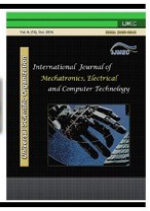
Vol. 7(24), Apr. 2017, PP. 3314-3324

Contents list available at IJMEC

International Journal of Mechatronics, Electrical and Computer Technology (IJMEC)

Journal Homepage: www.aeuso.org

PISSN: 2411-6173 - EISSN: 2305-0543



Designing a Smart Method for Load Balancing in Cloud Computing

Hamed Mahdizadeh

Università degli Studi, Firenze, Italy

Phone Number: *+39-329-4997543*

****Corresponding Author's E-mail:*** *hamed.mahdizadeh@unifi.it*

Abstract

Load balancing in cloud computing is an important management and scheduling issue; because, in cloud computing environment, each user may face with hundreds of virtual resources for running each task. Thus, selecting the best resource for performing the task is one of the most important challenges. Various load balancing algorithms have been proposed to respond these problems; of course, some of these algorithms have some weak points besides their strong points. In this paper, a smart method is presented for load balancing. Previous load balancing methods were based on software and relative to time, and optimization was performed locally. In the proposed method, global optimization is done along the time. In the proposed system, load balancing is based on hardware and computation of load balancing is done at two levels of the whole system and the virtual machine or server. The proposed method leads to a better use of the resources.

Keywords: *Cloud Computing, Load balancing, Virtual Machine, Data Center.*

1. Introduction

By rapid development of the processing and storage technologies and the success of internet, the computer resources became cheaper, more powerful, and more accessible than before. This technological trend has enabled a new realization of computing model called cloud computing, in which the resources (e.g., CPU and storage devices) are provided as services that can be rented and released by the users via the Internet in a demand-based fashion. In a cloud computing environment, the traditional role of the service provider is divided into two parts:

1. Providers of the infrastructures who manage the cloud operating system and rent the resources based on the usage-based pricing model.
2. Providers of the services who rent the resources from one or many infrastructure providers to serve the end users [12].

Generally, cloud computing includes a number of distributed servers of providing the services to the customers based on the requests in a scalable and reliable network. The load balancing problem in cloud computing is a new challenge, which always needs a distributed solution. Since, in practice, it is not always possible and cost-effective to keep one or more service providers as idle in order to respond to the needed requests, thus the tasks cannot be assigned (allocated) to the appropriate server. Load balancing refers to the procedure of reallocating the total load to the unique nodes of a collective system in order to effectively utilize the resources, improve the response time, and simultaneously eliminates a status in which some of the nodes are overloaded while some others are less loaded**.** The load balancing algorithms are categorized as the static and dynamic algorithms. Static algorithms are mostly suitable for homogeneous and stable environments and can yields very desirable

Article History:

IJMEC DOI: 649123/10.225122

Received Date: Nov. 28, 2016

Accepted Date: Feb. 14, 2017

Available Online: Mar. 01, 2017

results in such environments; however, they are usually inflexible and unable to adapt themselves with the dynamic changes of the features during runtime. On the other hand, dynamic algorithms are more flexible and take into account a variety of features in the system in both modes, i.e. before and during runtime [8]. Load balancing is a process which improves the system's performance by reallocating the load among the processors [2]; therefore, load balancing in cloud and performance of the system is one of the most important challenges of the experts and researchers who have attempted in their researches to investigate the resources and problems (troubles) occurring in running (executing) the applications and the problem of scheduling the system's performance as well. Responding to the requests and managing and scheduling such that it yields the best possible response in the shortest time requires readiness, scheduling, and management of the failures. In addition, many of the research findings including genetic algorithm, ant colony, artificial intelligence, etc. are used to solve the problem of load balancing. In this paper, it is attempted to design a smart system for managers of the system, which can provide suggestions for changing the system's configuration so that they can optimize the system and improve the load balancing through applying these changes.

