IJAMEC
e-ISSN: 2147-8228

INTERNATIONAL JOURNAL OF APPLIED MATHEMATICS ELECTRONICS AND COMPUTERS

www.dergipark.org.tr/ijamec

International Open Access

Volume 10
Issue 04

December, 2022

Research Article

Hybrid Load Balancing Policy to Optimize Resource Distribution and Response Time in Cloud Environment

Lencho Miesso Bokiye [a], ***İlker Ali Özkan*** [b]

[a] *The Graduate School of Natural and Applied Science, Selcuk University, 42250 Konya, Turkey*
[b] *Department of Computer Engineering, Faculty of Technology, Selcuk University, 42075 Konya, Turkey*

ARTICLE INFO

Article history:
Received 8 August 2022
Accepted 4 November 2022

Keywords:
Cloud Analyst
Cloud Computing
Load Balancing
Optimization
Resource Utilization
Response time

ABSTRACT

Load balancing and task scheduling are the main challenges in Cloud Computing. Existing load balancing algorithms have a drawback in considering the capacity of virtual machines while distributing loads among them. The proposed algorithm works toward solving existing issues, such as fair load distribution, avoiding underloading and overloading, and improving response time. It implements best practices of Throttled load balancing algorithm and Equally Shared Current Execution algorithm. Virtual machines are selected based on the ratio of their bandwidth and load allocation count. Requests are sent to a Virtual Machine with higher bandwidth and lower load allocation count. Proposed algorithm checks for the availability of VM based on their capacity. This process is performed by selecting two VMs and comparing their vmWeight capacity. The one with the least vmWeight is selected. CloudAnalyst is used for simulation, response time evaluation, and resource utilization evaluation. The simulation result of the proposed algorithm is compared with three well-known load-balancing algorithms. These are Round Robin, Throttled Load balancing algorithm, and Enhanced Active Monitoring. Load-balancing Proposed Algorithm selects VMs based on their Algorithm. The proposed algorithm has improved over other algorithms in load distribution, response time, and resource utilization. All virtual machines in the data centers are loaded with a relatively equal number of tasks according to their capacity. This resulted in fair resource sharing and load distribution.

1. Introduction

Cloud computing is a new technology that has gotten a lot of interest in both commercial and academic areas. It is a developing computing paradigm that has tilted all other entities in the digital industry, including the government and private sectors. Given the rising relevance of the cloud, finding ways to promote cloud services is a widespread issue for filling gaps in academic and commercial fields. As information technology and the internet have advanced rapidly, Computing resources have become more affordable, powerful, and widespread. This technological trend enables people to implement a new computing model called cloud computing. Resources are provided as general utility programs, and users can rent and release these utilities on demand through the internet [1]. Cloud computing is a current technological trend. Users can rent software, hardware, infrastructure, and computing resources per user [2].

According to a study from the University of California Berkeley, the core features of cloud computing are the illusion of unlimited computing power, the absence of an up-front contribution by cloud customers, and the freedom to pay for use as needed [3]. Cloud computing is a growing technological paradigm that uses virtualization technologies to respond to user requests from the Internet network and dynamic resource allocation depending on demand. Project coordination becomes a critical problem in effective resource selection in cloud computing [4]. Figure 1 shows the cloud computing components and a general overview.

* Corresponding author. E-mail address: lenchomiesso@gmail.com
DOI: https://doi.org/10.18100/ijamec.1158866

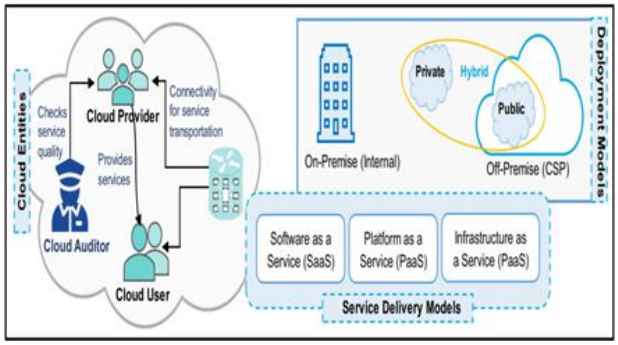

Figure 1. Cloud Computing overview [1]

Load Balancing: Load balancing is a process of evenly sharing the workload between all nodes of virtual machines. It is applied to get better service provisioning and resource utilization. The load balancer receives all incoming client requests and distributes them to data centers for balanced load distributions[2].

