Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 6, June 2015, pg.334 – 344

RESEARCH ARTICLE

A Parametric Weighted Approach to Perform Load Balancing in Cloud Environment

Priya[1], Er. Amit Batra[2]

[1]Research Scholar M.Tech., Department of Computer Science & Engineering, HCTM Kaithal, KUK, India, Email-Id: priya17122014@gmail.com

[2]Assistant Professor, Department of Computer Science & Engineering, HCTM Kaithal, KUK, India, Email-Id: amitbatra2011@gmail.com

Abstract- A cloud environment is the distributed system in which number of cloud servers and clients are connected using an intermediate service layer. The number of clients over the cloud system increases, the load on the best configuration and effective service provider server also increases. In such case, to provide the equal distribution of load among all servers, there is a requirement of some load balancing mechanism in cloud system. In this presented work, a two sided dynamic trust mechanism will be applied to perform the load balancing. The first level trust will be implemented on server side to perform the prioritization. On server side, the trust mechanism will be estimated under three vectors. By assigning the weightage to these three vectors, priority to each cloud server will be assigned. Once the server side priority will be defined based on trust analysis, the next work is to prioritize the clients based on the trust analysis. User trust will be defined under three vectors. Based on three vectors, the scheduling of the user requests will be performed. Now the scheduled process will be allocated to the cloud servers under the prioritized sequence. If the server is already loaded, the low priority process will be migrated to the low priority cloud. The main work is to execute the high priority user task on high priority cloud.

Key Words: Cloud Computing, Load balancing, Threshold, Priority, Intermediate Layer.

I. INTRODUCTION

Cloud computing is a recent technology that concern with online distribution of computing resources and services. In cloud computing, end-user knowledge about the configuration of service delivering system may not be required because client just use services on pay per model where all system configuration and resource management is taken care by cloud system automatically [16]. Cloud Computing has become one of the popular techniques adopted by both industry and academia providing a flexible and efficient way to store and retrieve the data files [14]. The definition of cloud computing provided by National Institute of Standards and

Technology (NIST) says that: "Cloud computing is a model for enabling, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, data storage, software applications and other computing services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [16]".

Cloud computing provides resources to users through virtualization technology It will reduce the coupling between the software and the hardware, and greatly improve the utilization of the resources [5]. Clouds use virtualization technology in distributed data centers to allocate resources to customers as the need them [6].

Virtualization technology provides an effective solution to the management of dynamic resources on Cloud Computing platform. Virtualization technology is able to carry out remapping between virtual machine (VM) and physical resources according to the load change so as to achieve the load balance of the whole system in a dynamic manner [7].

Load balancing is the task to be distributed among multiple computers, processes, disk, or other resources in order to get optimal resource utilization and to reduce the computation time. Load balancing is an important means to achieve effective resource sharing and utilization. It has been the hot issue of distributed computing, grid computing and cloud computing research.

Load balancing has two meanings: first, it puts a large number of concurrent accesses or data traffic to multiple nodes respectively to reduce the time users waiting for response; second, it put the calculation from a single heavy load to the multiple nodes to improve the resource utilization of each node [5].

Load Balancing is done using the prioritized list of data centers and client trust. When Load balancing is started, listing of trusted and un-trusted data centers/nodes is done. Trusted list consist of nodes having trust value greater than the threshold value in decreasing order i.e. the first node of list has the highest trust value. Similarly un-trusted node list consist of node with trust value less then threshold value in decreasing order [12].