2. Load balancing in cloud computing

The problem of load balancing in cloud computing is a new challenge. Always a distributed solution is required. Since, in practice, it is not always possible and cost-effective to keep one or more service providers as idle in order to respond to the needed requests, the tasks cannot be assigned (allocated) to the appropriate server and effective load balancing separately for customers in the cloud structure is very complicated and the components are present across a wide zone (region). Load balancing includes the procedure of reallocating the total load to the unique nodes of a collective system in order to effectively utilize the resources, improve the response time, and simultaneously remove a stats in which some of the nodes are overloaded while some others are less loaded [7]. The load balancing algorithms are categorized as the static and dynamic algorithms. The static algorithms are mainly suitable for homogeneous and stable environments and can yield very good results in such environments; however, they are usually inflexible and unable to adapt themselves with dynamic changes of the features during runtime. On the other hand, the dynamic algorithms are more flexible and bring into consideration a variety of feature-relevant tasks in the system in both modes, i.e. before and during runtime [8]. Load balancing is a process that improves the system's performance by reallocation of the load among the processors [7].

3. Related works

The study [3], in addition to focusing on load balancing and maximizing the use of cloud resources and appropriately responding to the requests, deals with the problems of similar requests as well. This article proposed a two-way method for fast loading of the files as dynamic load balancing in the cloud, called DDFTP (dual-direction FTP), so that if two similar versions of a file are present in two service providers, it will be divided and shared using the FTP protocol (File Transfer Protocol) and then the shared packages are controlled using the TCP protocol (Transmission Control Protocol). In this method, the service providers process the divided parts and, at the end, provide the requester with a general response.

In another article, a task scheduler algorithm based on the priorities in cloud computing was proposed [7]. The main objective of implementing the load balancing in this algorithm includes scheduling the task, achieving highly-efficient computing, and achieving the best throughput of the system. This algorithm is consisted of three levels of priority including scheduling level (target level), resource level (feature level), and task level (replacement level). For each requested task, a resource with priority is determined. Priority of each task relative to another task is examined separately. To investigate the tasks and allocate the resources, a task scheduling algorithm based on the priority in cloud computing was proposed so that, in the first step, all of the resources and tasks are inserted. Then, a comparison matrix compatible to all of the tasks is created according to the priority of access

to the resources and, finally, the priority vector is calculated for all the matrices according to a series of formula.

Another article investigated the load balancing and providing quick QOS service to the requests. This algorithm is consisted of five parts. The first part includes the cloud customers who are the end user receiving the services from the cloud. They demand a different type of services that is faster since each task has its own specific priority. The second part is divided into two categories, namely the pre-processing and separator tasks:

(1) Pre-processing determines the ratios of different tasks based on the time cost.
(2) Separator classifies the tasks depending on the type of the task and provides them for the scheduler.

The third part is consisted of the central and output data, which is a main component of providing the service for the users. It includes the storage resources, service providers, storage units, etc.; further, it is the main processing unit and is considered as a computational resource in this paper. Each processing unit takes the task from the relevant distributor queue and schedules it using the scheduler. In the fourth part, the manager of the central data collects the efficiency of the processing unit to specify that how many instructions have been performed by which machine at what length of time to determine its time cost. In the fifth part, which is the task scheduler, following the user's request, the information is received from the pre-processing and central data manager units in order to determine which processing unit can do the task at lower time cost and, eventually, the task is sent to an appropriate processor [1].

A well as maintaining the load balance, the main objective of this article is to use the full power of the cloud for providing a better and faster response for the requester. The clouds are classified into various types among which the present paper is focused on public clouds [6].

1- Nodes are distributed in a three-level structure in which the task is distributed among the nodes.
2- This method combines the opportunistic load balancing (OLB) method, to maintain each of the busy nodes, and the load balance Min-Min (LBMM), to achieve the minimum time for performing each task. The criteria considered in this study include efficiency and resource utilization [13].

In [10], a comparative study on load balancing algorithms for cloud computing was presented. This article considers three practical methods for load balancing in the large-scale cloud systems. Firstly, a nature-inspired algorithm might be used for self-organizing and achieving the global load balancing through applying the local server. Secondly, self-organization can provide the load balancing in all the nodes of the system based on random sampling from the system's range (domain). And thirdly, the system can be restructured to optimize the allocation of tasks in servers. Recently, several network models and nature-inspired computations have attracted the attention of many researchers toward a search for distributed methods to address the increasing scale and complexity of these systems.

