International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 06 | June -2017 www.irjet.net p-ISSN: 2395-0072

# An Enhanced Throttled Load Balancing Approach for Cloud Environment

**Er. Imtiyaz Ahmad [1], Er. Shakeel Ahmad[2], Er. Sourav Mirdha[3]**

*[1,2]M.Tech. Student, Computer Science & Engineering, International Institute of Engineering & Technology, Samani, Kurukshetra, Haryana, India*

*[3]Assistant Professor, Computer Science & Engineering, International Institute of Engineering & Technology, Samani, Kurukshetra, Haryana, India*

---

**Abstract -** *Cloud computing is advancing with a rapid speed. Already, it has been adopted by a huge set of users. Easy to use and anywhere access like potential of cloud computing has made it more attractive relative to other technologies. This has resulted in reduction of deployment cost on user side. It has also allowed the big companies to lease their infrastructure to recover the installation cost for the organization. Roots of cloud computing have extended from Grid computing. Along with the inherited characteristics of its predecessor technologies it has also adopted the loopholes present in those technologies. Some of the loopholes are identified and corrected recently, but still some are yet to be rectified. Two major areas where still scope of improvement exists are security and performance. The proposed work is devoted to performance enhancement for the user of the existing cloud system by improving the basic throttled mapping approach between task and resources. The enhanced approach has been checked using the cloud analyst simulator. The results are compared with the original and it has been found that proposed work is one step ahead of existing techniques.*

***Key Words***: **Cloud Computing, Load Balancing, Cloud Analyst, Throttled, Round Robin**

## 1. INTRODUCTION

Now days cloud computing environment has been served in almost each and every field in our day to day life. It is one of the internet based services which allows its users to access a number of resources as per their demands. Cloud is a pool of shared resources of information, software, databases and many more devices according to client's requests. Cloud computing environment can be distributed, inherited in nature to serve the requests of a number of users in various scenarios. The distributed architecture organizes its resources in a distributed manner to serve its services to users present at various geographical locations in the world. In these kinds of environments, the user generates random requests for any resource. So there is a major drawback of task assignment among various users which leads to load imbalance, i.e. at any particular moment of time some processors may be overloaded and some might be under-loaded [1].

To overcome this drawback we need load balancing. Load balancing assures that there is approximately an equivalent quantity of work among all processors at any particular moment of time. It decides at any instance of time to which virtual machine will be allocated or which client will be put on hold [2].

The rest of this paper has organized as follows: Section II discusses the basics of load balancing. Simulator introduction has discussed in section III. Related work has discussed in section IV. Improved throttled load balancing approach is given in section V. Results and analysis have been given in section VI. Conclusion and enhancement scope is given in section VII.

## 2. LOAD BALANCING BASICS

The objective of load balancing is to optimum resource utilization and satisfaction of service level agreement. Resources can be hardware or software. So, to achieve the set target, there is a need for load balancing algorithm to be implemented in scheduler level [3].

### 2.1 Load Balancing Types

Depending upon the factors algorithm has considered, while mapping, there can be following three types of load balancing algorithms [4, 5]:

- Static Approach: It suits to the homogeneous and static environment. It is configured at design time. After configuration, it can't be changed. It is simple in implementation.
- Dynamic Approach: It is ideal for changing environment & hard to implement. During mapping, it considers the size of the task and the capacity of VM.
- Adaptive approach: This approach is a more advanced version of a dynamic approach. During mapping, it not only considers the parameters of the VM and task, but it changes its approach as per requirement. So it is best suited to cloud environments.

### 2.2 Load Balancing Metrics

The performance of a load balancing technique can be examined on the following parameters [6]:

- Response time: It is the time taken by a load balancing technique to respond.
- Cost: It is composed of storage cost, processing cost & migration cost.
- Throughput: Number of tasks finished per unit of time.
- Scalability: How efficient algorithm is in handling the variety of tasks.
- Fault tolerance: How efficient algorithm is in handling the situation of failure.

## 3. CLOUD ANALYST SIMULATOR

To Cloud analyst [7], a graphical user interface based cloud simulator, which was given by given by B. Wickremasinghe et al. It was an extension of the Cloudsim which was a command based cloud simulator. The structure of cloud analyst is shown in figure 1.

