Kanagaraj. et al.[4] suggest a method to dynamically load balance using service queue where in every server computes load value by summing the load parameters. Mechanism ensures the memory, CPU, network utilization and also exchange load value with a central node in a certain cyclic period. Central node selects the least loaded server among available servers to process the request. Each server at the central node waits in a queue. This queue is called service queue and server wait for turn for requested process. Researcher's categories three threshold values – high, low ,normal and scheduler check the memory, CPU and network traffic load in every five second.

A Kopaneli. et. al. [5] proposed the adaptive cloud target selection approach which is near to the real work simulation environment. *KY Kabalan. et. al.*[6] proposed Adaptive Load Sharing with Never Queue Policy. Researchers use heterogeneous computing System and simulate the work on the modified SED, NQ and GT policies.

Agent Base Approach,- *J Cao, et. al.* [7] proposed Agent-Based Grid Management mechanism. This work based on performance-driven task scheduler with maximum utilization of resource. Researchers use PACE resource tool to simulate the proposed work and they also recommend the other simulation tool (e.g.. Globus MDS and NWS) to check the

978-1-5090-0612-0/16/$31.00 ©2016 IEEE

efficiency of the proposed mechanism. *A Singh . et. al.*[8] proposed Autonomous Agent Based Dynamic Load Balancing mechanism. The authors' main goal was maximize the resource utilization

Table 1. Review Analysis on Load Balancing Mechanism, Strength, Focus area and used tools

Author(s)	Technique	Strength	Focus Area	Used Tools	Recommendations
Cao J. et. al.,,[7]	Agent-based grid management	Performance-driven task scheduler	Resource Utilization	PACE resource tool	Use of another grid tool e.g.. Globus MDS and NWS
Singh A. et. al.,[8]	Autonomous Agent Based	Dynamic Load Balancing	Maximum resource utilization, maximum throughput, minimum response time	Clousim	User Can use Heuristic approach
Ferreto TC. et. al.,[9]	LP formulation and heuristic	Server consolidation with migration control	Energy Efficient	Zimpl language using for LP formulation and Python using for Heuristic approach performed with Inter Core 2 Deo with 2.4 Ghz and 4 GB primary memory.	Migration control without downtime
Forsman M. et. al.,[10]	Push and Pull Strategy	Rebalance the load when VMs added and removed	Load Management	OMNeT++	Downtime can be Less.
Joseph CT. et. al.,[11]	Genetic Approach	SLA based Resource Management, Maximize the hardware resource Utilization with performance	Resource Management	Cloudsim	Modify the algorithm to decrease the calculation time in terms of prediction process to improve the Genetic algorithm convergence speed.

.

III. DYNAMIC WEIGHTED LIVE MIGRATION STRATEGIES

A proposed strategy is working on Dynamic Weighted Live Migration (DWLM). To check the operating system availability can be calculate by u_t = uptime and d_t= downtime. OS_A = Operating system availability. Availability percentage calculated by,

$$OS\_A = \left(\frac{u\_t}{u\_t + d\_t}\right) * 100 \quad (1)$$

CPU utilization map by 0.0 to 1.0. 0.0 define there is no job allocated yet and 0.3 define only 30 percent CPU utilized remaining is free. High load can be check by threshold value.

.

Threshold value map the utilization level of CPU and it also help to find high load and low load machine. Algorithms 2 present the threshold model.

This entropy system formulated as $\boldsymbol{l_i} \geq 0$, load of all ith DataCenter with n Data Centers. The normalize load will be

$$\mathbf{0} \leq \boldsymbol{p_i} \leq \mathbf{1}$$

$$\boldsymbol{p_i} = \frac{\boldsymbol{l_i}}{\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}} \boldsymbol{l_i}} \quad (2)$$

Here P= $\{p_1,p_2,p_3,\ldots..p_n\}$ be the set of normalized value.

The entropy defined equation 3.

$$\boldsymbol{H(p) = \sum_{i=1}^{n} p_{i\,log\,p_i}} \quad (3)$$

The maximum entropy value is $\boldsymbol{H(p)_{max} = log\ n}$

Correspond to the case $E_{(p)}$ find by $H_{(p)}$ and $H_{(p)max}$. The normalize entropy metrics equation define by equation 5. This equation also used by forsman et. al.[10], and Qin et. al.[13]

$$p_i = {}^{1}/_{n} \quad (4)$$

E(P) as the normalized entropy matrices

$$\boldsymbol{E(p) = \frac{H(p)}{H(p)_{max}} = \frac{\sum_{i=1}^{n} p_{i\,log\,p_i}}{log\frac{1}{n}}} \quad (5)$$

Entropy mechanism proposed by Forsman et. al. [10] and Lui et. al.[14] . Our DWLM mechanism also used these policies