II. RELATED WORK

Shu-Ching Wang et al. [15] presented a new concept that used low-power hosts to achieve high reliability. This approach was about to utilize the computing resources on the network to facilitate the execution of complicated tasks that require large-scale computation. The proposed scheduling algorithm combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms that can utilize more better executing efficiency and maintain the load balancing of system. **Branko Radojevic** et al. [2] presented the analysis of detected issues for those load balancing algorithms is presented. The new algorithm incorporates information from virtualized computer environments and end user experience in order to be able to proactively influence load balancing decisions or reactively change decision in handling critical situations. **Gaochao Xu et al.** [4] introduces a better load balance model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. The algorithm applies the game theory to the load balancing strategy to improve the efficiency in the public cloud environment. **Punit Gupta** et al. [12] proposed a suitable trust model based on the existing model that is suitable for trust value management for the cloud IaaS parameters. Based on the above achieved trust values, a suitable load balancing algorithm is proposed for better distribution of load which further enhance the QOS of services being provided to the users. . **Yatendra Sahu** et al. [16] introduce a threshold based Dynamic compares and balance algorithm (DCABA) for cloud server optimization. Presented approach can serve the purpose of service cost reduction in cloud industry with effective utilization of available resources. **Ren [9]** proposed a load balancing technique in dynamic environment based on WLC (weighted least connection) algorithm. It allocates the resource with least weight to a task and takes into account node capabilities. Based on the weight and capabilities of the node, task is assigned to a node.

III. OBJECTIVES

The objectives associated with presented work are given here under

- The main objective of the work is to design a two stage trust mechanism to improve the load balancing over the cloud system.
- The objective of the work is to design a weighted prioritization mechanism to increase trust for server side.
- The objective of the work is design a trust based scheduling approach on client side to decide the service execution order.
- The objective of the work is to handle the overload condition using load balancing and the migration approaches.
- The objective of the work is to reduce the failure rate of the process execution and to increase the reliability over the system

IV. PROPOSED WORK

The service providers provide Cloud Services to different users for their use. To provide the optimized distribution of these services; an interfacing is required between the users and the cloud servers. For this interfacing and composition we are providing an approach where we are defining an intermediate layer between the clouds servers and the users. The proposed system has a middle layer architecture called Intermediate Layer to perform the cloud allocation in case of under load and overload conditions. The over load conditions will be handled by using the concept of process migration. The middle layer will exist between the clouds and the clients. The intermediate layer will accept the user requests and also monitor the cloud servers for the available load over the VMs. The intermediate layer will perform the process allocation sequentially and if the service allocation is not possible for a specific VM, it will perform the migration of process from one VM to other.

The middle layer is responsible for three main tasks

1. Scheduling the user requests

2. Monitor the cloud servers for its capabilities and to perform the process allocation

3. Process Migration in overload conditions

In this presented work, a two side trust analysis approach is suggested to perform the effective service allocation to the cloud servers. The trust mechanism is here suggested for both the client side as well as on server side. On server side, the trust mechanism will be applied to assign the priorities to the cloud servers. The parameters that will be considered for the trust analysis are

(a) Server Configuration
(b) Availability
(c) Response Time

The server configuration will be presented in terms of physical capabilities or the resources available in the server. These resources include the CPU processing, memory capacity and number of I/O units. The second vector considered for the cloud server trust is the availability analysis. The availability of a server will be defined as the ratio for the capability to handle the requests. It will perform the analysis on the number of requests performed to the server and the number of request successfully handled by the server. It is also considered as the failure ratio analysis. The third vector considered here is the average response time. When a request will be handled by the server, the process time will be evaluated. The server with least average response time will be more effective. Once these parameters will be analyzed, the next work is to prioritize the cloud servers by assigning the weightage to these parameters.

Another side associated with the presented work is the client side trust analysis. This analysis will be defined under three main parameters

(a) Dead line criticality
(b) User requirement analysis
(c) User reliability

When a request will be performed to the server, one of the main request parameter will be dead line definition. The deadline is here defined in terms of the time definition by which the process will be executed completely. The deadline criticality will be considered as the trust requirement of a user. The another parameter considered is the user requirements in terms of CPU speed, memory or the number or I/O units required to run the process. The third vector that will be considered is the reliability vector. This vector is based on the user request and the abortion of the user request process. The ratio of the service requests and the successful completion of the process from user side will decide the trust ratio of the user. Higher the reliability ratio, more trustful the user will be. Once the parameters will be identified, the next work is to schedule the requests under these parameters.

After the scheduling process, the request allocation will be performed to map the user requests to the servers. If the server is already overloaded, the migration of the low priority process will be performed. The work is about to allocate a high trust process on highly trustful server. Greedy algorithm is used in the proposed work.