Load balancing refers to the right to move some part of a system request's execution to another separate system that performs it simultaneously. It distributes the load from a single server to various other computers. As there is less competition for energy and more machines handling the whole load, this reduces the amount of processing performed by the main receiving processor, allowing it to manage more requests and increase efficiency [3]. Figure 2. shows basic architecture of load balancing algorithm.

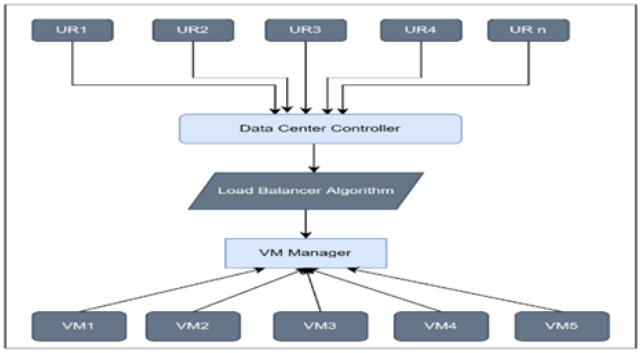

Figure 2. Load balancing algorithms architecture[4]

This article aims to design a Load Balancing algorithm for a cloud computing environment by bringing the best features of throttled load balancing algorithm and Equally shared Current Execution algorithm with better average response time and resource utilization. This focus on resource utilization and minimizing average response time. This study is an addition to research in cloud computing, especially in the load balancing environment, which is a big headache for cloud service providers. This is due to heavy network trafficking and continuous customer needs.

2. Literature Review

There is a lot of need for load balancers in cloud computing. Some of the reasons are resource utilization and response time maximization. To fulfill these needs, the system should have a load balancing algorithm to decide task allocation between the virtual machines [5]

Different load balancing algorithms have been designed and developed. Richhariya et al. proposed an algorithm that assigns time slices to different requests based on their priority. They proposed an algorithm that eliminates the drawbacks of implementing a simple round-robin architecture in cloud computing by introducing a concept of assigning different time slices to individual processes depending on their priorities [6]. In the throttled Algorithm, there is a table that contains Virtual Machines and their states. When VM is allocated to a specific task, a request is sent to the data center. Then the data center looks for the best virtual machine that matches the required task requirement [7]. In a compressive analysis of three LBA, i, e Round Robin, ESCE, and Throttled LBA studies, Bhatia et al. concluded that TLBA showed less response and minimum processing time. Even though it had a better response time, it had some issues [8].

Authors in [9] proposed a load balancing algorithm where the task allocation process passes through three layers. The work is received at the first tier and assigned to one of the second layer's service managers. Finally, the third layer divides the requests into subtasks, which speeds up the process. This method keeps every node engaged and working to fulfill user requests. However, processing the request hierarchically may be sluggish since it must pass through each layer of the framework.

In [10], a hybrid particle swarm optimization approach was developed for load balancing in centralized cloud-based multimedia storage. In this technique, the weight of each particle was determined using the multi-kernel support vector machine (MKL-SVM) and fuzzy simple additive weight (FSAW) techniques. Using three concurrent and dynamic approaches increases processing time and load balancing complexity.

3. Load Balancing Algorithms

3.1. Round Robin Algorithm

This Algorithm is quite simple to use nowadays. This Algorithm works by selecting one of the available virtual machines and assigning a job to that virtual machine. The arrangement of this process is circular [11]. The allocated Virtual machine rolls to the end of the list, and the previous second Algorithm becomes ready to take the next task. This Algorithm assigns tasks to every VM despite their capability. Figure 3 shows the diagram for the Round Robin algorithm.

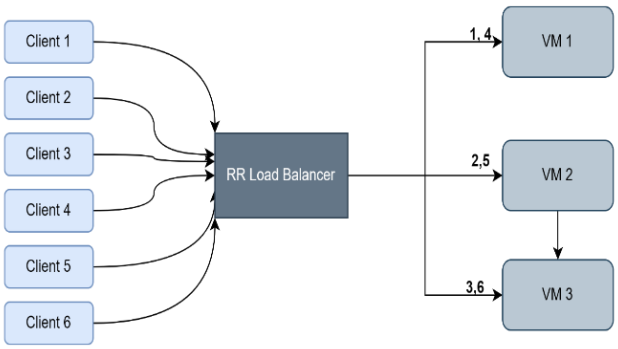

Figure 3. Round Robin Algorithm diagram[12]

This Algorithm gives weights to every VM according to its capacity. Then the loads are assigned according to the capability of the VM. This has improved the Round Robin algorithm to better performance despite still some drawbacks [13].