This article [4] also proposes a load balancing model based on the tree view of a network. This load balancing strategy has two main objectives: (a) reducing the average response time of the tasks sent to the grid, and (b) reducing the communication costs in transferring the tasks. This strategy focuses on three layers of the algorithm (intra-site, intra-cluster and intra-grid).

4. Cloud load balancing criteria (measures)

Various parameters are taken into account in cloud load balancing techniques [11]. Figure (1) shows different types of load balancing.

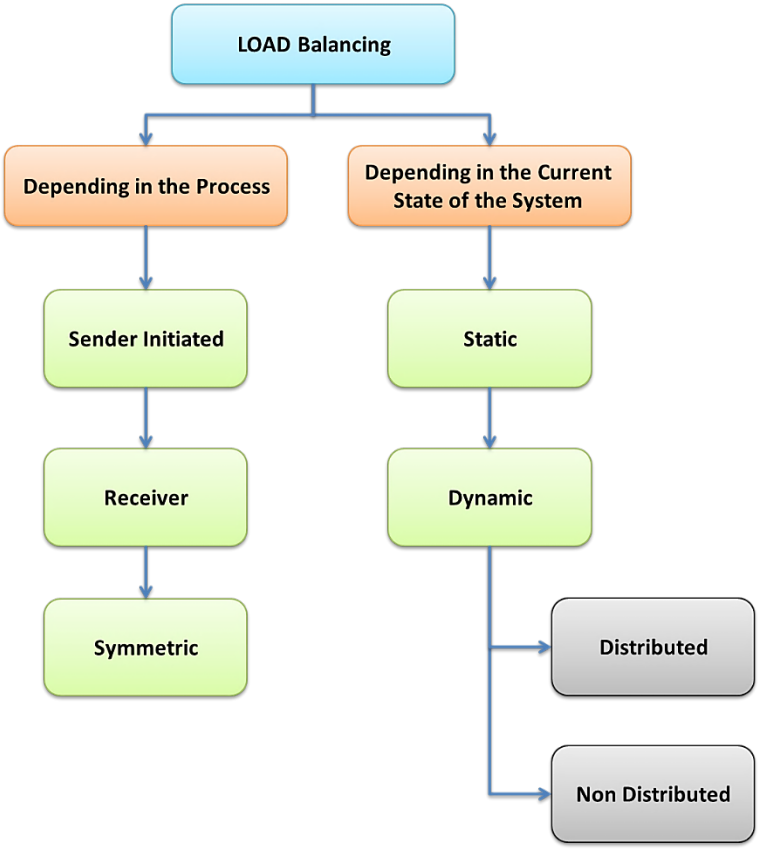


Figure 1: System Based on Cloud Computing

Relevant overhead: when a load balance algorithm is executed, it determines value of the involved overhead, including the overhead specified for improving the tasks, relationships between processors, and processes. This value should be minimized in order for the load balancing technique to work efficiently.

Operational power: is used for calculating the terminated tasks. This value should be high for improving the systems' efficiency.

Efficiency: is used for controlling the system's productivity. This criterion should improve the reasonable expenses. For example, it should reduce the task time (duration) while keeping the delays at an acceptable level.

Exploitation of resources: is used for controlling exploitation of the resources. This value should be optimized for an efficient load balancing.

Scalability: is the ability of an algorithm to execute load balancing for a system with infinite number of nodes. This criterion should be improved.

Response time: is the time considered for responding that has been distributed in the system by a specific load balancing algorithm. This parameter should be minimized.

Error tolerance: is the ability of an algorithm to monotonously execute load balancing in case of connection failure. Load balancing should be a good error tolerance technique.

Migration time: is the time of migration of tasks or resources from one node to another. This time must be minimized in order to increase the system's efficiency.