Algorithm 1: DWLM Algorithm

```
1: if Event then
2: if Event = hotspot then
3: for all physical machines do
4: Send information about resource usage and information about all virtual machines
/*Identify the number of executing/pending tasks in each VM and arrange it in increasing order on a Queue */
// method- getNextAvailableVm()
5: end for /* Waiting for allocations*/
6: if PMs >0
7. for all virtual machines do
8.VM_Id==Available (State)
9. If the number of allocation in the list item of the queue is smaller than to VM of list ,
Then
10. Check the availability of VM. vmStatesList.size() > 0 /* VM id of the first available VM from the VM States List in the calling . */
Return vmId.
10: Receive VMs
11. Check the availability of VM. vmStatesList.size() > 0 /* VM id of the first available VM from the VM States List in the calling . */
Return vmId.
12: Receive VMs
13: end for // request allocated on VMs
14: for all VMs (allocations) do
15: Calculate the utility for all VM Candidates and append to utility Array
16: end for
17: for all utilities in utility Array do
18: migrated and Calculate migration time
19: end for
20: Sort utility Array based on utility
21: waiting candidates that take high time for migration, according to CPU utilization, MIPS and process element (PE).
22: Set all process to value in utility Array
23: for all physical machines do
24: Send information about acknowledgement (completion process)
25: end for
26: Listen for a confirmation message from the winning physical machine
27: Get response about total number of process, virtual machine, host, datacenter Success or fail.
28: Calculate response time / average response time, throughput, scalability, reliability
29: exit
30: else
31: Run migration and exit
32: end if
```

Algorithm 2: Threshold Model

```
1: if Event then
2: if Event = hotspot then
3: for all physical machines do
4: Migration and Utilization calculate to threshold value //utilization Threshold = 0.8
5: if compare to SLA violation value
6: Allocated process on VM
7: end if
8: Run backoff algorithm and exit
9: else
10: Run migration and exit
11: end for
12. end if
```

DWLM algorithm checks the VMs availability and available jobs in queue which are not allocated in VMs due to overloaded condition of VMs. Migrated jobs time duration calculated by migration time. CPU capacity map by MIPS (Million Instructions per Seconds). MIPS capacity maps by 0.0 to 1.0 our proposed mechanism highest utilization is 0.8. Migration and Utilization part depend on threshold value. We have a system as in Assuming that a physical machine's load never goes beyond 0.8. If the system will continue to run in an imbalanced state then we have applied the variable threshold. As time goes on the threshold will continuously decrease. An imbalanced system that is unable to perform migrations due to the physical machines' CPU load not reaching the threshold. This situation is more critical because the services provided to the customers might fall below acceptable levels and this might result in a violation of the SLA.

IV Result Analysis Report and Graph

Outcome of proposed algorithm is based on 38 User Base (UB). These User Base request are allocated in 5 Data Centers (DC). Each DC has 5 VMs. We used X-86 processor architecture with Xen Virtual Machine Monitor(VMM) hypervisor system used by with Linux operating system. Developed and implemented the Dynamic Weighted Live Migration (DWLM) algorithm using Java Programming and simulate the algorithms on Cloudanalyst[15]. Proposed algorithm compared with two different algorithms (Equally Spread current execution load algorithm (ESCEL) and Push Pull algorithm). Result analysis is based on Migration time, Throughput, Scalability of load, and fault tolerance. Table 2 shows the Migration Time in millisecond, Throughput, scalability and fault tolerance.

Table 2 shows the comparative analysis among the proposed DWLM mechanism Push PULL, and ESCEL. Proposed mechanism minimizes the migration time and maximizes the throughput. Mechanism effectively achieved high scalability, availability and reliability features of load balancing matrices [16].

Figure 1 screen shot shows the column chart of Scalability, Availability and Reliability of ESCE, Push Pull and DWLM

Chart shows the proposed algorithm is best among compared algorithm. Chart shows DWLM management best in CPU utilization.

Table 2 : DWLM, ESCEL and Push Pull algorithm with Migration Time in millisecond, Throughput, scalability and fault tolerance.

Result Analysis Table					
Algorithm(s)	Migration Time	Throughput	Scalability	Availability	Reliability
DWLM	43.00279114633795	41.83684919568717	66.868983724960200	87.174867881916300	88.700076514668
Push Pull	60.92049288738691	28.607421181056570	45.724026023603000	59.608890483990400	60.651805679206100
E S C E L	57.94009811898519	28.955908233710660	46.281022439448960	60.335028167919950	61.390647894164730