Advantages:

- If the jobs have equal processing time, then this Algorithm performs well.

Disadvantages:

- The Load balancer loads a task to a server without checking for the load of the work.
- Not efficient if the tasks do not have equal processing time.
- Not efficient when the nodes have different capabilities.

3.2. Equally Spread Current Execution Algorithm (ESCE)

In ESCE, the Algorithm works to allocate loads to every Virtual machine equally. The load balancer holds the index of virtual machines and the number of tasks assigned to them. At the time of task allocation, the load balancer scans for a virtual machine with the most negligible load. If there is more than one virtual machine, then the load balancer selects the first virtual machine and notify id of the virtual machine to the data center. The data center communicates the request to the VM identified by that id. Then the data center updates the virtual machines' information by incrementing the number of tasks allocated to that virtual machine. Upon task completion, the load balancer notifies the data center about the task completion. The data center updates the virtual machine information by decreasing the allocation count for that virtual machine. Figure 4 shows the diagram for the ESCE algorithm.

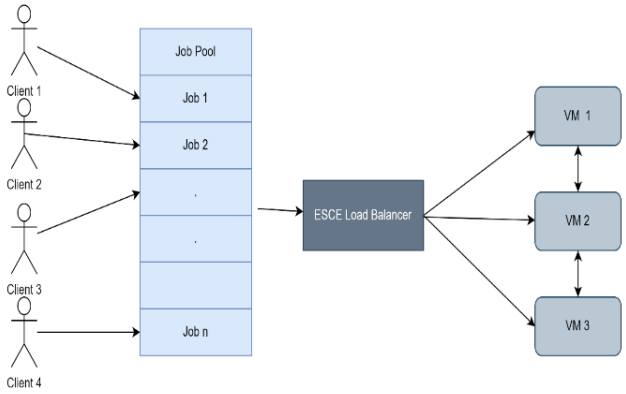

Figure 4. ESCE Algorithm diagram[14]

ESCE works on sharing an equal load between the virtual machines, but task allocation takes longer while the Algorithm looks for under-allocated virtual machines.

This Algorithm's resource utilization is lower when compared to other load-balancing algorithms.

3.3. Throttled Load Balancing Algorithm (TLB)

In the throttled Algorithm, there is a table that contains Virtual Machines and their states. When VM is allocated to a specific task, a request is sent to the data center. The data center looks for the best virtual machine that matches the task requirement [13], [15].

This Algorithm repeatedly scans for free virtual machines in the index list for every request from the user. This prolongs the time response time, and the overall process takes longer. It only checks for the availability of the virtual machine, and it does not check for the property and capability of the virtual machine. This causes overloading and underloading[16].

4. Proposed Methodology

This section outlines the problem statement that was derived from the research review. The proposed algorithm is grounded on the ESCE Load balancing algorithm and Throttled Load balancing algorithm. It works to grab the best features of these two algorithms and make improvements to the drawbacks of these algorithms. The fundamental drawback of these algorithms is as the following.

Throttled Load Balancing Algorithm (TLB***)*** checks for the availability of the virtual machine, and it does not check for the property and capability of the virtual machine. This causes overloading and underloading[16].

ESCE works on sharing an equal load between the virtual machines, but task allocation takes longer while the algorithm looks for under-allocated virtual machines. This Algorithm's resource utilization is lower when compared to other load-balancing algorithms. Table 1 shows some existing load balancing algorithms and their advantage. and drawbacks.

Table 1. Load balancing algorithms, advantages, and their drawback.