5. Load balancing proposed for cloud computing

In cloud computing, whenever a virtual machine (VM) is loaded with several (multiple) tasks, these tasks should be eliminated and sent to less loaded VMs in the same data center. In this case, when we eliminate more than one task from the overloaded VM and if more than one VM exists for processing that task, then the tasks must be sent to the VMs which have an appropriate combination of priorities, meaning that the tasks should not wait too long for processing. In the VM level, load balancing is done inside the data center.

Previous implementations of load balancing have been done in software mode and as local optimization. In the proposed method, load balancing for cloud computing is done in hardware mode and as global optimization. As stated earlier, in previous methods, load balancing was done in software mode at the server level. In the proposed system, load balancing is performed at two levels:

1. Whole system level [9].
2. Virtual machine or server level [5].

At both levels, load balancing is calculated so that if there is no load balancing at any level of the system, the server or the whole system which is less loaded is transferred to an overloaded place in hardware mode. In the proposed system, regardless of the scheduling algorithms, the changes applied on the system lead to the load balancing; therefore, the proposed algorithm for load balancing works well on the VMs in the cloud computing environment.

5.1 Calculation of load balancing in the whole system

Capacity of a VM (All formulas of this section has been used from reference [9]).

$$C_j = Pe_{numj} \times Pe_{mipsj} + VM_{bwj} \quad (1)$$

where Pe_{numj} is the number of processors in a VM_j, Pe_{mipsj} is million instructions per second of all the processors in VM_j, and VM_{Bwj} is the ability of the VM_J communication bandwidth.

Capacity of All the VMs

$$C = \sum_{i=1}^{m} C_j \quad (2)$$

Capacity of the data center is the total capacity of all the virtual machines.
Load in a VM
The total length of the tasks allocated to a virtual machine is called load.

$$L_{V,M_j,t} = \frac{N(T,t)}{S(VM_i,t)} \quad (3)$$

The load of a virtual machine can be calculated as the number of the tasks at the time *t* in the service queue of VM_i divided by the service rate of VM_i at the time t. The load of all the virtual machines in a data center is calculated as follows:

$$L = \sum_{i=1}^{m} L_{VM_j} \quad (4)$$

Time of processing a virtual machine

$$PT_i = \frac{L_{VM_i}}{C_j} \quad (5)$$

Time of processing all the virtual machines

$$PT = \frac{L}{C} \quad (6)$$

Load standard deviation

$$\sigma = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(PT_i - PT)^2} \quad (7)$$

After finding the workload and the standard deviation, the system must decide whether load balancing should be performed or not. For this purpose, there are two possible states: (1) Finding out whether the system is balanced (2) Find out whether the whole system is saturated or not (whether the whole group is overloaded or not). If it is overloaded, load balancing is meaningless.

Finding the state of the virtual machines group
If the standard deviation of the VM load (σ) is less than or equal to the threshold set of the conditions (Ts) [1-0], then the system is balanced, otherwise it is in an unstable state.

Finding overloaded groups
When the workload of the virtual machine group exceeds the maximum capacity of the group, then the group is overloaded.

5.2 Calculation of load balancing at server level

The load balancing method presented in this study is dynamic method, which not only balances the load at the whole system level and the server level, but it also considers the task priority in the machines' waiting queue. (All formulas of this section have been used from reference [5]).

Calculation of load imbalance function
The sum of the loads of all the virtual machines can be defined as follows:

$$L = \sum_{i=1}^{K} L_i \quad (8)$$

Where *i* represents the number of the VMs (Virtual Machines) in a data center. The unit capacity of each load is defined as follows:

$$LPC = \frac{L}{\sum_{i=1}^{m} C_i} \quad (9)$$

where C_i is the node's capacity. The load imbalance function of a particular VM is as follows:

$$\text{Threshold } T_i = LPC * C_i \quad (10)$$

$$\text{If VM} \begin{cases} < |T_i - \sum_{v=1}^{K} L_v|, & Underloaded \\ > |T_i - \sum_{v=1}^{K} T_v|, & Overloaded \\ = \left|T_i - \sum_{i=1}^{T} T_i\right|, & Balanced \end{cases} \quad (11)$$

Switching (transferring) the load from the overloaded VM to the less loaded VM can yield a better load balancing in the system; further, in the detected less loaded VM in which the sum of the loads of the VMs is below the threshold value, transferring the load from the overloaded VM

to the less loaded VM is continued until their load reaches below the threshold value. The less loaded VM can accept the load when it does not exceed the threshold level.