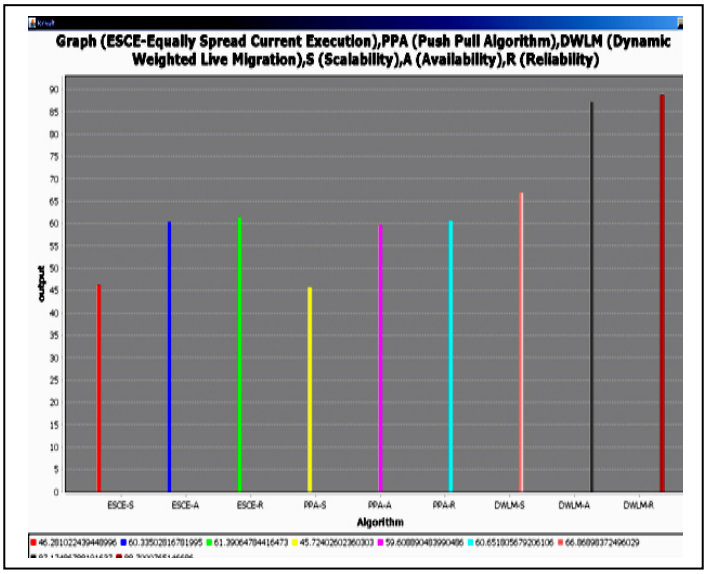

Fig. 1. Graph result shows the result of table 2. DWLM mechanism maximizes the scalability of VMs and resources, availability of VMs and reliability of system.

Conclusion

Load balance manager policy manages the physical and logical resources. UB Jobs request approach to nearby VMs. VMs executed the UBs requested jobs. Load Imbalance management policy manage by threshold model. Threshold value helps to separate the high and low load VMs. DWLM manager manages the load and finds the VMs and migrate the jobs with the help of threshold model. Mechanism effectively allocates and reallocate of VMs. The result shows that proposed DWLM is the best option to load management. The

algorithm divide in two parts, first algorithm are responsible for managing the VMs, migration of VMs and management of UB request in VMs. And second Algorithm the threshold of VMs to check overload condition. Algorithm is simulated in Cloudanalyst simulator. The result may be differences in real environment. The researcher can be used the real environment to implement the proposed algorithm

REFRENCES

[1] P. K Tiwari and S. Joshi , "A Review on Load Balancing of Virtual Machine Resources in Cloud Computing," In Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 2 ,pp. 369-378,. 2016 Springer International Publishing.

[2] J. García-Galán, P Trinidad , Rana OF, A. Ruiz-Cortés, "Automated configuration support for infrastructure migration to the cloud," Future Generation Computer Systems pp 200-212, Feb-2016

[3] CT Joseph, K . Chandrasekaran, R. Cyriac., "A novel family genetic approach for virtual machine allocation," Procedia Computer Science.pp 558-565, Dec-2015.

[4] G. Kanagaraj, V. Shanmugasundaram, S Prakash, "Adaptive Load Balancing Algorithm Using Service Queue," In2nd International Conference on Computer Science and Information Technology Singapore (ICCSIT') . pp 28-29, 2012.

[5] 5 A. Kopaneli, G. Kousiouris, GE. Velez, A. Evangelinou, T. Varvarigou, "A Model Driven Approach for Supporting the Cloud Target Selection" .Process. Procedia Computer Science. pp. 89-102, Dec- 2015

[6] KY Kabalan, WW Smari, JY Hakimian, "Adaptive load sharing in heterogeneous systems," Policies, modifications, and simulation. International Journal of Simulation, Systems, Science and Technology. Pp. 89-100, June 2002.

[7] .J. Cao, DP Spooner, SA Jarvis, S. Saini, GR .Nudd, "Agent-based grid load balancing using performance-driven task scheduling,' InParallel and Distributed Processing Symposium, roceedings. International IEEE .2003.

[8] A .Singh, D .Juneja, M. Malhotra, "Autonomous agent based load balancing algorithm in cloud computing," Procedia Computer Science pp. 832-841, Dec 2015.

[9] TC Ferreto, MA Netto, RN Calheiros, CA De Rose," Server consolidation with migration control for virtualized data centers," Future Generation Computer Systems, pp 1027-1034, Oct 2011.

[10] M Forsman, A Glad, L Lundberg, D Ilie.,"Algorithms for automated live migration of virtual machines," Journal of Systems and Software, pp.110-126, Marh 2015.

[11] CT Joseph, K Chandrasekaran, R Cyriac," A novel family genetic approach for virtual machine allocation," Procedia Computer Science. pp.558-565, Dec 2015.

[12] G Kanagaraj, N Shanmugasundaram, S Prakash," Adaptive Load Balancing Algorithm Using Service Queue,". In2nd International Conference on Computer Science and Information Technology (ICCSIT') Singapore pp 28-29, April 2012.

[13] X Qin, W Zhang, W Wang, J Wei, X Zhao, T Huang," Towards a cost-aware data migration approach for key-value stores," In2012 IEEE International Conference on Cluster Computing, pp. 551-556, Sep 2012

[14] H Liu, H Jin, CZ Xu, X Liao," Performance and energy modeling for live migration of virtual machines", Cluster computing". pp. 249-264, June 2013.

[15] B Wickremasinghe, RN Calheiros, R Buyya, " Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications." In2010, 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 446-452, April 2010 . IEEE.

[16] P. K. Tiwari, and S. Joshi, "Resource Management Using Virtual Machine Migrations". In Proceedings of the International Congress on Information and Communication Technology (pp. 1-10). Springer Singapore.