No.	LB Algorithms	Description and advantage	The main drawback
1	Throttled LBA [20].	Check the availability/Busy status of the VM	This algorithm checks for available VM in the list from the beginning and return the available VMs index list. This process takes longer if the first VMs are busy and the response time will be higher.
2	Modified throttled LBA[21].	When the request arrives, looking for an available Virtual Machine starts from the last appointed VM index, this reduces the time that is wasted looking for an available Virtual Machine.	Some VM states may be changed without receiving another task. In this case, the response time may be higher and the request may be sent to a Virtual Machine with a higher load.
3	Round-robin LBA [22]	This algorithm is simple and better in load distribution as it allocates requests circularly.	This algorithm's Response time is high as the tasks wait in the queue for a longer time.
4	Randomized weighted throttled Algorithm [19].	This algorithm better performs scanning of the next available VM index. It avoids scanning the VM index again and again from the first VM index to the last.	This algorithm's load distribution and response time are relatively not good. There is also a chance that two random numbers may be chosen.
5	Enhanced active monitoring [23].	This algorithm performs request allocation to the least loaded virtual machine and avoids allocating to VM that is loaded recently.	High overhead association response time is also maximum when compared to others.

This study aims to design and develop a load-balancing algorithm that minimizes overloading and underloading, which distributes the load across all virtual machines fairly and according to the capacity of the virtual machines. The capacity of the virtual machine is measured by its amount of bandwidth and the number of allocations allocated to a specific Virtual Machine. In the first round, all VM is available, and tasks are allocated directly to available virtual machines. In this study, we used the term vmWeight ratio to describe the weight of the Virtual Machines and the number of tasks allocated to Virtual machines. These two factors decide which Virtual machine to choose.

In Table 2, an example to calculate the vmWeight ratio is shown by taking sample values for allocation count and Bandwidth for n number virtual machines.

Table 2. Sample Hashmaps and their corresponding allocation count, bandwidth and vmWeight

		VM1	VM2	VM3	.	VMn
Hash Map 1	Index(vmId)	0	1	2		n
	allocationCount	40	50	35		.
Hash Map 2	Index(vmId)	0	1	2		n
	Bandwidth	10000	20000	10000		.
Hash Map 3	Index(vmId)	0	1	2		n
	vmWeight	250	400	285		.

Equation (1) is used to get the value of the vmWeight variable, which is used to make a decision to select a VM.

$$vmWeight = Bandwidth/allocationCount \quad (1)$$

vmWeight for VM1=10000/40=250,

vmWeight for VM2=20000/50=400

vmWeight for VM3=10000/35=285

According to the proposed algorithm, two VMs are randomly selected and their vmWeight is compared to each other, and the one with a higher vmWeight is selected for the task.

If VM2 and VM3 are randomly selected as a candidate to take the next task, their vmWeight is compared, and the one with higher vmWeight is selected, i.e., VM2 will be allocated with a task as its vmWeight is 400 and VM3 has a value of 285.

To maintain the minimum response time, two virtual Machines are selected randomly, their weight ratios are compared, and the VM with a higher Weight ratio is selected. If the weight ratio is equal, then from the properties of the Virtual machine, their bandwidth is compared to one another, and the one with the higher bandwidth is selected. If still, their bandwidth is equal, the first Virtual machine is selected. In the existing Load balancing algorithm, loads are allocated to Virtual machines without checking the capacity and nature of the Virtual machine.

Steps in the proposed virtual machines:

- HashMap holds the load and their respective

bandwidth of the Virtual Machines

- HashMap vmWeight holds the weight of the vmId as a key and its weight as a value. Weight is the ratio of the allocation task to bandwidth.
- vmAllocatiom HashMap holds the number of allocation counts of every Virtual Machine.
- When requested at first, vmWeight size is less than vmList

When the request arrives

- First, it checks the vmList hash map; if not empty, assign the request to the first VM
- If vmList is empty, It picks two random virtual machines from the vmWeight hash map and compares their vmWeight value, assigning them to a virtual machine with a higher vmWeight.
- If they are equal, it checks their bandwidth and assigns them to a virtual machine with greater bandwidth.

The following diagram shows the flowchart of the proposed Load balancing algorithm.

4.1. Response Time Calculation

The following Equation (2) calculates response time in load-balancing algorithms[17].

$$\text{Response Time} = FT - AT + DT \quad (2)$$

Where FT is the Task finish time, AT is the arrival time, and DT is the delay time.

Figure 5. shows the flowchart of the proposed algorithm. Decisions are made based on criteria in the flowchart.

