International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 2, April 2022, pp. 2087~2096
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i2.pp2087-2096

Simulation and performance assessment of a modified throttled load balancing algorithm in cloud computing environment

Noha G. Elnagar[1], Ghada F. Elkabbany[2], Amr A. Al-Awamry[1], Mohamed B. Abdelhalim[3]
[1]Electrical Department, Benha Faculty of Engineering, Benha University, Benha, Egypt.
[2]Informatics Research Department, Electronics Research Institute, Cairo, Egypt
[3]College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport, Cairo, Egypt

Article Info

Article history:

Received Feb 20, 2021
Revised Aug 3, 2021
Accepted Sep 1, 2021

Keywords:

Cloud computing
CloudAnalyst
Load balancing
Modified throttled load balancer
Task allocation

ABSTRACT

Load balancing is crucial to ensure scalability, reliability, minimize response time, and processing time and maximize resource utilization in cloud computing. However, the load fluctuation accompanied with the distribution of a huge number of requests among a set of virtual machines (VMs) is challenging and needs effective and practical load balancers. In this work, a two listed throttled load balancer (TLT-LB) algorithm is proposed and further simulated using the CloudAnalyst simulator. The TLT-LB algorithm is based on the modification of the conventional throttled load balancer (TLB) algorithm to improve the distribution of the tasks between different VMs. The performance of the TLT-LB algorithm compared to the TLB, round robin (RR), and active monitoring load balancer (AMLB) algorithms has been evaluated using two different configurations. Interestingly, the TLT-LB significantly balances the load between the VMs by reducing the loading gap between the heaviest loaded and the lightest loaded VMs to be 6.45% compared to 68.55% for the TLB and AMLB algorithms. Furthermore, the TLT-LB algorithm considerably reduces the average response time and processing time compared to the TLB, RR, and AMLB algorithms.

Corresponding Author:

Noha G. Elnagar
Electrical Department, Benha Faculty of Engineering, Benha University
Benha, Egypt
Email: eng.noha17@gmail.com

1. INTRODUCTION

Cloud computing has emerged as a durable and evolving paradigm that offers massive opportunities for the information and communication technology industry. It is a powerful distributed computing platform for several indispensable applications including financial accounting feature [1], health care systems [2], meteorological prediction [3], networking, file storage and sharing as well as data collections [4], just to name a few. These potential applications of cloud computing are due to its several merits, such as resource maximization, customization, fast response to business needs, scalability, elasticity, off-site access, and cost-saving compared to hardware systems [5].

The cloud computing environment comprises three main layers, namely infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) [6]. The basic layer of cloud computing is the infrastructure layer, where the cloud offers shared hardware needed for the users. In IaaS, the virtualization software controls the access of users to partition and multiplex physical resources such as CPU, memory, I/O, storage, and network resources [7]. However, access of several thousands of users to a cloud data center (DC) leads to overloading, power consumption, and longer response time that could end up with a system failure.

Load imbalancing is one of the major issues that restrict the development of cloud computing technology and still need to be addressed. Consequently, designing and applying a practical load balancing system to manage the workloads is of particular interest. An efficient load balancing algorithm is characterized by a decrease in the loading gap between the heaviest loaded and the lightest loaded virtual machines (VMs), with enhanced resource utilization [8]. Moreover, it reduces the average required processing time and response time, which is essential for performing the received tasks in cloud DCs [9]. In this respect, several load-balancers have been proposed considering the aforementioned aspects [10]–[22]. Among the algorithms, round robin (RR), active monitoring load balancer (AMLB), throttled load balancer (TLB) are common load balancing algorithms in large-scale cloud environment using cloud simulator [23]–[29].

Load balancing algorithms can be categorized into static and dynamic algorithms [24]. Static algorithms such as the RR algorithm are typically suitable for a stable and homogeneous environment [30], [31]. On the other hand, dynamic algorithms such as AMLB and TLB algorithms are more flexible and can easily offer run time changes ability [26], [31]. The RR algorithm works in a circular arrangement, whereas all the VMs are employed, and the load is equally distributed among all servers depending on the time interval. The main issue of the RR algorithm is that it directly assigns the task to the VM in sequences without checking the capacity of the VM. Furthermore, it does not consider the state of the VM (available or busy) [32], [33]. In the AMLB algorithm, the number of requests assigned to each VM is monitored. When a new task arrives, it will be allocated to the VM with the least number of tasks. In case of more than one VM is available, the first recognized VM is selected, and AMLB returns the VM id to the data center controller (DCC) [34], [35]. Besides, TLB is a dynamic algorithm in nature, which assigns all received tasks to the available VMs. It determines the suitable VM for assigning a specific task by considering the state of the VM (available or busy) stated in the allocation table. When the number of requests is larger than the number of the available VM, the requests will be waiting until the VM becomes available [24], [25], [27], [36]. The limitation of TLB is that it checks for the available VM in the allocation table starting from the first VM every time, and it does not continue from the last allocated VM. Moreover, it does not count the number of the assigned tasks to each VM [34]. Based on these observations, the conventional algorithms have some limitations which produce less-effective load balancing. Therefore, the potential modification of these algorithms, particularly the TLB algorithm would be superior to develop an applicable algorithm for efficient load-balancing in cloud computing environment.