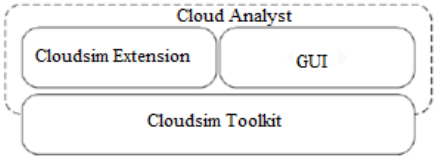


**Fig-1**: Cloud analyst structure [7]

**Fig-2:** Cloud analyst GUI [7]

Major components of the cloud analyst simulator are as follows:

- GUI Package: It is responsible for generating the interface which helps the user to configure the environment easily. Figure 2 shows the GUI-based simulation configuration environment of cloud analyst.
- User Base: This component is responsible for generating the virtual traffic from various geographical locations as per simulation configuration.
- Simulation: This component manages the simulation parameter configuration.
- Internet: This component simulates the internet.
- Internet Characteristics: This component manages the internet characteristics related to simulation configuration.
- Cloud Application Service Broker: This component simulates the traffic routing between user base and data center.
- Data Center Controller: This component manages the tasks of the data center.
- VM Load Balancer: This component simulates the different load balancing policies configured by the user.

## 4. RELATED WORK

In distributed computing, load management is needed to allocate the dynamic native work equally among all the machines. It assists in attaining a great client fulfillment and resource consumption ratio by guaranteeing an effective and impartial distribution of every computing resource. Accurate load balancing helps in reducing resource usage, minimizing failure, allowing scalability and dodging jams, etc. In this section, a systematic analysis of current load balancing methods is discussed. This analysis accomplishes that all the current methods, chiefly emphasis on decreasing related overhead, service reaction time and refining performance, etc. A number of parameters are also recognized, & these are utilized to analyze the existing techniques.

- N. Swarnkar et al. in [8] have discussed round robin approach for load balancing. Data Center Controller allocates leading task to randomly chosen virtual machine from the group of available virtual machines. Then, it allocates tasks to the virtual machines in rounded direction. When a task allocated to a virtual machine, then it moves to the last of the available list of VM. The benefit of this algorithm is that it does not need inter-process communication. Due to non-consideration of task size before mapping, there are chances that several nodes may get overloaded.
- S. Swaroop et al. in [9] have discussed Weighted Round Robin algorithm for load balancing in a cloud environment. Based on the capacity of VM weight is assigned to the VM. VM is selected on the basis of its weight and after VM selection traditional round robin approach is followed.
- K. Mahajan et al. in [10] have proposed another variation of round robin approach named Round Robin with Server Affinity. In this approach information about the previous task allocated to the

VM is stored. It maintains two data structures. Their details are as follows:

1) Hash map: It stores the information about last task VM mapping.

2) VM state list: It stores the status of each VM (Idle/Busy)

Whenever a task is received at the data center, scheduler first checks the Hash map. If its entry exists in Hash map table and a VM is idle, then instead of calling the round robin algorithm, it directly assigns the task without executing the round robin approach. This will improve the response time.

- A. Makroo et al. in [11] have discussed Throttled load balancing approach for cloud environments. In its basic version, a VM can accommodate only one task at a time. Whenever a task arrives at the data center, the scheduler assigns the task to the idle VM. If more than one VM's are idle, then anyone can be selected randomly.
- S.G. Donamal in [12] has discussed the Modified Throttled algorithm for load balancing in a cloud environment. It does the task VM mapping intelligently. An index table maintains the VM list with their state. In traditional throttled approach, for task-VM mapping, it always starts from the beginning of the table. But in this modified approach, instead of starting from the beginning, it started searching from VM next to already assigned VM. This little modification has improved the response time significantly.
- Meenakshi Sharma et al. in [13] have discussed active VM load balancing approach. In this approach, information about the load assigned to each node is maintained & whenever a new task arrives, it is mapped with the least loaded machine. Due to the consideration of VM present load during mapping, it is somewhat dynamic in nature.
- Meenakshi Sharma et al. in [14] have given a new approach for load balancing algorithm in a cloud environment. The algorithm maintains the expected response time corresponding to each VM. It is estimated on the basis of load assigned to each VM. Whenever a new request arrives, it maps it with that VM whose response time is least.
- Jasmin James et al. in [15] have analyzed weighted active monitoring approach for load balancing in a cloud environment. It was the improved version of basic active VM load balancing approach. During task VM mapping, it not only considers the capacity of VM but also considers the no of tasks already assigned to the VM. On arrival of the task, scheduler maps the task with that VM whose power is highest and weight is least.
- Another modified version of active monitoring load balancing algorithm was given by M. M. Ladani et al. in [16]. This strategy maps the task with the highest power available VM. Weight count is assigned to every VM based upon the allocated resources. Higher weight count and availability is the selection criteria.
- M. Vaidehi et al. in [17] have given an idea to balance the load among the nodes of the data center. It does not do the load balancing my task VM mapping. Rather, it does the load balancing by migrating task from overloaded node to under-loaded node. For this purpose, it stores the information related to VM id, task id and no of active tasks allocated to the VM. It continuously watches the status of VM & whenever it found any VM is overloaded it shifts the tasks to under loaded machines.

