Computers and Electrical Engineering 000 (2016) 1–7

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

CLB: A novel load balancing architecture and algorithm for cloud services[☆]

Shang-Liang Chen, Yun-Yao Chen[*], Suang-Hong Kuo

Institute of Manufacturing Information and Systems, National Cheng Kung University, Tainan, Taiwan, ROC

A R T I C L E   I N F O

Article history:
Received 8 September 2014
Revised 10 January 2016
Accepted 12 January 2016
Available online xxx

Keywords:
CLB
Load balance
Cloud computing

A B S T R A C T

Cloud services are widely used in manufacturing, logistics, digital applications, and document processing. Cloud services must be able to handle tens of thousands of concurrent requests and to enable servers to seamlessly provide the amount of load balancing capacity required to respond to incoming application traffic in addition to allowing users to obtain information quickly and accurately. In the past, researchers have proposed the use of static load balancing or server response times to evaluate load balancing capacity, a lack of which may cause a server to load unevenly. In this study, a dynamic annexed balance method is used to solve this problem. Cloud load balancing (CLB) takes into consideration both server processing power and computer loading, thus making it less likely that a server will be unable to handle excessive computational requirements. Finally, two algorithms in CLB are also addressed with experiments to prove the proposed approach is innovative.

1. Introduction

With the rapid development of the Internet, many vendors have started to provide cloud services. More and more services can be obtained in the cloud, and users do not have to do operations on a local computer. All operations are computed on the cloud. When a large number of users attempt to access cloud services, this often causes the server to fail to respond. Determining a method by which to provide users with timely and accurate responses is a subject worthy of advanced study. Several studies have been proposed to evaluate and to develop algorithms and load balancing methodologies for cloud-based applications. It is difficult for a server to deal effectively with the flow of information generated by all of the various enterprises attempting to access it. Excessive flow causes server overload with a subsequent loss of information. A server load balancing mechanism can disperse the transmission of information flow and data operations and can also reduce the probability of increased computational time and loss of information. When one server fails in the cloud, the cloud services can be transferred to another server. Services are therefore non-stop. The advantages of server load balance include:

1. Efficiency: improve the efficient use of the server and network bandwidth.
2. Reliability and Safety: improve the reliability and security of the server.
3. Scalability: improve the scalability of services.

[☆] Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. W-H Hsieh.
[*] Corresponding author. Tel.: +886 62757575x61001
E-mail address: yewyewchen@gmail.com (Y.-Y. Chen).

http://dx.doi.org/10.1016/j.compeleceng.2016.01.029
0045-7906/© 2016 Elsevier Ltd. All rights reserved.