In this work, a two listed throttled algorithm (TLT-LB) is proposed and simulated. The TLT-LB algorithm aims at distributing the delivered tasks among the available VMs in an effective way taking into account the average processing time and response time. The suggested TLT-LB algorithm is based on the modification of the TLB algorithm by classifying the list of the VMs into two lists, namely available VMs and busy VMs. Consequently, when the available VM receives a specific task, its status turned into busy, and it is moved to the end of the busy VMs list. After performing the required task, the state of the busy VM switched to be available at the end of the available VMs list. CloudAnalyst simulation tool was employed to simulate the TLT-LB algorithm using two configurations based on different user bases to evaluate the average Processing time and Response time. Furthermore, a comparative study between the present TLT-LB algorithm and the TLB, RR, and AMLB algorithms is performed in CloudAnalyst considering the main performance metrics. The results reveal that the TLT-LB algorithm remarkably improved the load distribution among the VMs and minimized the average processing time and response time compared to the TLB, RR, and AMLB.

The remaining of the paper is organized as follows; section 2 summarizes a selection of the recent related studies to illustrate the main concepts of cloud load balancing and its categories. Section 3 describes in detail the proposed TLT-LB algorithm to be easily reproduced. Section 4 shows the experimental part for the simulation of the proposed algorithm on the CloudAnalyst platform. Section 5 displays the results and discussions, providing a clear overview regarding the merits of using the proposed algorithm over the TLB, RR, and AMLB algorithms. Finally, in Section 6 the conclusions are demonstrated.

2. RELATED WORK

Load balancing in cloud computing systems has been extensively studied over the last decades [35], [37]. In this section, some recent research studies on cloud load balancing are highlighted. Sakthivelmurugan *et al.* [38] introduced an algorithm named hospitality load balancing (HLB) algorithm. In the beginning, all VMs are available, and there is an index table that contains different VMs and their status BUSY/AVAILABLE. When a new task arrives, the capacity and load of all VMs are calculated. Then, it has to check the system, whether it is balanced or not. If the load is larger than the maximum capacity of VMs, the VMs are grouped into underloaded, overloaded, and balanced VM. To solve the problem of load imbalance, the HLB algorithm finds the underloaded VM and overloaded VM as well as sorting them in ascending/descending order, and then the tasks are migrated from the overloaded to the underloaded VMs. The migrated task must be assigned to the underloaded VM with the shortest path. Although this algorithm helps to achieve low response time, low make span, low task migration time, and handle heterogeneous host, it does not consider the status of the VM before task migration.

Mirobi and Arockiam [39] proposed an enhanced throttled load balancer (ETLB) that allocated the workloads uniformly to all VMs. By using the threshold value, the load balancer classified the VMs to overloaded, balanced, and under-loaded VMs. In the case of an overloaded VM is found, the ETLB searches for an appropriate underloaded VM and automatically sends the task to it, thereby balancing the load on VMs. This balancer determines the load of each VM after assigning the task to a VM and then migrates the tasks between overloaded and underloaded VMs. This may lead to a decrease in system efficiency and cause a delay in achieving the tasks.

Shetty and Shetty [40] suggested a modified central load balancer (MCLB) algorithm which has a table that comprises the ids, status, and priority of the VMs. This algorithm finds the VM with the highest priority and which is available for assigning the arrived task. In case of all VMs are busy, the arrived task is assigned to the first available VM without considering the priority of this VM, which means that the sequence of the algorithm will not be applicable.

Ghosh and Banerjee [41] suggested a priority-based modified throttled algorithm (PMTA). The idea of this algorithm depends on the TLB, which assigns the submitted tasks directly to the available VMs. In case of all VMs are busy, the algorithm compares the priority of the new received task to the executing task. A Switching queue was proposed to hold the task which has been removed temporarily from the VM due to the arrival of a higher priority task. The waiting task resumed the execution after completion of the higher priority task. This algorithm suffers from task delay, where some of the paused tasks could be lost or delayed because of their low priority.

Domanal and Reddy [42] proposed a modified throttled algorithm that assigned the incoming tasks uniformly to a set of VMs and improved the response time. In this algorithm, the VMs are indexed in a table, with status either available or busy similar to the standard TLB algorithm. An available VM is selected for the request, and its id is returned to the DC. For processing the following request, the VM that is found next to the assigned VM in the table is selected. The choice of the selected VM is depending on the status of the VM without parsing the index table from the beginning every time. This study focused on minimizing response time without considering the processing time. These studies reveal the influential role of load balancers to boost the performance of the cloud computing environment. However, some of these mentioned algorithms do not consider the requirements and priorities of the user. For example, some of these algorithms do not monitor the resource load during task allocation, which can lead to defeat the task execution and increase the processing time and response time. The main objective of this work is to enhance the TLB algorithm by decreasing both the processing time and response time. Besides, it effectively improves the load distribution among the VMs. In the next section, the proposed TLT-LB is illustrated.

3. RESEARCH METHOD

3.1. Proposed TLT-LB algorithm

The TLB algorithm has several advantages such as simplicity, dynamicity, and effectiveness in terms of response time and processing time [25], [27], [34] However, it resolves the VM list from the first VM every time, resulting in unequally load distribution. In this paper, a Two Listed Throttled Load Balancing TLT-LB algorithm is introduced. TLT-LB distributes the required tasks among the VMs effectively, for improving the response time and processing time compared to the conventional TLB, RR and AMLB algorithms. In the TLT-LB algorithm, the VMs are categorized into two lists, namely "Available VMs list" and "Busy VMs list". Accordingly, when an available VM delivers a specific task; its status turned into busy and moved to the end of the "Busy VMs list". After execution, the status of the busy VM switched to be available and moved to the end of the "Available VMs list". In the TLT-LB algorithm, the DCC receives all requests from different regions and forwards the tasks to TLT-VmLoadBalancer. Then, TLT-VmLoadBalancer allocates the tasks to the available VMs. When a new task arrives, the TLT-VmLoadBalancer maintains the "Available VMs list" and returns the "id" of the first available VM to the DCC. Then, the TLT-VmLoadBalancer assigns the task to the selected VM. If all VMs are busy, the received tasks will be queued, and the DCC waits for the first available VM to assign the waited task(s). Then, the allocated VM is removed from the "Available VMs list" and moved to the end of the "Busy VMs list". The basic methodology and steps of the proposed algorithm are shown in Figure 1(a) and (b).