6. Results and evaluation of the proposed method

A cloud computing system must handle various obstacles such as the network's flow, load balance in virtual machines, scalability, trust management, and other instances. Generally, research on cloud computing focuses on these problems based on their different importance. Clouds provide a set of services (software and hardware) in an unprecedented scale. Cloud services are responsible for handling the demand diversity in time through dynamic supplying or not supplying from the clouds.

After all, we cannot directly use the cloud computing system. Testing the new techniques or strategies in actual real operation of cloud computing is not practically possible, since these tests and experiments endanger the quality of the service required by the end-users such as security, cost, and speed. In this section, the performance of the proposed algorithm is analyzed based on the results of the simulation performed using C# software; further, the balance of migration between the virtual machines is studied at the end. To implement the proposed method, NASA's data set for two months of August and July 1995 has been used. To make a better assessment of the proposed method, in Figure (2), a part of the system with three data center has been considered, in each of the data centers there are three servers or virtual machines.

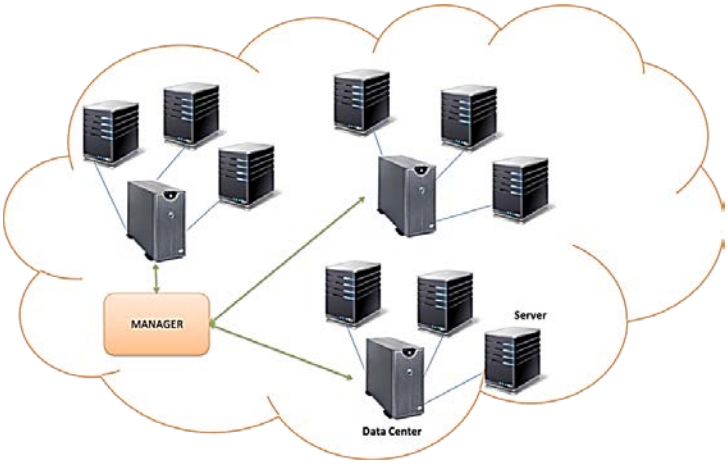


Figure 2: System Based on Cloud Computing

To implement the proposed method, the number of data centers, and number of each virtual machine (or server) in data centers, server memory, bandwidth, and server capacity were considered; for example, for the data center (1) with three servers, the above-mentioned parameters are presented in Table (1).

Table 1: Profile servers for data centers

DataCenterID	VM_ID	CPU_Count	CPU_MIPS	Memory	BandWidth	Capacity
1	1	2	4000	4000	200	0
1	2	2	1000	4000	100	0
1	3	4	2000	4000	50	0

Figure (2) shows a part of the system based on cloud computations (computing). To implement the method, numerous servers were considered; so that, after applying the proposed method, the system's load balancing was investigated at different times.

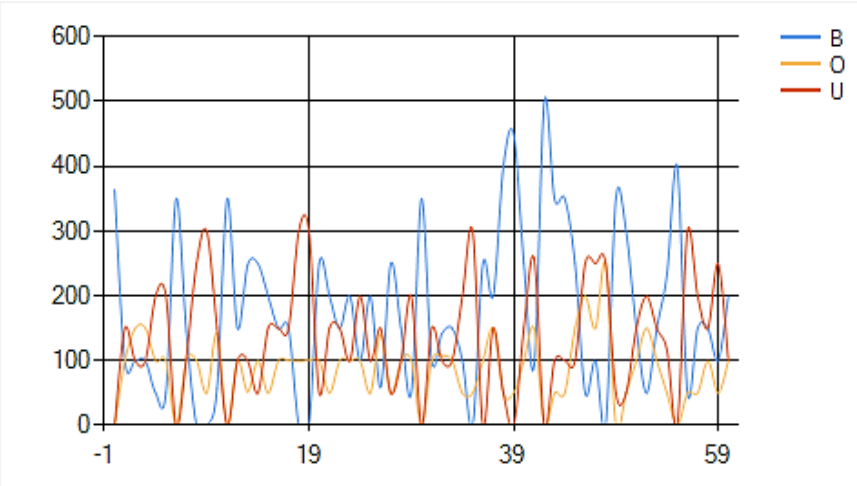


Figure 3: Results for load balancing in proposed system for initial architecture of system

Figure (3) shows the number of the servers which are balanced, overloaded, or less loaded at different times.