4.2. Simulation Environment

Many toolkits can be used to model, design, and simulate cloud applications and study their behavior and efficiency. But to do the studies and measure the efficiencies, it is better to have something more visual and with more features than only having a toolkit. Real-time testing and simulations are challenging to accomplish due to a large number of concurrent users, the location of necessary components, and the location of data centers. Models of applications and infrastructures are used in simulation tests [18]. Even if cloud computing makes large-scale application installation easier and less expensive, it also introduces new challenges for developers. Because of the nature of cloud infrastructure, its deployment is in different geographical locations and the chosen locations can have an impact on the performance of its usage by users who are far from the data center[19], [20].

4.2.1. CloudSim:

CloudSim is a toolkit that is used to model, simulate and perform other experiments in cloud computing. The primary issue with CloudSim is that all work must be done

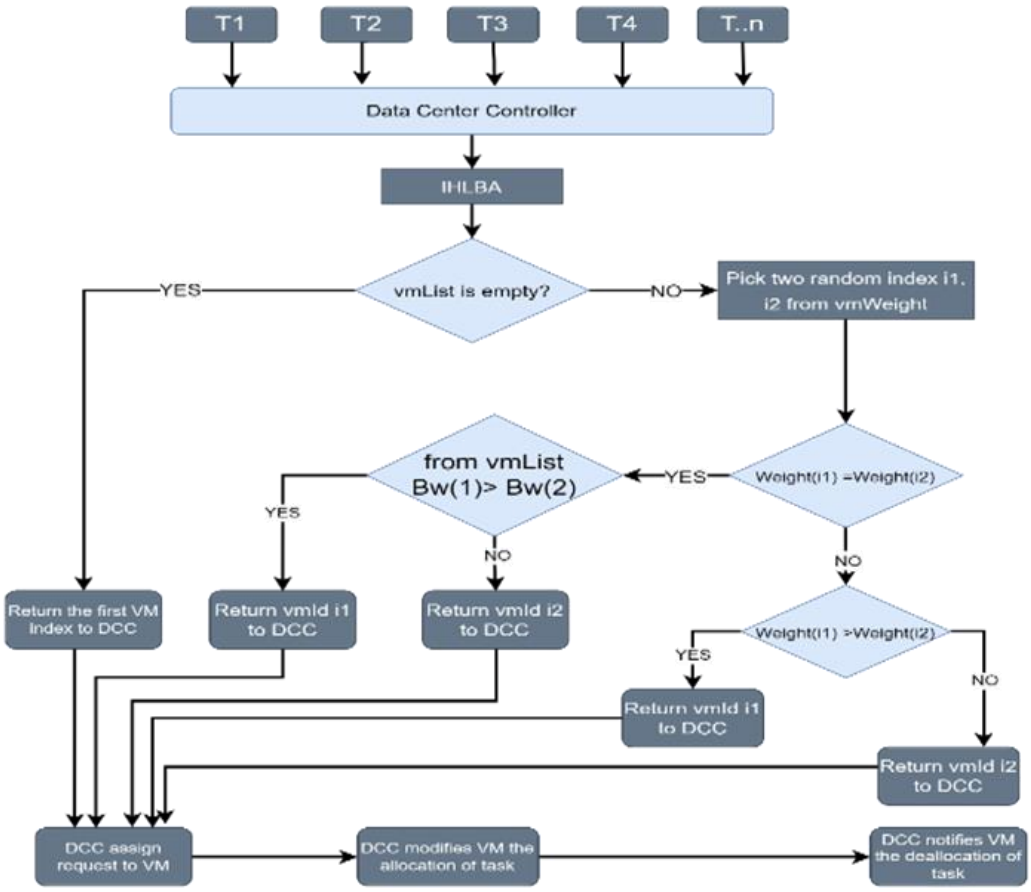

Figure 5. Flowchart of the proposed Algorithm

programmatically [1]. CloudSim employs Java, the most powerful object-oriented programming language available. CloudSim modules can be conveniently extended to meet the consumer's needs, thanks to the OOP functionality[21]. It can significantly reduce the requirement for and costs associated with computer facilities for performance evaluation and modeling of the research solution[22].

4.2.2. CloudAnalyst Simulator

The CloudAnalyst structure is built on CloudSim simulators. CloudAnalyst has a graphical user interface for instantly and conveniently configuring experiment parameters[23]. This enables CloudAnalyst better tool for the design, simulation, and evaluation of cloud environment applications as GUI based interface make it easy to compare and present simulation result rather than cloudsim.