3.2. Simulation

CloudSim is a toolkit that can be used for modeling, simulation, and scheduling of a large-scaled cloud platform [43], [44]. CloudAnalyst simulation packages are developed on CloudSim Toolkit-3.0.3 architecture. The main components of CloudAnalyst are region, internet, user base, InternetCloudlet, GUI, Cloud Service Broker, VmLoadBalancer, and DCC. These components are explained in Figure 2. In this work, CloudAnalyst is employed to analyze the proposed TLT-LB, TLB, RR, and AMLB algorithms because of its graphical user interface and high level of visualization [20], [35], [45]. CloudAnalyst package is configured using Eclipse IDE for Java Developer 2.0.1 and JDK 1.8.0 on Windows 10 operating system. The model of the system comprises one DC which has several homogeneous VMs ‘m’ (VM_0, VM_1, VM_2 ....VM_{m-1}) with 1000 MB Image size, 512 MB memory, and 1000 MB bandwidth, as well as the number of independent tasks ‘n’ (T_1, T_2, T_3......T_n). The selected parameters for the proposed TLT-LB algorithm using the two configurations are user base, region, peak hour start, peak hour end, and average users at peak and off-peak hours, as shown in Table 1 and Table 2, respectively. The DC configurations are X86 architecture, Linux operating system, xen virtual machine manager (VMM). The DC contains two physical hardware units. Each physical unit has the following configuration: 204800 RAM (MB), 100 storage space (TB), 1000000 available bandwidths, 4 processors that have a capacity power of 10000 MIPS, and a TIME_SHARED VM scheduling policy. The main influential metrics including load-distribution, response time, processing time, and the total cost of total VM cost and total data transfer cost are assessed for the proposed TLT-LB algorithm compared to TLB, RR, and AMLB algorithms using two configurations. Figures 3 illustrate the snapshot of advanced configuration in CloudAnalyst. Moreover, Figure 4 presents a snapshot of the regions used in the simulation.

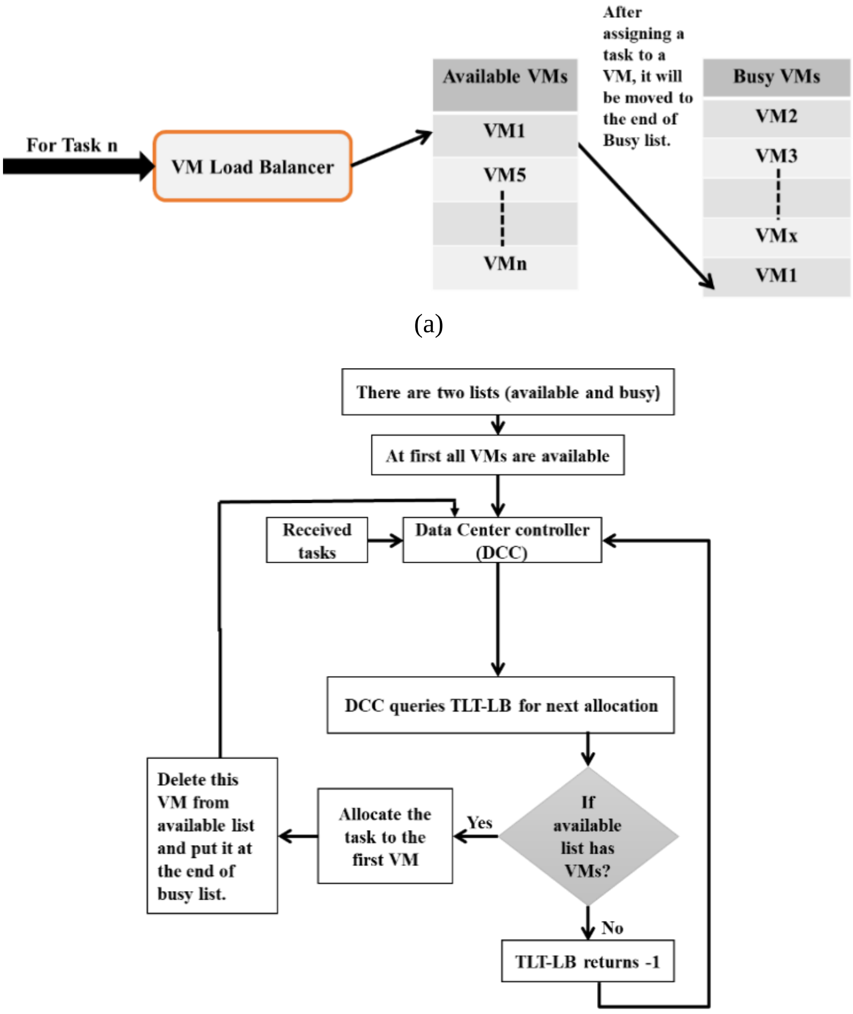


Figure 1. The TLT-LB algorithm shown as (a) TLT-LB block diagram and (b) TLT-LB workflow

The response time, processing time, load-distribution, and the total cost are assessed according to the following (1)-(4) [20]. The response time is the time required to respond to a user’s request for service [24] and can be calculated using (1):

$$Response\ time\ = T_{finish}\ -\ T_{arrival}\ +\ T_{delay} \tag{1}$$

where T_{finish} is the finish time of user request, $T_{arrival}$ is the arrival time of user request and T_{delay} is the transmission delay. T_{delay} can be calculated as:

$$T_{delay} =\ T_{latency} + T_{transfer}$$

where $T_{latency}$ is the network latency which is the delay time before the beginning of data transfer and $T_{transfer}$ is the time taken to transfer the amount of data of a single request (D) from the source location to the destination location. $T_{latency}$ is taken from the latency matrix (after applying Poisson distribution). $T_{transfer}$ can be calculated as follow:

$$T_{transfer} =\ D\ /\ Bw_{peruser}\ and\ Bw_{peruser} = Bw_{total}\ /\ N;$$

where $Bw_{peruser}$ and Bw_{total} are the bandwidth per user and the total available bandwidth (held in the InternetCharacteristics class in CloudAnalyst) respectively, while N is the number of user requests currently in transmission. The internet characteristics also keeps track of the number of user requests in-flight between two regions for the value of N.