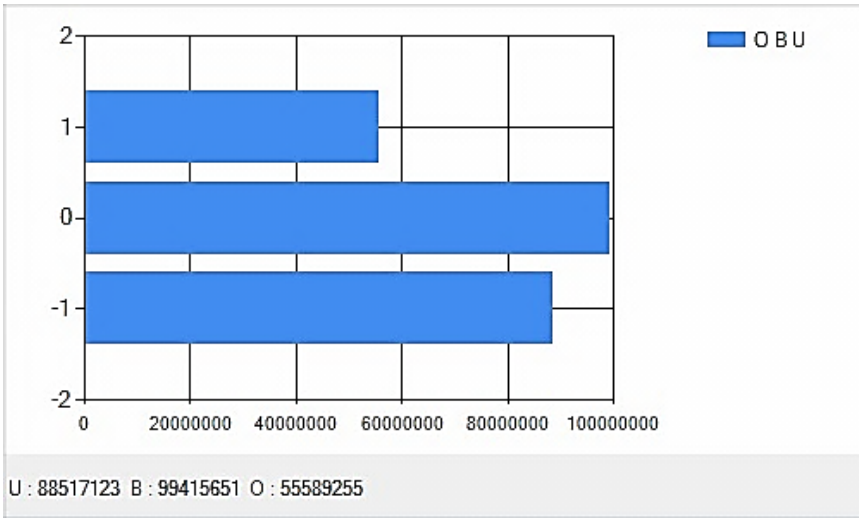


Figure 4: Values obtained in early steps of implementation

Figure (4) shows the value of load balancing before switching in the system. These values are related to the initial architecture of the system. As seen in Figure (4), in the first step of load balancing implementation in the system, as seen in Figure 4, the balance value is 99 million (server per second). This value has been obtained before the inter-server switching from a data center occurs.

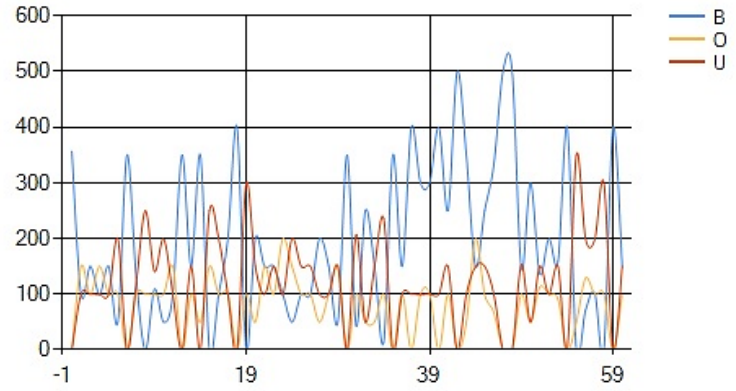


Figure 5: Results for load balancing in proposed system for initial architecture of system