The process flow of the presented work is given as under

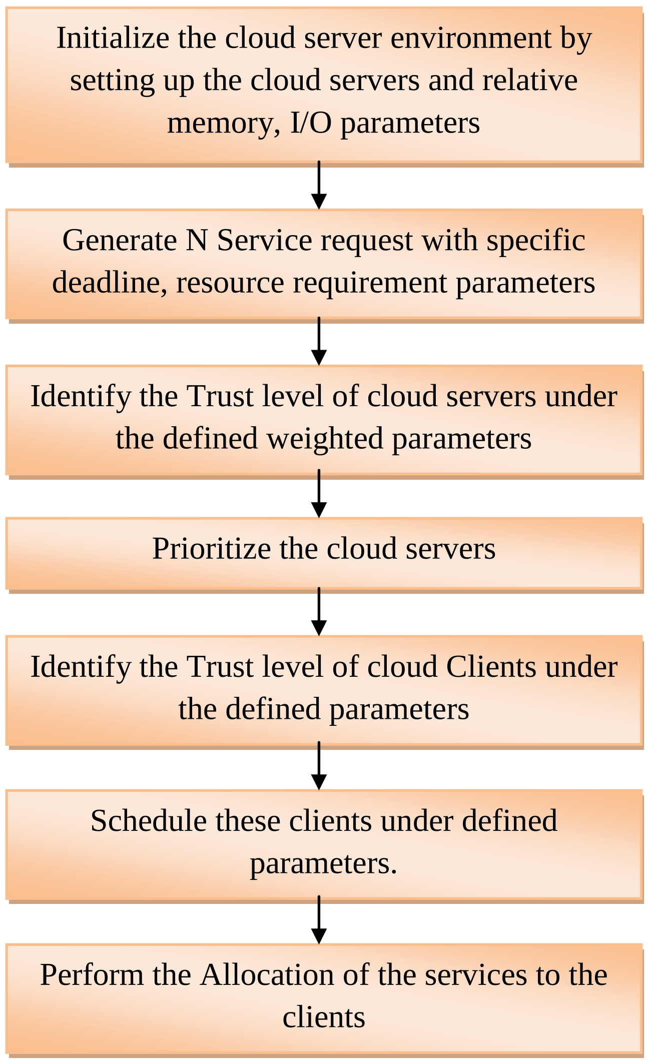

Initialize the cloud server environment by setting up the cloud servers and relative memory, I/O parameters
Generate N Service request with specific deadline, resource requirement parameters
Identify the Trust level of cloud servers under the defined weighted parameters
Prioritize the cloud servers
Identify the Trust level of cloud Clients under the defined parameters
Schedule these clients under defined parameters.
Perform the Allocation of the services to the clients

Figure 1: Flow of work

Algorithm

```
Algorithm (Cloud, VM, User)

/* The Cloud system is defined with L number of Clouds and M number of virtual machines. The number of users requests are N. As the system is generated, N numbers of parallel requests are performed over the server. The algorithm is here defined to perform the allocation of requests to cloud system*/

{
    1. Assign the Priorities to Cloud Servers
    2. Arrange the Cloud Server in decreasing order or priority
    3. For i=1 to L
       [Process All Clouds]
       {
       For j=1 to M
       [Process all virtual Machines]
       {
       If (VM (j).Memory>Threshold)
       {
       Set VM (j).Priority=High
       }
       If (VM (j).IO>Threshold)
       {
       Set VM (j).Priority=High
       }
       Else
       {
       Set VM (j).Priority=Low
       }
       }
    4. Arrange the User Requests in order of Memory Requirements
    5. Perform the Initial Level Assignment of User Request on Cloud Server under weighted parameter
    6. If the process get completed before deadline, execute the allocated process over the virtual machine
    7. If the process finish time exceeds the deadline perform process Migration
    }
```