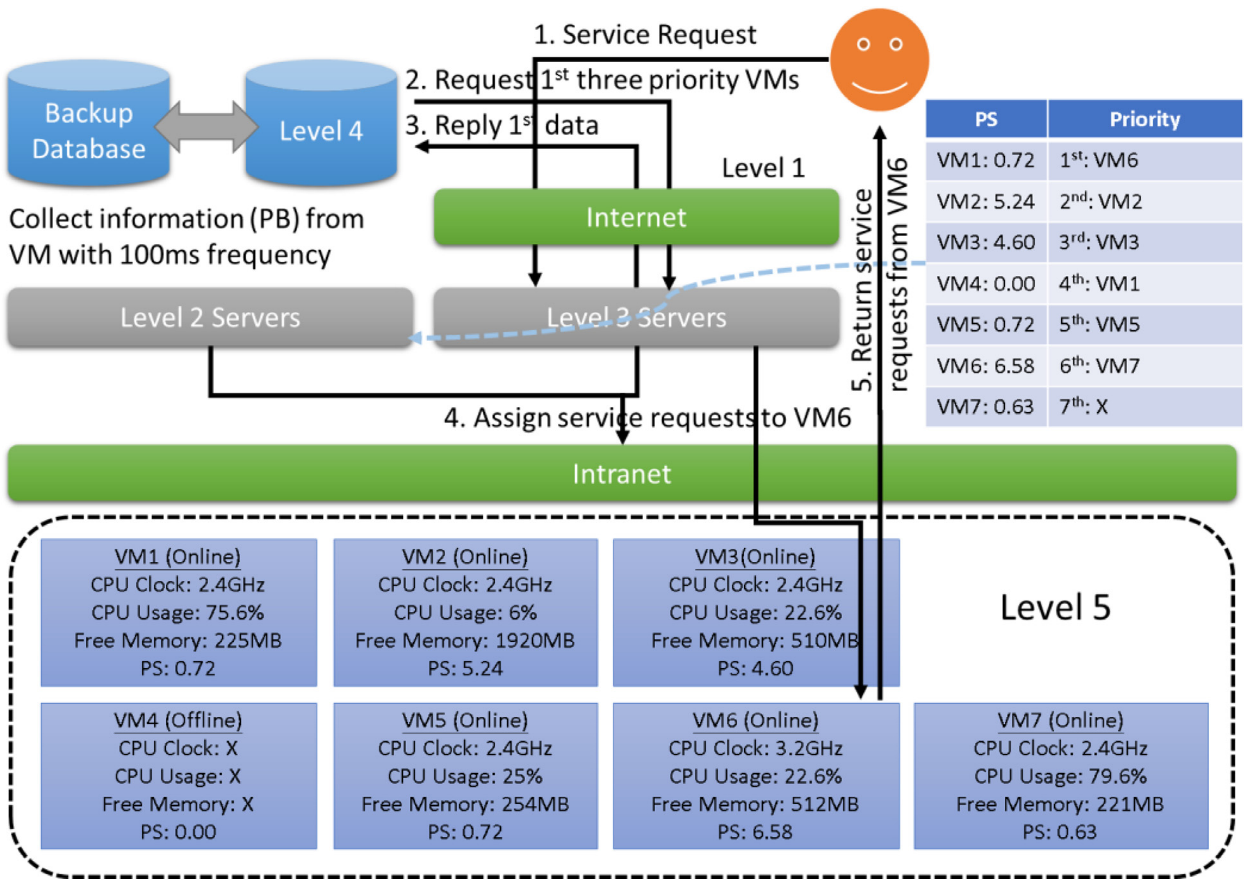
1. Service Request
2. Request 1st three priority VMs
3. Reply 1st data
Backup Database
Level 4
Level 1
Internet
Collect information (PB) from VM with 100ms frequency
Level 2 Servers
Level 3 Servers
5. Return service requests from VM6
4. Assign service requests to VM6
Intranet
PS
Priority
VM1: 0.72 | 1st: VM6
VM2: 5.24 | 2nd: VM2
VM3: 4.60 | 3rd: VM3
VM4: 0.00 | 4th: VM1
VM5: 0.72 | 5th: VM5
VM6: 6.58 | 6th: VM7
VM7: 0.63 | 7th: X
Level 5
VM1 (Online) CPU Clock: 2.4GHz CPU Usage: 75.6% Free Memory: 225MB PS: 0.72
VM2 (Online) CPU Clock: 2.4GHz CPU Usage: 6% Free Memory: 1920MB PS: 5.24
VM3(Online) CPU Clock: 2.4GHz CPU Usage: 22.6% Free Memory: 510MB PS: 4.60
VM4 (Offline) CPU Clock: X CPU Usage: X Free Memory: X PS: 0.00
VM5 (Online) CPU Clock: 2.4GHz CPU Usage: 25% Free Memory: 254MB PS: 0.72
VM6 (Online) CPU Clock: 3.2GHz CPU Usage: 22.6% Free Memory: 512MB PS: 6.58
VM7 (Online) CPU Clock: 2.4GHz CPU Usage: 79.6% Free Memory: 221MB PS: 0.63

Fig. 1. Cloud load balancing architecture.