Besides, the average processing time is the time required by the DC to process all the received tasks from the users. It is calculated using (2) [46]:

$$Processing\ time\ = L_{VMs}/C_{VMs} \tag{2}$$

where L_{VMs} is the sum of the load on each VM in the DC and C_{VMs} is the sum of the capacity of each VM in DC. The following formula is used to calculate the capacity of individual VM:

$$C_{VM} = p * q$$

where C_{VM} is the capacity for only one VM, P is the processing speed of the processor (CPU) in million instructions per second and q=number of busy CPU.

The load distribution (%) is calculated according to (3).

$$Load\ distribution\ \% = \frac{Number\ of\ tasks\ on\ VMn}{Total\ number\ of\ tasks\ on\ all\ VMs} \times 100 \tag{3}$$

The total cost is a very important parameter in cloud computing as it is paid on a pay-per-use basis, which depends on the percentage of resource utilization [24], [47]. The total cost of processing a task can be calculated using (4).

$$Total\ Cost = Transmission\ cost + Processing\ cost \tag{4}$$

Table 1. Simulation parameters of the first configuration

User Base	Region	Peak hour start (GMT)	Peak hour end (GMT)	Avg peak users	Avg off-peak users
1	0-N. America	3	9	1000	100
2	1-S. America	3	9	1000	100
3	2-Europe	3	9	1000	100
4	3-Asia	3	9	1000	100
5	4-Africa	3	9	1000	100
6	5-Oceania	3	9	1000	100

Table 2. Simulation parameters of the second configuration

User Base	Region	Peak hour start (GMT)	Peak hour end (GMT)	Avg peak users	Avg off-peak users
1	0-N. America	1	4	235000	23500
2	1-S. America	4	6	225000	22500
3	2-Europe	18	21	535000	53500
4	3-Asia	8	12	755000	75500
5	4-Africa	21	24	100000	10000
6	5-Oceania	6	8	170000	17000