V. RESULT ANALYSIS

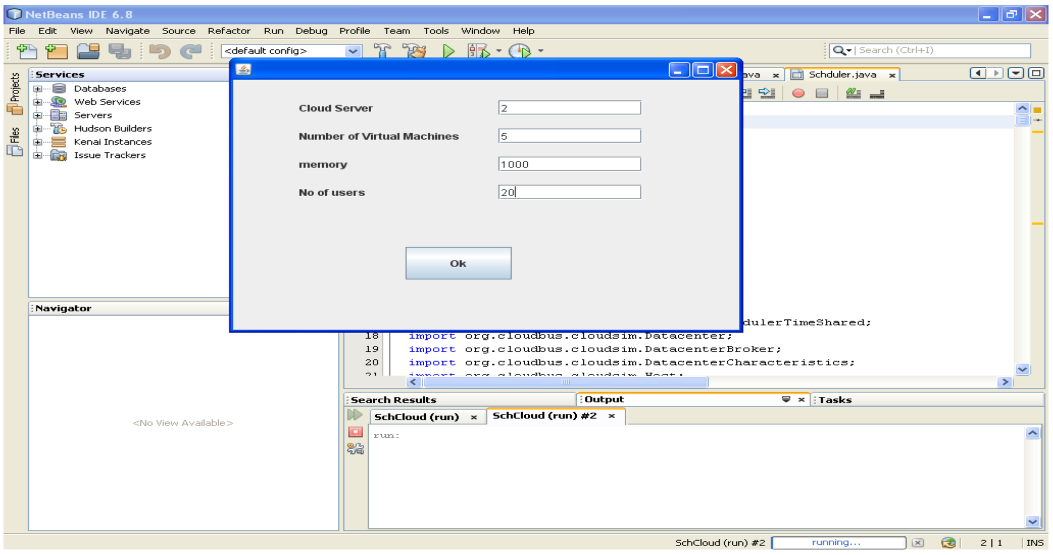
Tool: Cloudsim

The CloudSim toolkit has been chosen as a simulation platform as it is a modern simulation framework aimed at Cloud computing environments. In contrast to alternative simulation toolkits (e.g. SimGrid, GandSim), it supports modelling of on-demand virtualization enabled resource and application management. It has been extended in order to enable power-aware simulations as the core framework does not provide this capability. Apart from the power consumption modelling and accounting, the ability to simulate service applications with variable over time workload has been incorporated.

Language: Java

The presented work required an interface between the cloud users and the servers. It is required to present this interface in a user friendly way so that the possible options under different categories will be presented to users in effective way. Based on these values user can take the effective decision regarding the security options selected by the users. To provide the user friendly environment java-swing will be selected as the language library in which the graphical user and the code will be implemented.

Results



Cloud Server	2
Number of Virtual Machines	5
memory	1000
No of users	20

```
dulerTimeShared;
import org.cloudbus.cloudsim.Datacenter;
import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
```

```
run:
```

Figure 2: GUI of proposed work

Here figure 2 is showing the graphical interface to work on cloud system to perform the process scheduling. Here the input parameters are taken for server side settings and to generate the user requests.

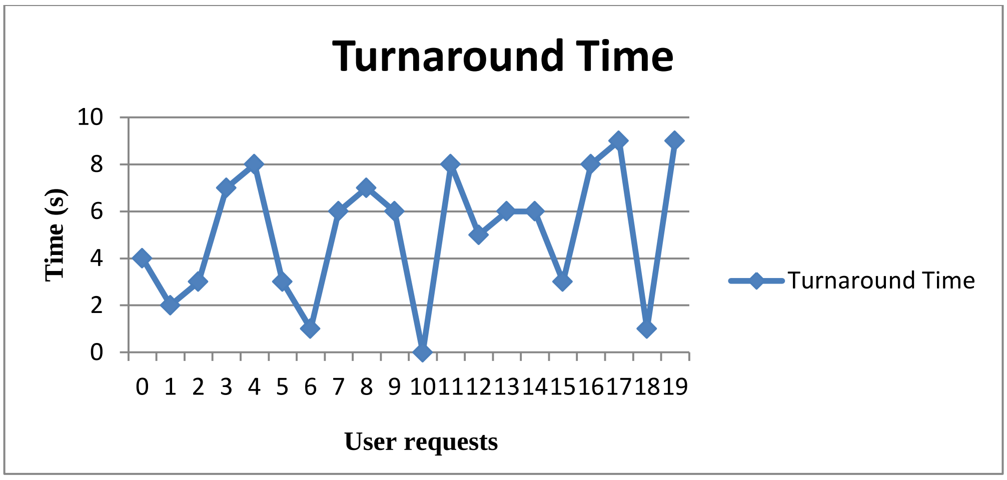


Figure 3: Turn Around time Analysis

Here figure 3 is showing the turnaround time anlaysis for 20 input user requests. Here x axis represents the number of user requests and y axis represents the turn around time in seconds. The figure shows the process time is between 0 and 10.