Currently, there are several types of load balancing techniques for cloud-based applications. A server load balancing method is divided into two types of hardware and software:

1. Hardware: Fourth layer network switches provide server redundancy and load balancing.
2. Software: Used to calculate server usage and to allocate resources.

Regardless of whether software or hardware is used, the load balancing method is divided into two types: dynamic and static, which are explained as follows:

1. Static: Used to deal with predictable processing loads in advance.
2. Dynamic: Used to deal with unpredictable processing loads. Based on network storage virtualization, the host and storage devices use fiber channel switches that link together, and all virtualization requests return to the network storage device. This method does not depend on the operating system.

Static load balancing uses the polling approach (also called the Round Robin approach). This approach is sequentially assigned to each host, as shown in Fig. 1. This method is simple and uses fewer resources, but is usually unable to detect the attached server, resulting in annexation or uneven distribution. Shadrach proposed an RTSLB algorithm based on a weighted metric and compared the results to three algorithms, including a random load distribution algorithm, a Round Robin load distribution algorithm and a competitive learning algorithm [1]. A comparison of existing load balancing techniques for current research hotspots was proposed by Raghava [2] and is listed as Table 1.

Based on previous studies on this topic [13], we propose a new paradigm for load balancing architecture and a new algorithm, which can be applied to both virtual web servers and physical servers. There was several discussions of load balancing techniques for hardware and software, including grid-based, microprocessor-based, wireless sensor network-based approaches [14–18]. In addition, we discuss more current approaches to load balancing techniques for cloud-based applications, which are listed in Table 2.

The rest of the paper is organized as follows: The introduction and earlier studies on cloud load balance are briefly summarized in Section 1. The methodology of the proposed segmentation is detailed in Section 2 to Section 4. Experimental results and discussions are shown in Section 5. Finally, conclusions are drawn in Section 6.

Table 1
Comparison of existing load balancing techniques for current research hotspots.

Metrics/Techniques	Throughput	Overhead	Fault tolerance	Migration time	Response time	Resource utilization	Scalability	Performance
Round Robin [3]	YES	YES	NO	NO	YES	YES	YES	YES
Dynamic Round Robin [4]	YES	YES	YES	YES	NO	YES	NO	NO
PALB [5]	YES	YES	YES	YES	YES	YES	YES	NO
Active Monitoring [6]	YES	YES	NO	YES	YES	YES	YES	NO
FAMLB [7]	YES	YES	YES	YES	NO	YES	NO	YES
Min–Min [8]	YES	YES	NO	NO	YES	YES	NO	YES
Max–Min [9]	YES	YES	NO	NO	YES	YES	NO	YES
OLB+LBMM [10]	NO	NO	NO	NO	NO	YES	NO	YES
Throttled [11]	NO	NO	YES	YES	YES	YES	YES	YES
Honeybee Foraging [12]	NO	NO	NO	NO	NO	YES	NO	NO
Active Clustering [12]	NO	YES	NO	YES	NO	YES	NO	NO
Biased Random Sampling [12]	NO	YES	NO	NO	NO	YES	NO	YES
CLB (our approach)	YES	YES	YES	NO	NO	YES	YES	YES

Table 2
Discussion of current approaches to load balancing techniques for cloud-based applications.