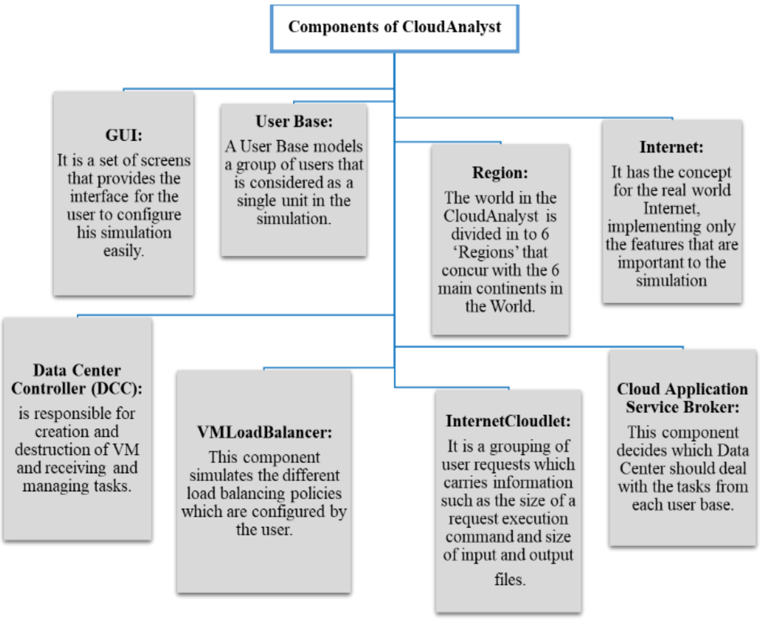


Figure 2. Main components of CloudAnalyst

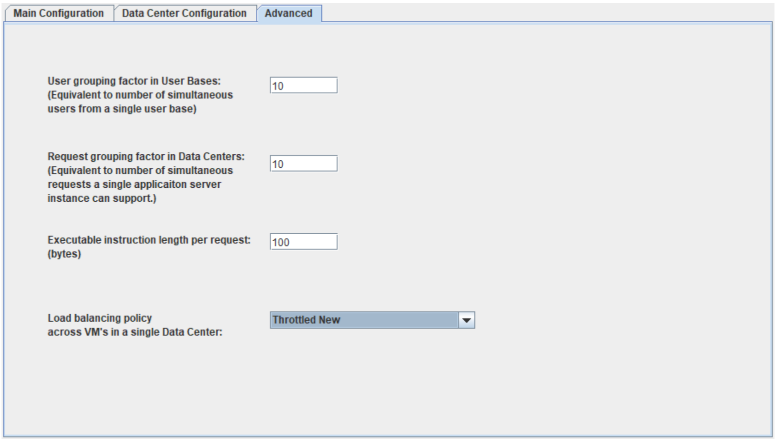


Figure 3. Snapshot of advanced configuration in CloudAnalyst

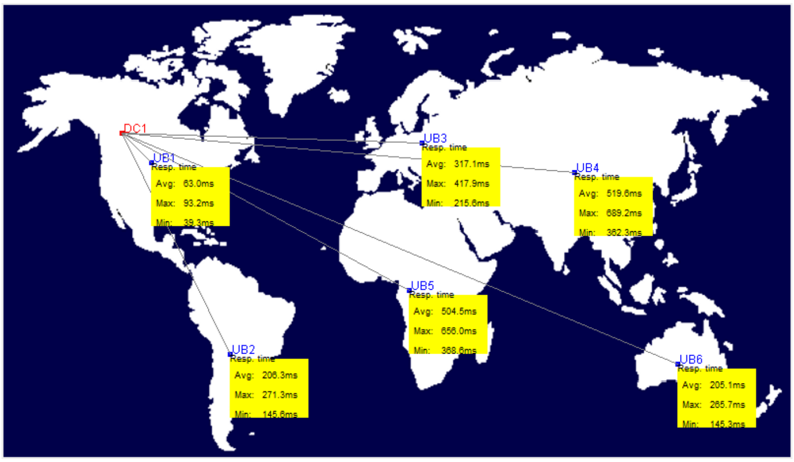


Figure 4. Snapshot of the regions used in the simulation

4. RESULTS AND PERFORMANCE EVALUATION

In this section, the simulation results of the TLT-LB algorithm using both the first and second configurations are presented to evaluate the performance of the proposed algorithm. The load distribution (%) for the TLT-LB algorithm compared to TLB and AMLB algorithms are presented in Figure 5. Interestingly, the load among the VMs is significantly distributed using the TLT-LB algorithm compared to TLB, and AMLB algorithms. In particular, the difference in the load distribution between the first VM and the last VM using the TLT-LB algorithm is in the range of 3-7%. In comparison, the load gap between the first VM and the last VM for TLB, and AMLB algorithms is more than 68%, which indicates that some VMs are over-loaded (VM_0 and VM_1), and the others are under-loaded (VM_3 to VM_9). These results reveal the applicability and high-performance of the TLT-LB algorithm, which decreases the load gap between the first and the last VM from 68.55% to 6.446% compared to the conventional TLB, and AMLB algorithms.

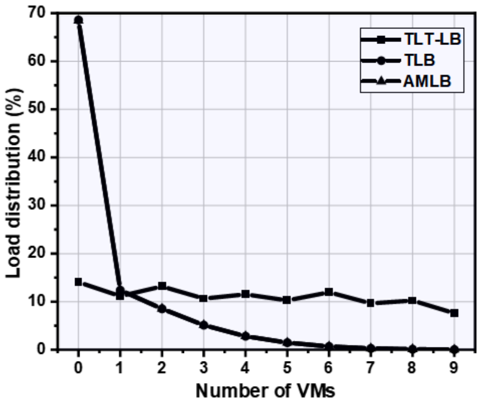
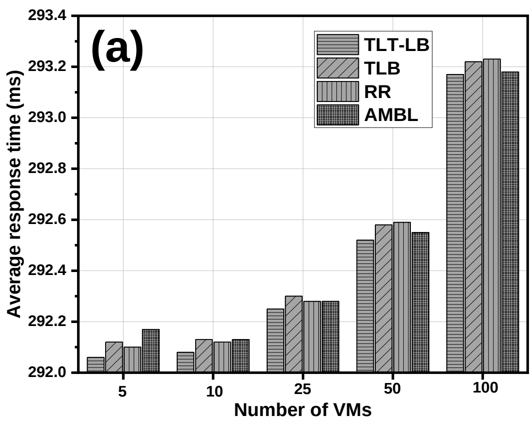
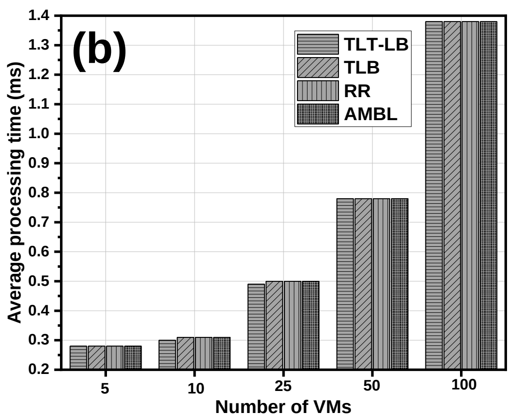


Figure 5. The load distribution (%) among the VMs after simulation of TLT-LB, TLB, and AMLB

The average response time and the average data center processing time are analyzed in milliseconds for the TLT-LB, TLB, RR, and AMLB algorithms. The results for the first configuration are shown in Figure 6(a) and 6(b). Generally, the average response time increases when the number of VMs increases for all the simulated algorithms. Besides, the response time decreases using the TLT-LB algorithm in contrast to the other algorithms for different numbers of VMs due to reducing the loading gap between the heaviest loaded and the lightest loaded VMs. However, there are no obvious changes in the processing time for the TLT-LB, TLB, RR, and AMLB algorithms. Therefore, the simulation conditions are changed by increasing the number of users to get more insights about the performance of the TLT-LB algorithm in terms of the average processing time related to TLB, RR, and AMBL algorithms.