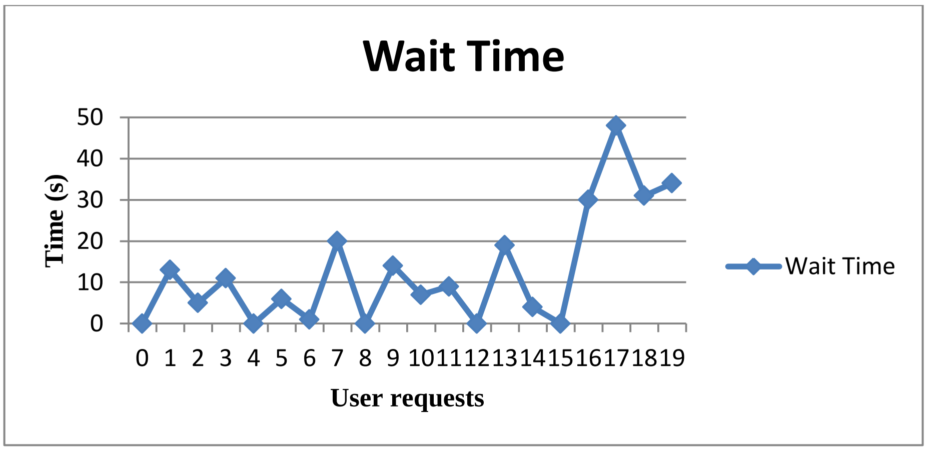


Figure 4: Wait Time Analysis

Here figure 4 is showing the Wait time anlaysis for 20 input user requests. Here x axis represents the number of user requests and y axis represents the Wait time in seconds. The figure shows the finish time is between 0 and 50

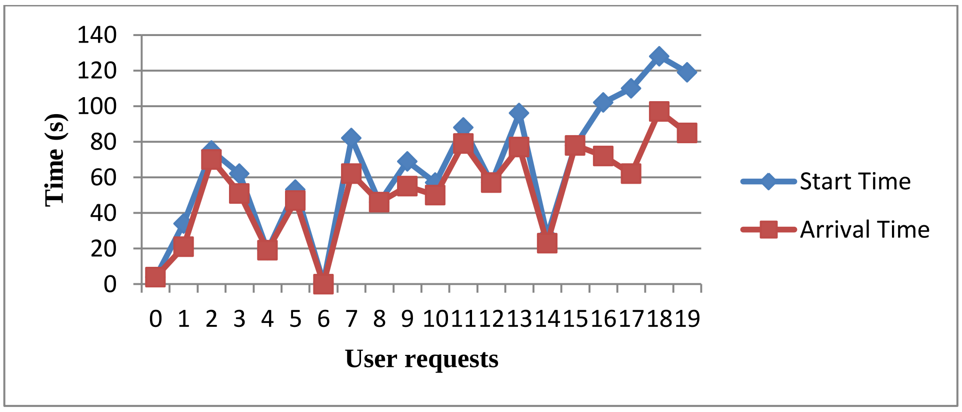


Figure 5: Start Time And Arrival Time Analysis

Here figure 5 is showing the start time and arrival time anlaysis for 20 input user requests. Here x axis represents the number of user requests and y axis represents the start and arrival time differnce in seconds. The figure shows the most or processes are executed without much delay.