No.	Authors	Research contributions
1	Ajit [14]	This paper presented the analysis of three contemporary algorithms in a cloud analyst tool to resolve the issue of cloud load balancing as a preparation phase for a new load balancing technique. A Weighted Signature based load balancing (WSLB) algorithm is proposed to minimize user response time. Further, this paper also provides the anticipated results with the implementation of the proposed algorithm.
2	Sahu [15]	In this paper, a threshold-based Dynamic compare and balance algorithm (DCABA) was introduced for cloud server optimization. Unlike the traditional server optimization strategies which consider only load balancing and scheduling of resources based on the usage of CPU, RAM and BW in physical servers, DCABA also minimizes the number of host machines to be powered on in order to reduce the cost of cloud services.
3	Wu [16]	An algorithm of prediction-based elastic load balancing resource management (TeraScaler ELB) was presented in this paper to overcome specific drawbacks. Experiments have shown that the required number of virtual machines changes in compliance with changes in the network load; thus, TeraScaler ELB is able to dynamically adjust the processing capacity of backend server cluster with the applied load. In addition, it can make full use of the 'use on demand' feature of cloud computing. TeraScaler ELB thus leads to a better application of prediction-based load balancing in cloud computing. This study concludes that compared with the traditional elastic resource management algorithm, TeraScaler ELB is more reasonable for providing scalability and high availability.
4	Soni [17]	In this paper, the "Central Load Balancer," a load balancing algorithm was proposed to balance the load among virtual machines in a cloud data center. The results showed that this algorithm can achieve better load balancing in a large scale cloud computing environment as compared to previous load balancing algorithms.
5	Suresh [18]	This paper introduced a better load balancing model for the public cloud based on the cloud partitioning concept with a switch mechanism to select different strategies for different situations. An adaptive neuro fuzzy inference system (ANFIS) -based load balancing algorithm and a Glowworm swarm optimization (GSO) -based load balancing algorithm were proposed in the load balancing strategy in order to improve the efficiency of the public cloud environment. The experimental results showed that the proposed method gives better results when compared with other traditional methods.
6	Zhang [19]	The contributions of this paper include: (1) a model that forecasts the load and estimates the resource requirements of virtual machines in the IaaS Cloud, (2) a scalable framework for load balancing which uses the proposed resource requirement forecasting model. The experiments conducted in this work showed that this method can accurately estimate the resource requirements of virtual machines and suggest that it works well in the proposed load balancing framework.
7	Li [20]	The distributed load balancer proposed in this work takes advantage of decentralized architecture for the purpose of providing scalability and high availability capabilities in order to service a large number of cloud users. A neural network-based dynamic load balancing algorithm called the nn-dwrr (neural network-based dynamic weighted round robin) was also proposed to dispatch a large number of requests to different VMs that are actually providing services. In the nn-dwrr, VM load metrics (CPU, memory, network bandwidth, and disk I/O utilizations) are combined to monitor and provide neural network-based load prediction to adjust the weight of each VM. The experimental results supported the premise that the proposed load balancing algorithm, nn-dwrr, can be applied to a large cloud datacenter and indicate that it is 1.86 times faster than the wrr, 1.49 times faster than the Capacity-based method, and 1.21 times faster than ANN-based load balancing algorithms in terms of average response time. In addition, the tldlb can reduce the SLA (service level agreement) violation rate via in time activating VMs from a spare VM pool.

2. Dynamic load balance for web servers

A dynamic load balancing method is based on tasks related to processing time and the computational power consumption of each server as well as other factors that determine the allocation of users to different servers in order for them to obtain services. Because the processing time for tasks will lead to an uneven distribution of the server, scholars have used three different dynamic equilibrium algorithms to solve this problem.

Table 3
Pseudo code for the least connection load balancing method.

```
for (j=0; j < n; j++) {
    for (i=j+1; i < n; i++) {
      if (C(Si) < C(Sj))
        m=i;
      }
    return Sj;
  }
  return NULL;
```

Table 4
Values of WRR Dynamic algorithm symbols.

Symbol	Value
ϕ_i	Weight, which represents the shared load percentage
λ	Arrival rate
μ_i	Service rate

Table 5
Values of cloud load balancing algorithm symbol.