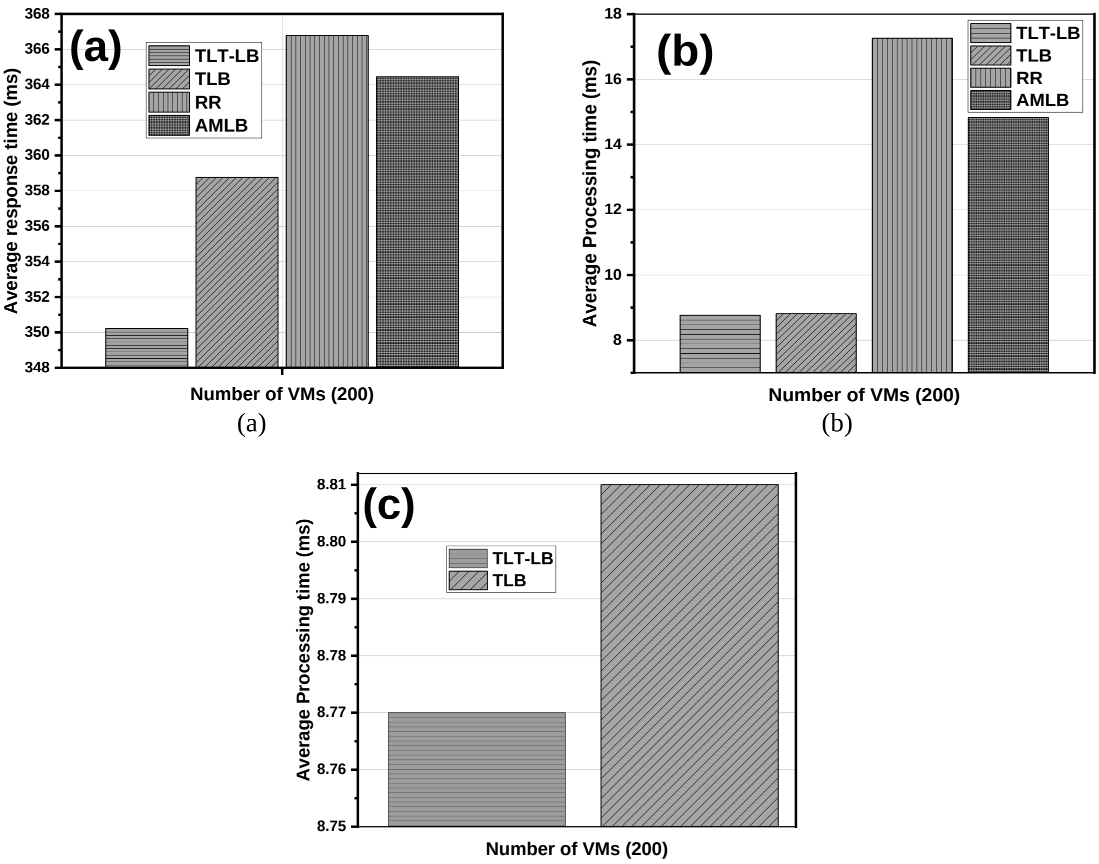
(a)



(b)

Figure 6. The result of different algorithms using the first configuration in (a) the response time and (b) processing time

Figure 7 illustrates the response time and processing time for the second configuration using 200 VMs. The response time and processing time considerably decrease for the TLT-LB algorithm compared to the other algorithms. As shown in Figure 7(a), the response time after using the TLT-LB algorithm decreases by 2.380%, 4.515%, and 2.164% compared to the TLB, RR, and AMLB algorithms respectively. In addition, Figures 7(b), (c) show that the processing time for the TLT-LB algorithm sustainably minimizes by 0.454%, 49.159%, and 40.462% in comparison to TLB, RR, and AMLB algorithms respectively. Figure 7(c) illustrates the difference between the TLT-LB and TLB algorithms in terms of processing time. The response time and processing time decrease using the second configuration rather than the first one. These results reveal that optimizing the conditions and selecting the proper parameters is a crucial step to evaluate the performance of different algorithms. Finally, the estimated total cost for the TLT-LB algorithm is $147.93, which equals to its counterpart for the TLB, RR, and AMLB algorithms.



(a) (b)

(c)

Figure 7. The result of different algorithms using the second configuration in (a) the response time and (b), (c) processing time

5. CONCLUSION

The efficiency of task allocation in the cloud environment depends on the efficiency of the load balancing algorithm. Therefore, using an adaptable load balancer for a cloud computing environment is a critical step to achieve maximum utilization of resources. In this respect, a TLT-LB algorithm is proposed to allocate the workload dynamically and effectively among the VMs. Two configurations employing different simulation parameters (user bases) are applied to assess the performance of the TLT-LB algorithm concerning the average response time, processing time, and cost. The performance of the proposed TLT-LB algorithm is compared with three well-known algorithms, namely RR, TLB, and AMLB. The CloudAnalyst package which is developed on CloudSim Toolkit-3.0.3 architecture is employed for the simulation process. Results indicate the applicability of the proposed TLT-LB algorithm to distribute the tasks among the VMs in an effective way since the load difference between the VMs is in the range of 3-7%. Moreover, it significantly reduces the average response time and processing time, which leads to an increase in the throughput of the system when comparing to the conventional algorithms. These findings show that the proposed TLT-LB is more practical than TLB, RR, and AMLB algorithms for a stable and effective cloud computing environment. In the future, we plan to extend this work considering more factors for utilizing the proposed TLT-LB algorithm in healthcare systems.

REFERENCES

[1] O. Dimitriu and M. Matei, "A new paradigm for accounting through cloud computing," *Procedia Economics and Finance*, vol. 15, pp. 840–846, 2014, doi: 10.1016/S2212-5671(14)00541-3.

[2] N. Sultan, "Making use of cloud computing for healthcare provision: Opportunities and challenges," *International Journal of Information Management*, vol. 34, no. 2, pp. 177–184, Apr. 2014, doi: 10.1016/j.ijinfomgt.2013.12.011.

[3] M. Yang *et al.*, "An efficient storage and service method for multi-source merging meteorological big data in cloud environment," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, Dec. 2019, Art. no. 241, doi: 10.1186/s13638-019-1576-0.