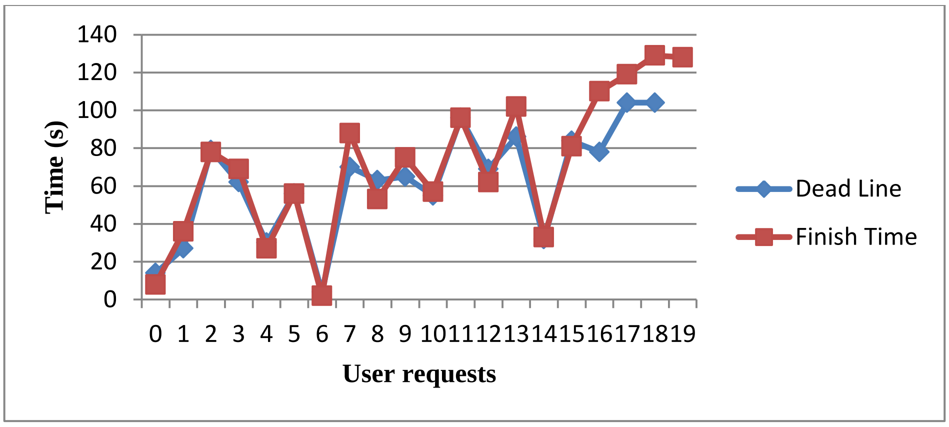


Figure 6: Dead Line and Finish Time Analysis

Figure 6 is showing the finish time and dead line anlaysis for 20 input user requests. Here x axis represents the number of user requests and y axis represents the finish time and deadline differnce in seconds. The figure shows that most or processes are executed within the dead line.

Comparative Analysis

Comparison between proposed work and existing work is shown as under:

Proposed work

Table 1: Analysis of average wait time

Number of user	average wait time	number of migration
10	4.9	2
15	5.3	4
20	6.1	7
25	7.3	8
30	7.8	10

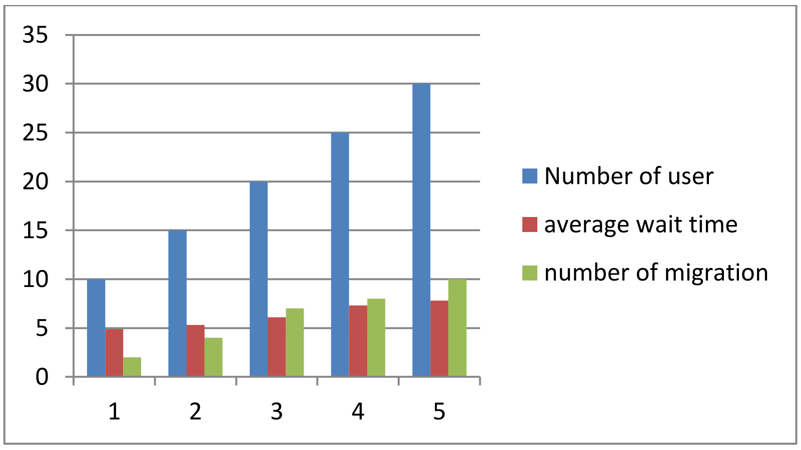


Figure 7: Analysis of users, wait time and migrations

Here figure 7 shows users, reduced wait time and migrations.

Existing Work

Table 2: Analysis of Convergence rate

Number of VMs	Convergence Rate
10	0.8
20	1.2
30	1.26
40	1.6
50	2.15
60	2.36
70	2.93
80	3.46
90	3.69

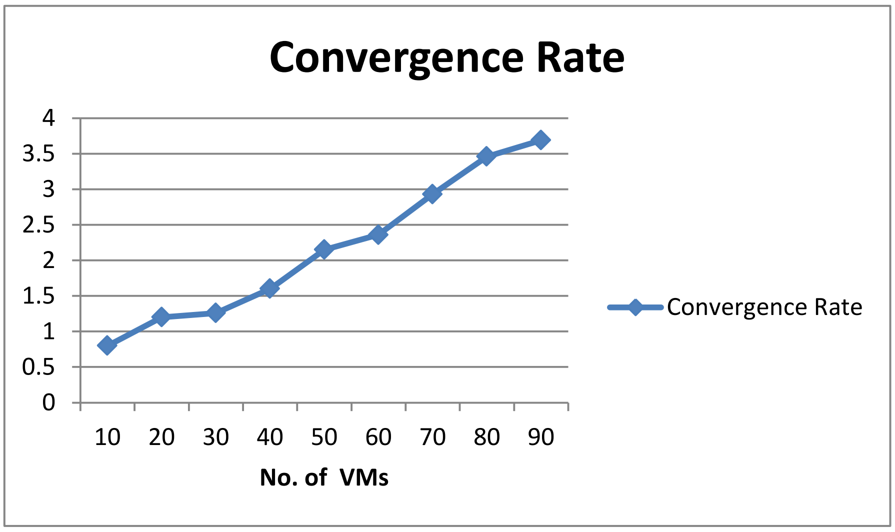


Figure 8: Analysis of Convergence Rate

Figure 8 shows Analysis of convergence rate. Here x-axis represents the no. of VMs and y-axis represents the convergence rate. Figure shows that effective convergence rate increases as VMs increases.