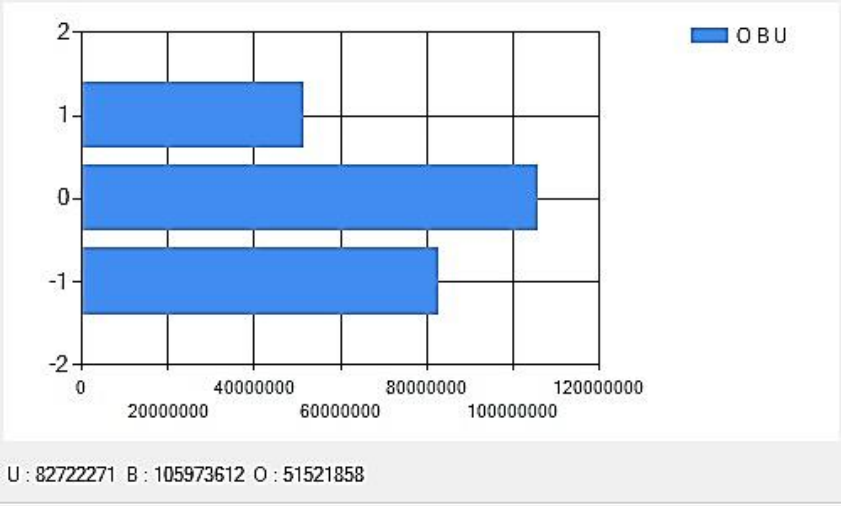


Figure 6: Values obtained in early steps of implementation

Figure (5) shows the number of servers that are balanced, overloaded, or less loaded at different times. In the first step, after occurrence of the inter-server switching from the data center and modification of the whole system, the load balancing at the server and the whole system levels is recalculated. The obtained values are presented in Figure (6). As seen in Figure (6), the balance value in the early steps is 99 million (server per second) which reaches to 105 million (server per second) in the second step, as a result of which a better load balancing is achieved in the system.

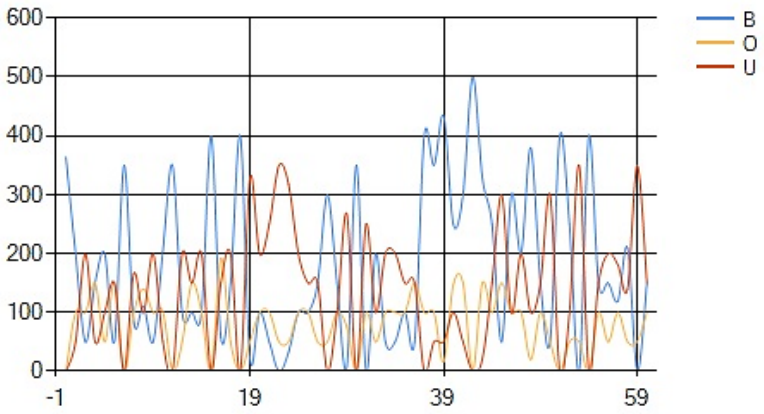


Figure 7: Results for load balancing in proposed system for initial architecture of system

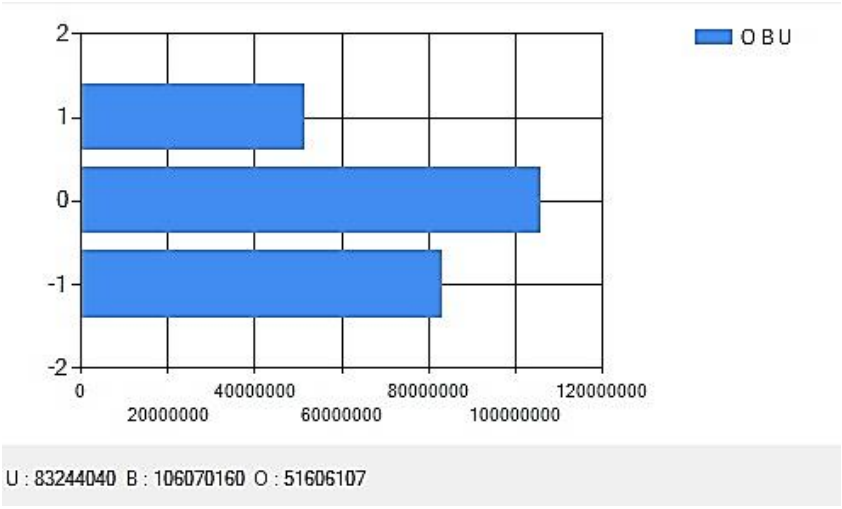


Figure 8: Values obtained in early steps of implementation

Figure (7) shows the number of the servers that are balanced, overloaded, and less loaded at different times. In the third step, again after the inter-server switching from the data center, calculation of the load balancing is performed at the server level and the whole system level. The obtained values are presented in Figure (8). As seen in Figure (8), the balance value was 105 million in previous steps, which reached to106 millions (server per second) in the third step. These values show no significant increase; therefore, repetition of the steps is no longer necessary, and the results obtained in the third stage show the best result for load balancing in the system.