Symbol	Value
PS	Priority service value
C_S	CPU speed(GHz)^2
L_C	CPU idle rate (1 CPU utilization rate)
L_R	RAM idle size (MB)= The total size of RAM(MB)–The used size of RAM(MB)
R_V	If L_R>500(MB) R_V=1
	If $L_R \leq$500(MB) R_V= L_R/500

2.1. Least connection scheduling

The least connection load balancing method assigns the user's connection to each server to which the user connects. The load balance architecture is designed based on the quantity of user connections. The distributor server is connected to the HTTP request, and each new user sends an HTTP request that will be assigned to the minimum number of connections with the server [2]. Suppose a group of servers S={the S0, S1... the Sn−1}, C (Si) denotes the current number of connections the server Si. Its pseudo code is shown in Table 3. However, the number of connections making up the server load is not necessarily high, so this method will result in uneven load distribution.

2.2. Load balanced min–min

Kokilavani etc., proposed the Load Balanced Min–Min algorithm, which divides a network into n subnetworks. Each sub network is allocated to elect the minimum execution time. Since Min–Min chooses the smallest tasks first, it loads more of the fast executing resources, which leaves the other resources idle. However, it is simple and produces a good makespan compared to other algorithms [3].

3. Weighted scheduling

The weighted load balancing method is where the server connections deal with performance or the weighted operation of service providers, and the results are shown as a load balancing priority weight. Chang [5,6] proposed the WRR-Dynamic algorithm using the M/M/1 Model [4], with the arrival rate and service rate calculation of the weight values as shown in Formulae (1) and (2), and where the symbols represent values as shown in Table 4. However, this study did not take into account each server's different load approach in which, the current study, the weighted load balancing algorithms and server load are taken into account; a new load balancing algorithm is designed, and the designed algorithm is actually applied to the cloud server.

$$\phi_i = \frac{\mu_i}{\lambda}\left(1 - \sqrt{\frac{\lambda}{\mu_i \beta}}\right) \tag{1}$$

$$\beta = \frac{\frac{1}{\lambda}\left[\sum_{i=1}^{N}\sqrt{\mu_i}\right]^2}{\frac{1}{\lambda^2}\left[\sum_{i=1}^{N}\mu_i\right]^2 - \frac{2}{\lambda}\left[\sum_{i=1}^{N}\mu_i\right] + 1} \tag{2}$$

4. Cloud load balance

4.1. The cloud load balance architecture

In this study, the design of the cloud load balancing (CLB) mechanism, the structure shown in Fig. 1, is divided into five layers:

Level 1: Users of the cloud service request, from the PS results from the X, is able to pick the server before the big to the small sort based on the network packet flow.

Level 2: Cloud load balance monitoring platform (CLBMP)

(1) Detects all service loads and determines whether the service is online.
(2) Sorts the server load multiplied by the weight.
(3) Stores the sorted load values to a database.

Level 3: Cloud load balance distribution platform (CLBDP)

(1) Receives user requests.
(2) Obtains the first X overall server.
(3) Assigns user requests to the first X server based on the inquiry wheel load balancing method.

Level 4: The database of the load information server stores the priority service value (PS) of each server calculated according to each time interval (0.1 s).

Level 5: Cloud server

(1) Provides storage from the cloud-service pool server group.
(2) Responds to user requests.

4.2. The cloud load balance algorithm

The cloud load balancing algorithm is designed for monitoring platforms in order to obtain each server loading, computing power and the priority service value (PS). It computes PS values every 0.1 of a second, and values are stored in the database. The PS formula is as shown in Eq. 3, and the value is as shown in Table 3. When the user sends a request to the cloud server, demand for services will enter into the cloud load balancing distribution platforms. The cloud load balancing distribution platform will obtain the first half of the servers sorted by the PS values from the service priority database, and it then uses a polling method to dispatch the user's requests to the first half of the servers.

$$PS = L_C \times C_S \times R_V \tag{3}$$

An example of the CLB algorithm is illustrated in Fig. 2. First, we separated the packets based on the weighting of 4 virtual machines (VM1-4). Given the weighting as w1 to w4, according to the performance of each virtual machine, the first round is used to dispatch the resources for all 4 VMs (the VM1 for the first packet, the VM2 got the second packet, and so on.) Then, in the second round, due to the order of each virtual machine, if the VM1 loading is too heavy to process the packets, the second round starts from VM2, and so on. This algorithm considers the basic loading of each server based on CPU, RAM and other performance differences. The time complexity is (O(1)), which is lower to estimate. For a simple queue, a 100Mbit/s bandwidth intranet can at a minimum gain 20Mbits/s of bandwidth for the lowest node.