Table 3: Analysis of Average Execution Time

Number of VMs	Average Execution Time
10	12
20	16
30	19
40	21
50	27
60	34
70	43
80	52
90	57

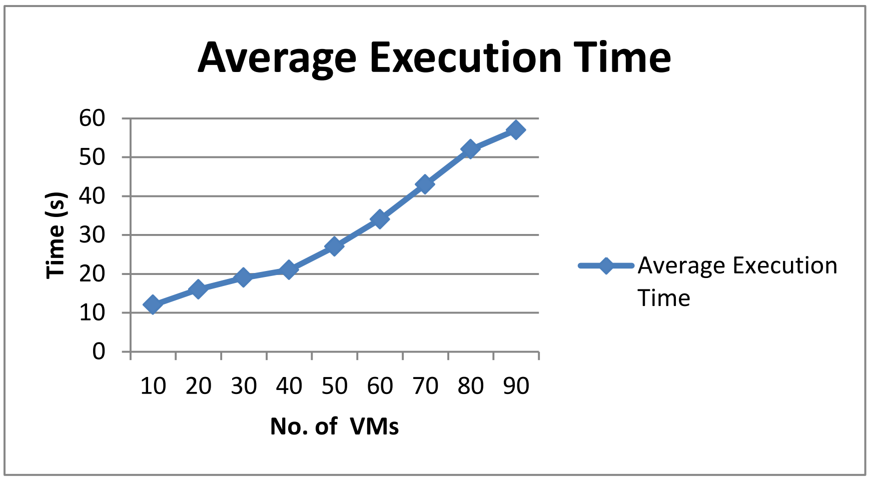


Figure 9: Analysis of average execution time

Figure 9 shows the analysis of average execution time. Here x-axis represents the no. of VMs and y-axis represents the time in seconds. The figure shows that effective time increases as VMs increases.

Our proposed is compared with Energy Efficient VM Scheduling for Cloud Data Centers: Exact allocation and migration algorithms [3]. Figure 8 and 9 address the performance of the consolidation algorithm. When the number of servers to consolidate increases further, as shown in figure 8, the convergence times move to orders of tens to hundreds of seconds (for the extreme case on the curve upper right corner, this reaches 180 minutes for 120 hosted VMs).These figures highlight the limits of exact migration algorithm with increasing number of servers to consolidate. But our proposed work performs the Load Balancing by reducing wait time.

VI. CONCLUSION

In this present work, a resource allocation scheme is applied on multiple clouds in both the under load and the over load conditions. As the request is performed by the user, certain parameters are defined with each user request, these parameters includes the arrival time, process time, deadline and the input output requirement of the processes. The Cloud environment taken in this work is the public cloud environment with multiple clouds. Each cloud is here defined with some virtual machines.

To perform the effective allocation, we have assigned some priority to each cloud. The virtual machines are here to perform the actual allocation. These are defined with certain limits in terms of memory, load etc. As the allocation begins, at first the scheduling of the processes is performed respective to the memory requirements. And along with it, the allocation of the process is done to the Cloud based on the requirement and the availability analysis. If the allocated process cannot be executed in its required time slot, in such case the migration of the process is required. The migration of the processes is here defined in case of overload conditions. The overload condition is defined in terms of simultaneous processes that are required to execute at particular instance of time. The analysis of the work is done in terms of wait time, process time of the processes. The obtain results show the successful execution of all the processes within time limit. The work is performed on a generic system that can have n number of Clouds.

VII. FUTURE WORK

The presented work is about to perform the scheduling and the allocation of the processes to the clouds in case of under load and overload conditions. In case of over load condition, the migration of the processes is performed from one cloud to other.