[4] O. Ali, A. Shrestha, J. Soar, and S. F. Wamba, "Cloud computing-enabled healthcare opportunities, issues, and applications: A systematic review," *International Journal of Information Management*, vol. 43, pp. 146–158, Dec. 2018, doi: 10.1016/j.ijinfomgt.2018.07.009.

[5] M. G. Avram, "Advantages and challenges of adopting cloud computing from an enterprise perspective," *Procedia Technology*, vol. 12, pp. 529–534, 2014, doi: 10.1016/j.protcy.2013.12.525.

[6] P. M. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, 2011. doi: 10.6028/NIST.SP.800-145.

[7] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," *IEEE Internet Computing*, vol. 13, no. 5, pp. 14–22, Sep. 2009, doi: 10.1109/MIC.2009.119.

[8] S. Mustafa, B. Nazir, A. Hayat, A. U. R. Khan, and S. A. Madani, "Resource management in cloud computing: Taxonomy, prospects, and challenges," *Computers & Electrical Engineering*, vol. 47, pp. 186–203, Oct. 2015, doi: 10.1016/j.compeleceng.2015.07.021.

[9] A. Subhi Abdalkafor, A. Abdalqahar Jihad, and E. Tariq Allawi, "A cloud computing scheduling and its evolutionary approaches," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 21, no. 1, pp. 489–496, Jan. 2021, doi: 10.11591/ijeecs.v21.i1.pp489-496.

[10] R. Mishra, "Ant colony Optimization: A Solution of Load balancing in Cloud," *International journal of Web & Semantic Technology*, vol. 3, no. 2, pp. 33–50, Apr. 2012, doi: 10.5121/ijwest.2012.3203.

[11] D. B. L.D. and P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Applied Soft Computing*, vol. 13, no. 5, pp. 2292–2303, May 2013, doi: 10.1016/j.asoc.2013.01.025.

[12] K. Dubey and S. C. Sharma, "An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework," *Journal of King Saud University - Computer and Information Sciences*, Nov. 2020, doi: 10.1016/j.jksuci.2020.11.001.

[13] D. Chaudhary and R. Singh Chhillar, "A New Load Balancing Technique for Virtual Machine Cloud Computing Environment," *International Journal of Computer Applications*, vol. 69, no. 23, pp. 37–40, May 2013, doi: 10.5120/12114-8498.

[14] M. Hijab and A. Damodaram, "Weighted randomized algorithms for efficient load balancing in distributed computing environments," *Materials Today: Proceedings*, vol. 33, pp. 3782–3786, 2020, doi: 10.1016/j.matpr.2020.06.216.

[15] F. Garcia-Carballeira, A. Calderon, and J. Carretero, "Enhancing the power of two choices load balancing algorithm using round robin policy," *Cluster Computing*, vol. 24, no. 2, pp. 611–624, Jun. 2021, doi: 10.1007/s10586-020-03139-6.

[16] S. R. Gundu, C. A. Panem, and A. Thimmapuram, "Real-Time Cloud-Based Load Balance Algorithms and an Analysis," *SN Computer Science*, vol. 1, no. 4, Jul. 2020, Art. no. 187, doi: 10.1007/s42979-020-00199-8.

[17] S. Ouhame and Y. Hadi, "Enhancement in resource allocation system for cloud environment using modified grey wolf technique," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 20, no. 3, pp. 1530–1537, Dec. 2020, doi: 10.11591/ijeecs.v20.i3.pp1530-1537.

[18] T. Kokilavani and D. I. George Amalarethinam, "Load balanced MinMin algorithm for static MetaTask scheduling in grid computing," *International Journal of Computer Applications*, vol. 20, no. 2, pp. 42–48, Apr. 2011, doi: 10.5120/2403-3197.

[19] B. Mondal, K. Dasgupta, and P. Dutta, "Load balancing in cloud computing using stochastic hill climbing-a soft computing approach," *Procedia Technology*, vol. 4, pp. 783–789, 2012, doi: 10.1016/j.protcy.2012.05.128.

[20] B. Wickremasinghe and Others, "Cloudanalyst: a cloudsim-based tool for modelling and analysis of large scale cloud computing environments," *MEDC project report*, vol. 22, no. 6. pp. 433–659, 2009.

[21] J. McCall, "Genetic algorithms for modelling and optimisation," *Journal of Computational and Applied Mathematics*, vol. 184, no. 1, pp. 205–222, Dec. 2005, doi: 10.1016/j.cam.2004.07.034.

[22] A. K. Sharma, K. Upreti, and B. Vargis, "Experimental performance analysis of load balancing of tasks using honey bee inspired algorithm for resource allocation in cloud environment," *Materials Today: Proceedings*, Oct. 2020, doi: 10.1016/j.matpr.2020.09.359.

[23] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, and S. Musa, "A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach," *IEEE Access*, vol. 8, pp. 130500–130526, 2020, doi: 10.1109/ACCESS.2020.3009184.

[24] S. K. Mishra, B. Sahoo, and P. P. Parida, "Load balancing in cloud computing: A big picture," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, Feb. 2020, doi: 10.1016/j.jksuci.2018.01.003.

[25] A. M. Manasrah, A. Aldomi, and B. B. Gupta, "An optimized service broker routing policy based on differential evolution algorithm in fog/cloud environment," *Cluster Computing*, vol. 22, no. S1, pp. 1639–1653, Jan. 2019, doi: 10.1007/s10586-017-1559-z.

[26] A. Jyoti, M. Shrimali, S. Tiwari, and H. P. Singh, "Cloud computing using load balancing and service broker policy for IT service: a taxonomy and survey," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 4785–4814, Nov. 2020, doi: 10.1007/s12652-020-01747-z.