## 5. PROPOSED WORK

In the proposed approach, a centralized scheduler, named Improved Throttled is used. In the traditional throttled approach, only one task can be assigned to a VM and there is no selection criterion of VM. Any VM which is idle can be assigned the task. It does not consider the task size and VM capacity. So mapping was not good.

But in the proposed approach, i.e. Improved Throttled approach, priority is assigned to each VM. Priority is calculated based on the capacity of VM and active allocated task count and size. The improved throttled scheduler will select that VM whose priority is highest among the available set of VM. A priority threshold level is also set to avoid overloading. If the priority of VM is less than priority threshold level, then the task is not allocated to that VM. Also, the scheduler will start searching VM in VM allocation table from the next to the last allocated VM. This will maintain the randomness in task VM mapping.

The scheduler will maintain VM allocation table which will store VM id, VM capacity, Active task count, Status and Priority of VM.

$P_i$= Priority of i-th task

$$P_i = \frac{\sum_{i=1}^{n} C_i}{(m * \sum_{i=1}^{m} T_i)} \quad (1)$$

$C_i$= Capacity of i-th VM
$T_i$= Size of i-th Task
H= Threshold value
n= Total no of virtual machines
m= Total no of tasks allocated to virtual machine
H(t)= Least priority among the available set of VM at time t.

$$\text{Status } (S_i) = \begin{cases} idle & Pi > H(t) \\ Busy & Pi < H(t) \end{cases} \quad (2)$$

*Algorithm Improved Throttled ( )*
{

*V= {$V_1$, $V_2$, .....$V_n$} //Set of available virtual machines*
*For (i=1 to n)*
*{*
*$C_i$= Capacity($V_i$);*
*// Capacity of all VM in the beginning*
*$P_i$=$C_i$;*
*// Priority of VM in the beginning*
*$S_i$=Idle;*
*// Status of VM in the beginning*
*}*
*j=Random() mod n;*
*// j holds the index of last allocated VM*
*H(t)= Least ($P_i$) at time t;*
*While (Task is there in the Improved Throttled Scheduler Queue)*
*{*
*Pick the current task from the task queue of the scheduler;*
*i=j+1;*
*while (i≠j)*
*{*
*If( ($P_i$<$P_{(i+1)}$) and ($P_{i+1}$>H(t)))*
*{*
*K=i+1;*
*}*
*Else*
*{*
*K=i;*
*}*
*i=(i+1 )mod n*
*}*
*j=K;*
*Assign the received task to j-th VM.*

$$P_j = \frac{C_j}{(m*\sum_{i=1}^{m} T_i)}$$

*If (any VM i has completed the task)*

$$P_i = \frac{C_i}{(m*\sum_{i=1}^{m} T_i)}$$

*Update H($t_i$)= Least ($P_i$) at the time $t_i$*
*For (i=1 to n)*
*{*
*If ($P_i$>H($t_i$)) then*
*$S_i$=idle;*
*Else*
*$S_i$=busy;*
*}}}*

| Throttled | Improved Throttled |
|---|---|
| No provision of task queue at VM level. | The queue of tasks is maintained at the VM level |
| The VM can have only one task at a time. | The VM can have more than one task. It depends upon the priority of VM and threshold value. |
| VM searching procedure always starts from the start of VM allocation table. | It starts from the next to the VM which has received the last tasks. |
| No selection criteria for VM. | Selection of VM is done on the basis of priority of VM |

**Table 1:** Comparison of Throttled and Improved Throttled

## 6. RESULTS & ANALYSIS

### 6.1 Simulation configuration

**`Simulation Duration:** 5 Hours

**Cost Model:**

- Virtual machine usage cost per hour = $ 1
- Memory / second = $ .5
- Data storage / second = $ 0.25
- Data transfer cost / 1GB = $ 0.125

**Data Center hardware configuration:**

- No of processors on physical machine= 3
- Processing power= 100 MIPS
- Storage devices= 50 GB
- Memory= 2 GB
- Internal bandwidth= 1000 MBPS