The main features of CloudAnalyst are the following [20].

- Simple presentation of User Interface in Graphics (GUI)
- The ability to create a simulation that is highly configurable and flexible.
- Ability to support repeating experiments
- Output in graphical form.
- The use of consolidated infrastructure and the simplicity of which it can be extended.

Some essential CloudAnalyst components are used in modeling Cloud computing environments. They are namely User Base, Regions, service broker, data center controller, Internet characteristics configurations, Internet cloudlet, and VM Load balancer. Figure 6 shows the general user interface of cloud Analyst. In the GUI, every continent are shown as region one to region six.

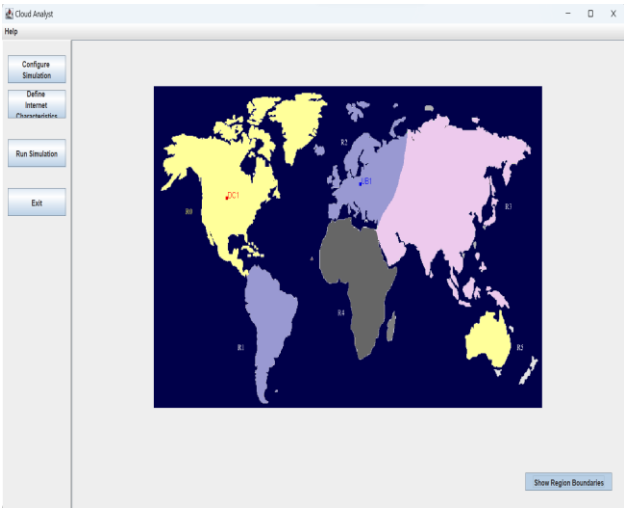

Figure 6. CloudAnalyst GUI interface [24]

5. Experiment Result and Discussion

Two experiments were done, including three load balancing algorithms and the proposed Algorithm. The first experiment focuses on evaluating the average response time of the algorithms. The second Algorithm compares the virtual machine resource utilization of the algorithms. There are two different configurations for both experiments.

5.1. Experiment 1 Evaluating resource allocation for large requests per user bases

This experiment focuses on resource utilization and fair load distribution between virtual machines in data centers with more user bases. The load distribution in the data centers considers the specification of the virtual machines, such as bandwidth, memory, and load allocation counts. The proposed Algorithm's response time is evaluated by comparing the results with other load-balancing algorithms.

5.1.1. Configuration

In this experiment, two data centers with four virtual machines and five virtual machines are configured. It is supposed to handle 25000 requests per user base. Its simulation duration is 60 minutes on average. In table 3, the configuration values for experiment one has shown. This configuration is based on previously proposed algorithms. In all regions, there is user base when the users can be connected to the data center. Table 3 shows the configuration of experiment one.

Table 3. Simulation configuration of experiment one.

Userbase	Region	Requests per user per hour	Data size per request	Peak hours (GMT)	Avg peak user	Avg off-peak users
UB1	5	25000	1000	3:00-9:00	1000	100
UB2	1	25000	1000	3:00-9:00	1000	100
UB3	3	25000	1000	3:00-9:00	1000	100
UB4	2	25000	1000	3:00-9:00	1000	100
UB5	0	25000	1000	3:00-9:00	1000	100
UB6	4	8000	100	3:00-9:00	1000	100

5.1.2. Simulation Result

The table below lists virtual machine allocation counts for every load balancing policy in the simulation for the first data center. Table 4. shows the load allocation count for all VM in the first and second data centers for every load balancing algorithm. In table 4, every virtual machines load distribution/ task allocations for experiment one has shown.

Table 4. VM allocation count for data center one

Data Center One				
VM's	Round Robin	ESCE	Throttled	Proposed algorithm
VM 1	393151	1447869	1447765	393793
VM 2	393151	110246	110230	393928
VM 3	393151	13178	13189	392387
VM 4	393150	1319	1302	392495

Table 5. lists virtual machine allocation counts for every load balancing policy in the simulation for the second data center.