[27] Z. Benlalia, K. Abouelmehdi, A. Beni-hssane, and A. Ezzati, "Comparing load balancing algorithms for web application in cloud environment," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 17, no. 2, Feb. 2020, Art. no. 1104, doi: 10.11591/ijeecs.v17.i2.pp1104-1108.

[28] A. A. AlKhatib, T. Sawalha, and S. AlZu'bi, "Load balancing techniques in software-defined cloud computing: an overview," in *2020 Seventh International Conference on Software Defined Systems (SDS)*, Apr. 2020, pp. 240–244, doi: 10.1109/SDS49854.2020.9143874.

[29] S. Afzal and G. Kavitha, “Load balancing in cloud computing – A hierarchical taxonomical classification,” *Journal of Cloud Computing*, vol. 8, no. 1, Dec. 2019, Art. no. 22, doi: 10.1186/s13677-019-0146-7.

[30] S.-L. Chen, Y.-Y. Chen, and S.-H. Kuo, “CLB: A novel load balancing architecture and algorithm for cloud services,” *Computers & Electrical Engineering*, vol. 58, pp. 154–160, Feb. 2017, doi: 10.1016/j.compeleceng.2016.01.029.

[31] N. R. Tadapaneni, “A survey of various load balancing algorithms in cloud computing,” *International Journal for Science and Advance Research in Technology*, vol. 6, no. 4, pp. 484–487, 2020.

[32] R. V. Rasmussen and M. A. Trick, “Round robin scheduling – a survey,” *European Journal of Operational Research*, vol. 188, no. 3, pp. 617–636, Aug. 2008, doi: 10.1016/j.ejor.2007.05.046.

[33] S. Banerjee, A. Roy, A. Chowdhury, R. Mutsuddy, R. Mandal, and U. Biswas, “An Approach Toward Amelioration of a New Cloudlet Allocation Strategy Using Cloudsim,” *Arabian Journal for Science and Engineering*, vol. 43, no. 2, pp. 879–902, Feb. 2018, doi: 10.1007/s13369-017-2781-y.

[34] S. Patel, R. Patel, H. Patel, and S. Vahora, “CloudAnalyst : A survey of load balancing policies,” *International Journal of Computer Applications*, vol. 117, no. 21, pp. 21–24, May 2015, doi: 10.5120/20679-3525.

[35] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, “CloudAnalyst: a cloudsim-based visual modeller for analysing cloud computing environments and applications,” in *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010, pp. 446–452, doi: 10.1109/AINA.2010.32.

[36] G. Ramadhan, T. W. Purboyo, R. Latuconsina, and A. R. Robin, “Experimental model for load balancing in cloud computing using throttled algorithm,” *International Journal of Applied Engineering Research*, vol. 13, no. 2, pp. 1139–1143, 2018.

[37] M. Xu, W. Tian, and R. Buyya, “A survey on load balancing algorithms for virtual machines placement in cloud computing,” *Concurrency and Computation: Practice and Experience*, vol. 29, no. 12, Jun. 2017, Art. no. e4123, doi: 10.1002/cpe.4123.

[38] V. Sakthivelmurugan, R. Vimala, and K. R. Aravind Britto, “Magnum opus of an efficient hospitality technique for load balancing in cloud environment,” *Concurrency and Computation: Practice and Experience*, vol. 31, no. 14, Jul. 2019, doi: 10.1002/cpe.5078.

[39] G. J. Mirobi and L. Arockiam, “Dynamic load balancing approach for minimizing the response time using an enhanced throttled load balancer in cloud computing,” in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Nov. 2019, pp. 570–575, doi: 10.1109/ICSSIT46314.2019.8987845.

[40] S. M. Shetty and S. Shetty, “Analysis of load balancing in cloud data centers,” *Journal of Ambient Intelligence and Humanized Computing*, Jan. 2019, doi: 10.1007/s12652-018-1106-7.

[41] S. Ghosh and C. Banerjee, “Priority based modified throttled algorithm in cloud computing,” in *2016 International Conference on Inventive Computation Technologies (ICICT)*, Aug. 2016, vol. 2016, pp. 1–6, doi: 10.1109/INVENTIVE.2016.7830175.

[42] S. G. Domanal and G. R. M. Reddy, “Load balancing in cloud computingusing modified throttled algorithm,” in *2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, Oct. 2013, pp. 1–5, doi: 10.1109/CCEM.2013.6684434.

[43] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, Jan. 2011, doi: 10.1002/spe.995.

[44] R. N. Calheiros, R. Ranjan, C. A. F. De Rose, and R. Buyya, “CloudSim: a novel framework for modeling and simulation of cloud computing infrastructures and services,” Mar. 2009, [Online]. Available: http://arxiv.org/abs/0903.2525.

[45] S. P. Singh, A. Sharma, and R. Kumar, “Analysis of load balancing algorithms using cloud analyst,” *International Journal of Grid and Distributed Computing*, vol. 9, no. 9, pp. 11–24, Sep. 2016, doi: 10.14257/ijgdc.2016.9.9.02.

[46] M. Kumar and S. C. Sharma, “Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing,” *Procedia Computer Science*, vol. 115, pp. 322–329, 2017, doi: 10.1016/j.procs.2017.09.141.

[47] N. A. Kofahi, T. Alsmadi, M. Barhoush, and M. A. Al-Shannaq, “Priority-based and optimized data center selection in cloud computing,” *Arabian Journal for Science and Engineering*, vol. 44, no. 11, pp. 9275–9290, Nov. 2019, doi: 10.1007/s13369-019-03845-3.