**Resource scheduler policy on VM:** Time-shared

**Service Broker Policy:** Optimize Response Time

**Grouping Model:**

- Number of parallel users from a single user base= 100
- No of simultaneous demands a different application server instance can bear= 25
- Size of executable instruction per request= 100 bytes

**Data Center virtual machine specification:**

- RAM = 256 MB
- Storage quota= 3 GB
- Architecture = x86
- Operating system = Linux
- Virtualization technique = VMware
- Bandwidth = 250 MBPS

**User Base Specification:**

**Table 2:** User base specification [18]

| User Base | Region | No of requests per user per hour | Data Size Per requests In bytes | Peak hour start time | Peak hour end Time |
|---|---|---|---|---|---|
| UB1 | 0 | 50 | 50 | 19 | 21 |
| UB2 | 1 | 45 | 75 | 14 | 16 |

| | | | | | |
|---|---|---|---|---|---|
| UB3 | 2 | 60 | 100 | 3 | 5 |
| UB4 | 3 | 75 | 90 | 18 | 20 |
| UB5 | 4 | 35 | 65 | 11 | 13 |
| UB6 | 5 | 30 | 70 | 17 | 19 |

### 6.2 Results

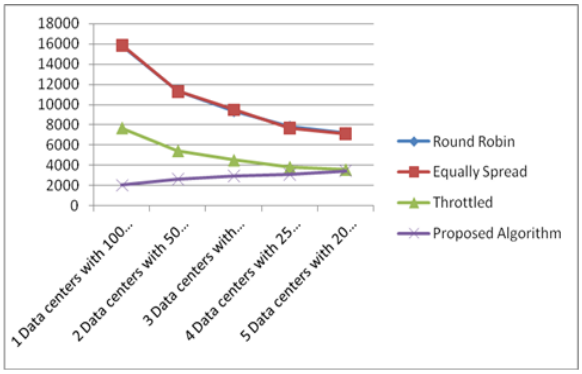


**Chart-1:** Response time analysis on different scenario

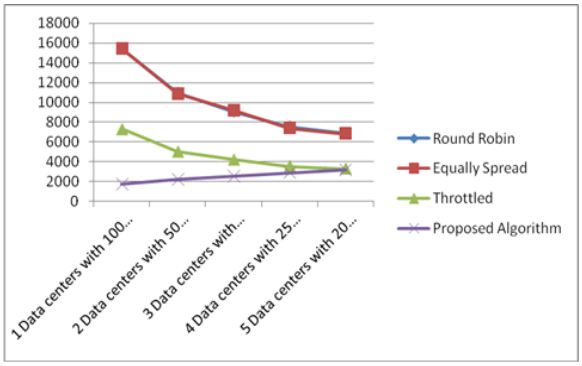


**Chart-2:** Data Processing time analysis on different scenario

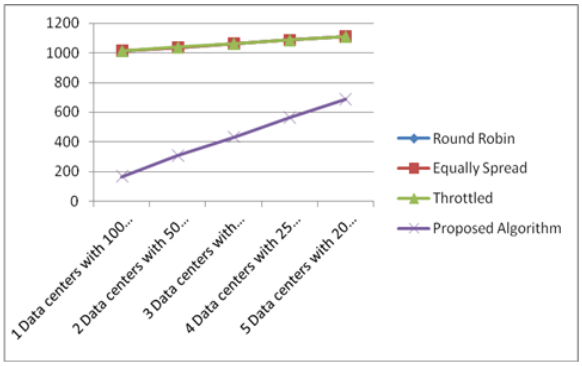


**Chart-3:** Cost analysis on different scenario

### 6.3 Analysis

Chart 1, 2 and 3 shows the performance of the proposed algorithm (improved throttled) in comparison to another centralized algorithm on response time, data processing time and cost parameter. It has been found that the proposed algorithm outperforms the other on all the relevant parameters

## 7. CONCLUSIONS

An Improved throttle algorithm has shown the improved results as compared to the other three i.e. round robin, throttled and equally spread the current execution load. The throttled is known best load balancing strategy among these. As the result of the comparison, it is clear that the performance for the improved throttled is better than the existing load balancing strategies. When it is compared by user bases the individual result may vary according the favorable situations. The values attained overall by the proposed algorithm are best in terms of average values. The values attained by these parameters lacks only in terms of Maximum time in some cases. Overall performance, of the improved throttled algorithm is better than previous best performing algorithm i.e. throttled.
The improved throttled approach is centralized in nature. By combining this with some other approach so that resultant becomes distributed in nature will result in a more suitable approach for cloud environments.