The Future enhancements of the work are possible in the following directions:

1. The presented work is defined the overload conditions in terms of deadline as well as the memory limit of the Clouds. In future some other parameters can also be taken to decide the migration condition.

2. The presented work is defined for the public Cloud environment, but in future, the work can be extended to private and the hybrid Cloud environment.

REFERENCES

[1] Abhishek Gupta, Osman Sarood and Laxmikant V Kale, “Improving HPC Application Performance in Cloud through Dynamic Load Balancing”, 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 2013 IEEE

[2] Branko Radojevic and M. Zagar ," Analysis of Issues with Load Balancing Algorithms in Hosted (Cloud) Environments", In proc. 34th International conventation on MIPRO, IEEE, 2011.

[3] Chaima Ghribi, Makhlouf Hadji and Djamal Zeghlache, “Energy Efficient VM Scheduling for Cloud Data Centers: Exact allocation and migration algorithms”, 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 2013 IEEE

[4] Gaochao Xu, Junjie Pang and Xiaodong Fu, " A Load Balancing Model Based on Cloud Partitioning for the Public Cloud", Tsinghua Science and Technology ISSNl l1007-0214, 2013

[5] Haozheng Ren, Yihua Lan and Chao Yin, “The Load Balancing Algorithm in Cloud Computing Environment”, 2nd International Conference on Computer Science and Network Technology, 2012 IEEE

[6] Jeffrey Galloway, Karl Smith and Jeffrey Carver, “An Empirical Study of Power Aware Load Balancing in Local Cloud Architectures”, Ninth International Conference on Information Technology- New Generations, 2012 IEEE

[7] Jinhua Hu, Jianhua Gu, Guofei Sun and Tianhai Zhao, “ A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment”, 3rd International Symposium on Parallel Architectures, Algorithms and Programming, 2010 IEEE

[8] Klaithem Al Nuaimi, Nader Mohamed, Mariam A1 Nuaimi and Jameela A1-Jaroodi, “A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms”, Second Symposium on Network Cloud Computing and Applications, 2012 IEEE

[9] Lee, R. & Jeng, B., “Load-balancing tactics in cloud”, In proc. International Conference on Cyber- Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 447-454, 2011 IEEE.

[10] Maggie Mashaly, Paul J. Kuhn, “Load Balancing in Cloud-based Content Delivery Networks using Adaptive Server Activation/Deactivation”, 2012 IEEE

[11] Minxian Xu, Wenhong Tian, “An Online Load Balancing Scheduling Algorithm for Cloud Data Centers Conssidering Real-Time Multi-Dimensional Resource”, 2012 IEEE

[12] Punit Gupta, Mayank Kumar Goyal and Prakash Kumar, “Trust and Reliability based Load Balancing Algorithm for Cloud IaaS”, 2012 IEEE

[13] Rui Wang, Wei Le, Xuejie Zhang, “Design and Implementation of an Efficient Load-Balancing Method for Virtual Machine Cluster Based on Cloud Service”, 2011IEEE

[14] Shridhar G.Domanal and G.Ram Mohan Reddy, “Load Balancing in Cloud Computing Using Modified Throttled Algorithm”, 2013 IEEE

[15] Shu-Ching Wang, Kuo-Qin Yan, Wen-Pin Liao and Shun-Sheng Wang, “Towards a Load Balancing in a Three-level Cloud Computing Network”, 2010 IEEE

[16] Yatendra Sahu, R.K. Pateriya, Rajeev Kumar Gupta," Cloud Server Optimization with Load Balancing and Green Computing Techniques Using Dynamic Compare and Balance Algorithm", 5th International Conference on Computational Intelligence and Communication Networks, 2013 IEEE

[17] Yilin Lu, Jian Zhang, Shaochun Wu and Shujuan Zhang," A Hybrid Dynamic Load Balancing Approach for Cloud Storage", International Conference on Industrial Control and Electronics Engineering, 2012 IEEE

[18] Zheng Hu, Kaijun Wu, Jinsong Huang, “An Utility-Based Job Scheduling Algorithm for Current Computing Cloud Considering Reliability Factor”, 2012 IEEE

[19] Zhang, **“**A Model Based Load-Balancing Method in IaaS Cloud”, 42nd International Conference on Parallel Processing, 2013 IEEE