Conclusion

In this article, a smart load balancing algorithm was proposed, the objective of which is to achieve the load balancing in the global optimization and to achieve the maximum power. The proposed method calculates the load balance at two levels; namely, the server level and the whole system level. So, if there is no load balancing between each component, switching between the virtual machines is done in hardware mode. In the proposed system, regardless of the scheduling algorithms, the changes applied on the system led to the load balancing. The empirical (experimental) results show that our proposed algorithm is significantly effective. Our approach shows that significant considerable improvement occurs in the average runtime and creation of a logical load balance in the system. Thus, in future, we will plan to develop this type of load balancing for the workflow with relevant tasks. This algorithm considers priority as the main parameter of QoS; therefore, in future, we will plan to improve this algorithm by taking other factors of QoS into account.

Reference

[1] Abdullah Monir, "Cost-Based Multi-QoS Job Scheduling using Divisible Load Theory in Cloud Computing", sciverse science direct، ICCS 2013، Procedia Computer Science 18, pp. 928 – 935, 2013.

[2] Abhijit A. Rajguru, S.S. Apte, "A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters" , International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-1,Issue-3, August 2012.

[3] Al-Jaroodi، Jameela, "A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud", Journal of Network and Computer Applications ، Journal of Network and Computer Applications 36, pp. 1116–1130, 2013.

[4] B. Yagoubi, M. Medebber, A load balancing model for grid environment, computer and information sciences, 2007. iscis 2007, in: 22nd International Symposium on, 7–9 Nov, 2007, pp. 1–7.

[5] D. Chitra Devi and V. Rhymend Uthariaraj, "Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Non preemptive Dependent Tasks", Hindawi Publishing Corporation, the Scientific World Journal Volume 2016, Article ID 3896065, pp. 1-16.

[6] Fu‹ Xiaodong, "A Load Balancing Model Based on Cloud Partitioning for the Public Cloud", tsinghua science and technology‹ ISSNl, pp. l34 – 39, 2013.
[7] Ghanbari‹ Shamsollah, "A Priority based Job Scheduling Algorithm in Cloud Computing", sciverse Science direct‹ ICASCE 2012‹ Procedia Engineering 50, pp.778 – 785, 2012.
[8] Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi and Jameela Al-Jaroodi, "A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms" , 2012 IEEE Second Symposium on Network Cloud Computing and Applications, 978-0-7695-4943-9/12, 2012.
[9] L.D. Dhinesh Babu, P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", Applied Soft Computing, 2013, PP. 1-12.
[10] M. Randles, D. Lamb, A. Taleb-Bendiab, A comparative study into distributed load balancing algorithms for cloud computing, in: Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, Perth, Australia, April, 2010, pp. 551–556.
[11] Nidhi Jain Kansal , Inderveer Chana, Cloud "Load Balancing Techniques : A Step Towards Green Computing", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, January 2012.
[12] Qi Zhang, Lu Cheng, Raouf Boutaba, "Cloud Computing: state-of-the-art and research challenges", J Internet Serv Appl (2010) 1: 7–18 DOI 10.1007/s13174-010-0007-6 J Internet Serv Appl (2010) 1: 7–18, 2010.
[13] S.C. Wang, K.Q. Yan, W.P. Liao and S.S. Wang, "Towards a Load Balancing in a Three-level Cloud Computing Network", Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology, pp. 108-113, 2010.

Autor(s)

Hamed Mahdizadeh is a PhD candidate in department of Computer Science at the University of Florence, in Italy. He received his master's degree in Computer Engineering, in April 2014 field of Artificial intelligence at Eastern Mediterranean University in North Cyprus. His main research interests are Load Balancing in Cloud Computing, simulation modeling and operations research. He also is working on Multi Objective Optimization Algorithm, PSO, ABC, DAP Problem.

email: hamed.mahdizadeh@unifi.it
Phone Number: (+39-329) 4997543