5. Experiment results and discussions

In this study, a new cloud load balancing mechanism is proposed. This method can be used to calculate server processing power and can also load and obtain PS values. The testing server uses OS Windows7; the Programming Language is MS Visual Studio 2010 C# with the MS SQL Server 2005 database. The Server is Internet Information Services 2.0 with a RAM of 4,096 MB and an Intel T2390/1.86 GHz CPU.

Cloud load balancing distribution platforms are based on the PS value of the user assigned to the most appropriate server. This allows the user to get information quickly and accurately. In this study, load balancing architecture can be applied to the applications in the Cloud, thus allowing smoother system operation. Test results are shown in Tables 6 and 7, which indicate that the average response time increases with the number of connections for both physical and virtual web servers. Comparing performance, the results of Test 1 are similar to those of Test 3, both of which are faster than Test 2. However, this gap is within 0.002 s. The average response time gap between Test 2 and Test 1 increases with the number of connections. Therefore, a maximum number of 4,000 test results is chosen as a basis for further performance analysis. Given the same number of connections, the virtual web server architecture is shown to reduce the time by more than half of the time required by the physical web server architecture, and it is also shown to solve the problem of access failure. This proves that cloud service providers can improve server performance under the system architecture proposed in this study.

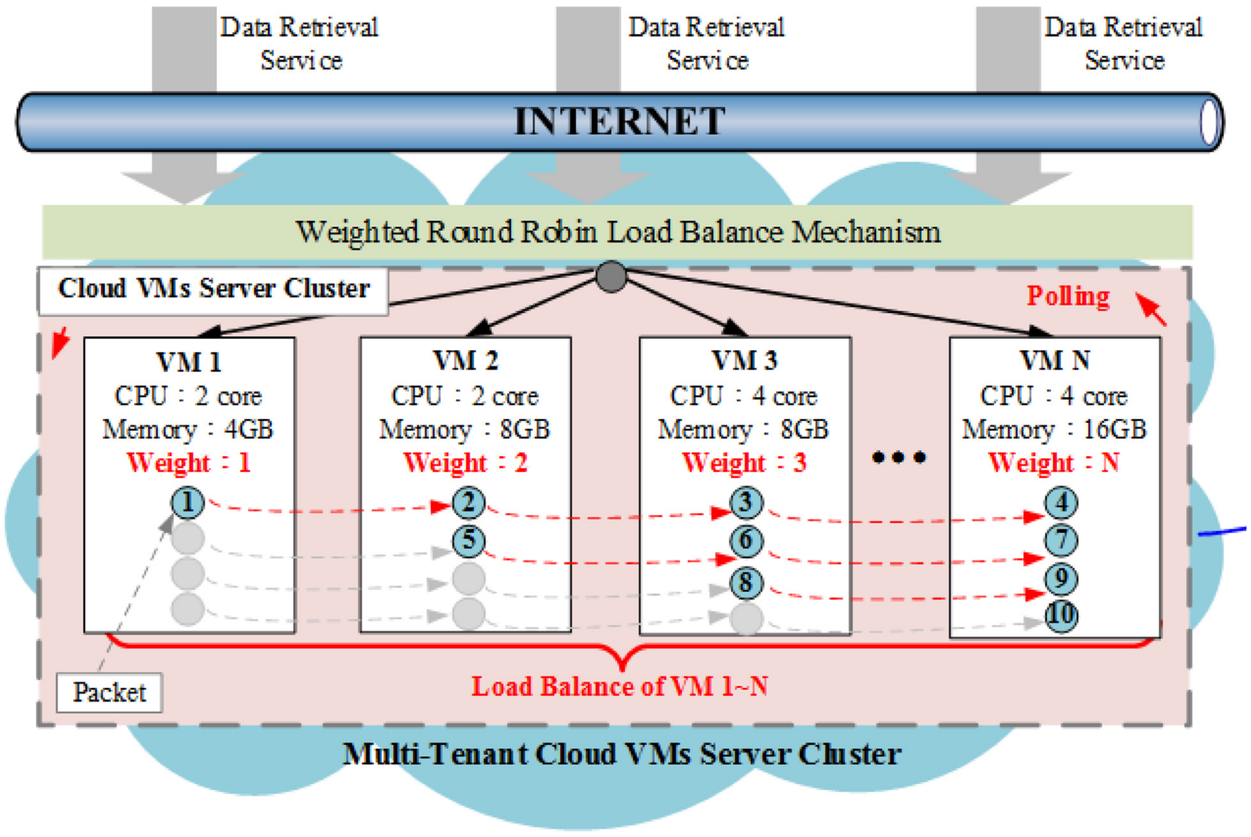

Fig. 2. The cloud load balancing algorithm process.

Table 6
Test results for 4,000/8,000 users concurrently logged into CLB-enabled physical servers.

Host no.	Testing environment	Total online users	Total time-cost(ms)	Average users/per ms
1	Test 1	4000	8.648 ± 1.000	0.0011
2	Test 2	8000	7.381 ± 1.000	0.0018

Table 7
Test results of 4000/8000 users concurrently logged into CLB-enabled virtual servers.

Host no.	Testing environment	Total online users	Total time cost(ms)	Average users/per ms
1	Test 1	4000	7.226 ± 1.000	0.0018
2	Test 2	8000	20.521 ± 1.000	0.0026

6. Conclusions

In this study, a new cloud load balancing mechanisms is proposed by comparing previous studies. The proposed new paradigm CLB for load balancing architecture and an algorithm can be applied to both virtual web servers and physical servers. From the experiments for CLB-enabled physical servers and virtual servers, the results show that cloud server performance based on the architecture proposed in this study can balance the loading performance when users logged in at the same time.

Acknowledgments

The authors thank the Ministry of Science and Technology No. MOST-103-2221-E-006 -085 -, who sponsored this research and provided related technological support. Due to support from the Ministry of Science and Technology, this research could proceed smoothly.