## REFERENCES

[1] A. Jain and R. Kumar, “A Taxonomy of Cloud Computing,” International Journal of Scientific and Research Publications. vol. 4(7), Jul. 2014, pp. 1-5.

[2] P. Sasikala, “Cloud computing: Present Status and Future Implications,” International Journal of Cloud Computing, vol. 1, no. 1, 2011, pp. 23-36.

[3] A. Jain and R. Kumar, "A multi stage load balancing technique for cloud environment," 2016 International Conference on Information Communication and Embedded Systems (ICICES), Chennai, 2016, pp. 1-7.doi: 10.1109/ICICES.2016.7518921.

[4] A. Khiyaita, M. Zbakh, H. El Bakkali, and D. El Kettani, “Load balancing Cloud Computing: State of Art,” National Days of Network Security and Systems (JNS2), pp. 106-109, Apr. 2012.

[5] K. A. Nuaimi, N. Mohamed, M. A. Nuaimi and J. Al-Jaroodi, “A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms,” Second Symposium on Network Cloud Computing and Applications (NCCA), pp. 137-142, Dec. 2012.

[6] Amandeep, V. Yadav and F. Mohammad, “Different Strategies for Load Balancing in Cloud Computing Environment: A Critical Study,” International Journal of Scientific Research Engineering & Technology (IJSRET), vol. 3, no. 1, pp. 85-90, Apr. 2014.

[7] B. Wickremasinghe, R. N. Calheiros and R. Buyya, “Cloudanalyst: A Cloudsim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications,” Proceedings of the 24th IEEE

International Conference on Advanced Information Networking and Applications (AINA 2010), pp. 446-452, Apr. 2010.

[8] N.Swarnkar, A. K. Singh and Shankar, "A Survey of Load Balancing Technique in Cloud Computing," International Journal of Engineering Research & Technology, vol. 2, no 8, pp. 800-804, August 2013.

[9] S. S. Moharana, R. D. Ramesh and D. Powar, "Analysis of Load Balancers in Cloud Computing," International Journal of Computer Science and Engineering (IJCSE), vol. 2, no 2, ISSN 2278-9960, pp. 101-108, May 2013.

[10] K. Mahajan, A. Makroo and D. Dahiya, "Round Robin with Server Affinity: A VM Load Balancing Algorithm for Cloud Based Infrastructure," Journal of Information Process System, vol.9, no.3, September 2013

[11] A. Makroo and D. Dahiya, "An efficient VM load balancer for Cloud. Applied Mathematics," Computational Science and Engineering 2014

[12] S.G. Domanal and G.R.M. Reddy, "Load Balancing in Cloud Computing using Modified Throttled Algorithm," in Cloud Computing in Emerging Markets (CCEM), 2013 IEEE International Conference on (pp. 1-5). IEEE.

[13] M. Sharma and P. Sharma, "Performance Evaluation of Adaptive Virtual Machine Load Balancing Algorithm," International Journal of Advanced Computer Science and Applications, vol. 3, no 2, pp. 86-88, ISSN: 2156-5570, 2012.

[14] M. Sharma, P. Sharma and S. Sharma, "Efficient Load Balancing Algorithm in VM Cloud Environment," International Journal of Computer Science and Technology, vol. 3, no 1, pp. 439-441, ISSN: 0976-8491[online], ISSN: 2229-433[print], Jan-March 2012

[15] J. James and B.Verma, "Efficient VM Load Balancing Algorithm for a Cloud Computing Environment," International Journal on Computer Science & Engineering, vol. 4,pp. 1658-1663, ISSN: 0975-3397, September 2012.

[16] M. M. Ladani and Vinit Kumar Gupta, "A Framework for Performance Analysis of Computing Clouds," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 2, no 6, pp. 245-247, ISSN: 2278-3075, May 2013.

[17] M. Vaidehi, K.S. Rashmi and V. Suma, "Enhanced Load Balancing to Avoid Deadlock in Cloud," International Journal of Computer Applications on Advanced Computing and Communication Technologies for HPC Applications, pp. 31-35, June 2012

[18] http://www.internetworldstats.com as on Aug. 2015G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," *Phil. Trans. Roy. Soc. London*, vol. A247, pp. 529–551, April 1955.