References

[1] Bryhni H. A comparison of load balancing techniques for scalable Web servers. IEEE Trans Netw 2000;14(4):58–64.
[2] Raghava NS, Singh Deepti. Comparative study on load balancing techniques in cloud computing. Open J Mob Comput Cloud Comput 2014;1(1):18–25.
[3] Subramanian S, Nitish Krishna G, Kiran Kumar M, Sreesh P, Karpagam G. An adaptive algorithm for dynamic priority based virtual machine scheduling in cloud. Int J Comput Sci Issues (IJCSI) 2012;9(6).
[4] Lin C-C, Liu P, Wu J-J. Energy-aware virtual machine dynamic provision and scheduling for cloud computing, in cloud computing (CLOUD). In: Proceedings of International Conference on 2011 IEEE. IEEE; 2011. p. 736–7.
[5] Galloway JM, Smith KL, Vrbsky SS. Power aware load balancing for cloud computing. In: Proceedings of the World Congress on Engineering and Computer Science, 1; 2011. p. 19–21.
[6] Wickremasinghe B. Cloudanalyst: a cloudsim-based tool for modelling and analysis of large scale cloud computing environments. MEDC Project Report 2009;22(6):433–659.
[7] Nine Z, SQ M, Azad M, Kalam A, Abdullah S, Rahman RM. Fuzzy logic based dynamic load balancing in virtualized data centers. In: Proceedings of 2013 IEEE International Conference on Fuzzy Systems (FUZZ). IEEE; 2013. p. 1–7.
[8] Kokilavani T, Amalarethinam D. Load balanced min–min algorithm for static meta-ask scheduling in grid computing. Int J Comput Appl 2011;20(2).
[9] Mao Y, Chen X, Li X. Max–min task scheduling algorithm for load balance in cloud computing. In: Proceedings of International Conference on Computer Science and Information Technology. Springer; 2014. p. 457–65.
[10] Wang S-C, Yan K-Q, Liao W-P, Wang S-S. Towards a load balancing in a three-level cloud computing network. In: Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 1. IEEE; 2010. p. 108–13.
[11] Wickremasinghe B. Cloudanalyst: a cloudsim-based tool for modelling and analysis of large scale cloud computing environments. MEDC Project Report 2009;22(6):433–659.
[12] Randles M, Lamb D, Taleb Bendiab A. A comparative study into distributed load balancing algorithms for cloud computing. In: Proceedings of the IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA). IEEE; 2010. p. 551–6.
[13] Chen S-L, Chen Y-Y, Kuo S-H. Design of cloud load balance architecture for web servers. Appl Mech Mater May. 2015;764-765:877–80.
[14] Liao W-H, Shih K-P, Wu W-C. A grid-based dynamic load balancing approach for data-centric storage in wireless sensor networks. Original Res Article Comput Electrical Eng January 2010;36(1):19–30.
[15] Semchedine F, Bouallouche-Medjkoune L, Tamert M, Mahfoud F, Aïssani D. Load balancing mechanism for data-centric routing in wireless sensor networks. Comput. Electrical Eng. January 2015;41:395–406.
[16] Yin Fei, Jiang C, Deng R, Yuan J. Grid resource management policies for load-balancing and energy-saving by vacation queuing theory. Comput Electrical Eng. November 2009;35(6):966–79.
[17] Wen Yean-Fu, Shen Jui-Chang. Load-balancing metrics: comparison for infrastructure-based wireless networks. Comput Electrical Eng February 2014;40(Issue 2):730–53.
[18] Acosta A, Blanco V, Almeida F. Dynamic load balancing on heterogeneous multi-GPU systems. Comput Electrical Eng November 2013;39(Issue 8):2591–602.
[19] Ajit M, Vidya G. VM level load balancing in cloud environment. Proceedings of the 4th International Conference on Computing, Communications and Networking Technologies (ICCCNT); 4-6 July 2013. p. 1–5.
[20] Sahu Y, Pateriya RK, Gupta RK. Cloud server optimization with load balancing and green computing techniques using dynamic compare and balance algorithm. In: Proceedings of the 5th International Conference on Computational Intelligence and Communication Networks (CICN); 27-29 Sept. 2013. p. 527–31.

Chen Shang-Liang received the PhD degree in Mechanical Engineering from University of Liverpool. He is currently a Professor with the Institute of Manufacturing and Information Systems at National Cheng-Kung University, Taiwan. He has published more than 70 papers in international peer reviewed journals. His research interests are in the areas of information and mechatronic integration, intelligent remote Monitoring System, PC-based multi-axis controller design, and CAD/CAM.

Yun-Yao Chen received the Ph.D. degree in Institute of Manufacturing Information and Systems, National Cheng Kung University, Taiwan. His current research interests include Computer Integrated Manufacturing, Cloud-based Applications and Internet of Things. He has published more than 60 journal & conference research papers. He is currently a computer integrated manufacturing supervisor of Taiwan Semiconductor Manufacturing Company (TSMC).

Suang-Hong Kuo received the master degree in Institute of Manufacturing Information and Systems, National Cheng Kung University, Taiwan.