Table 5. VM allocation count for data center two

Data Center two				
VM's	Round Robin	ESCE	Throttled	Proposed algorithm
VM 1	243178	1126145	1126193	244337
VM 2	243178	80478	80439	243210
VM 3	243178	8480	8479	243293
VM 4	243178	729	722	243363
VM 5	243178	58	53	241687

5.2. Experiment 2 Evaluation of average response time

This experiment evaluates the proposed Algorithm's response time compared with other load-balancing algorithms. The load distribution in the data centers considers the specification of the virtual machines, such as bandwidth, memory, and load allocation counts. The proposed Algorithm's response time is evaluated by comparing the results with other load-balancing algorithms.

5.2.1. Configuration

The following configurations are used in each data center: x86 architecture, Linux operating system, Xen virtual machine manager, 204800 RAM (MB), 1000000 available bandwidth, and TIME SHARED VM scheduling policy. Two data centers are configured with 10 virtual machines each. The following settings for each virtual machine: Image size: 10000 MB, memory: 512 MB, bandwidth: 1000 MB. The duration of the simulation is 60 minutes. In table 6, the configuration values for experiment two have shown.

Table 6. Simulation configuration of experiment two.

Userbase	Region	Requests per user per hour	peak hours	Average peak users	Active users, off-peak users
UB1	5	60	3:00-9:00	1000	100
UB2	1	60	3:00-9:00	1000	100
UB3	3	60	3:00-9:00	1000	100

5.2.2. Simulation Result

This experiment is repeated for four load balancing algorithms, and the results are compared between all load balancing algorithms discussed here. The objective of this experiment is to evaluate the response time of the load-balancing algorithms. The following diagram shows the average response time of every Algorithm. Figure 7 shows the graphical representation of the average response time for experiment two.

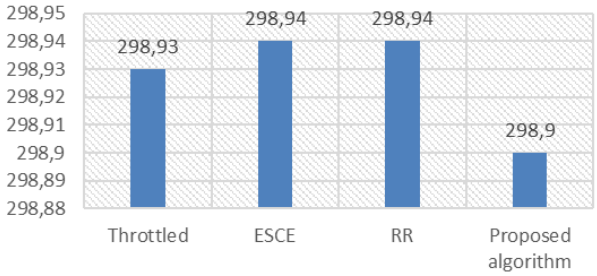

Figure 7. Average response time for experiment two

The above diagram shows the average response time for experiment two. Based on the configuration provided, the average of the proposed algorithm is lower than other load balancing algorithms. The average response time for the Round Robin algorithm and ESCE load balancing algorithm is 298.94 milliseconds, while Throttled load balancing algorithm shows a relatively better response time which is 298.93 milliseconds, and the proposed Load balancing algorithm shows a better average response time (298.9 milliseconds) than other algorithms in these experiments.

5.3. Experiment 3 Load distribution Variance experiment

This experiment also focuses on resource distribution among the Datacenters. It measures the fairness of load distribution between virtual machines in data centers. The load distribution in the data centers considers the specification of the virtual machines, such as bandwidth, memory, and load allocation counts. The proposed Algorithm's response time is evaluated by comparing the results with other load-balancing algorithms. This experiment uses standard deviation to evaluate the variance of load distribution.

This experiment evaluates the load distribution between the virtual machines. Load distribution variation is considered to evaluate the efficiency of the Algorithm. i.e., If the variation is low, the Algorithm distributes loads between virtual machines fairly.

Standard Deviation: Standard deviation measures the "average" scatter around the mean. It shows continuous variables' distribution (shape, center, range, variation). It is the most commonly used measure of variation. It shows variation in the mean and has the same units as the original data.

Equation 2 is used to calculate the standard deviation.

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}} \quad (3)$$

Where σ is the standard deviation, xi is the load distribution count for the ith VM, μ is the mean of the total number of tasks, and N is the number of Virtual machines.

5.3.1. Configuration

In this experiment, two data centers with four virtual machines and five virtual machines are configured. It is supposed to handle 25000 requests per user base. Its simulation duration is 60 minutes on average.

The following diagram shows the main configuration for experiment 3. This configuration has 3 data centers with five virtual machines each.

The following Table 7 shows the load distribution between virtual machines for every Algorithm in data center one.

Table 7. Algorithms task allocation for all virtual machines in datacenter one for experiment three.

Data Center one				
VM's	Round Robin	ESCE	Throttled	Proposed algorithm
VM 1	774	3602	3600	831
VM 2	774	236	240	795
VM 3	774	27	29	740
VM 4	774	3	1	748
VM 5	774	2	0	756
σ	0	1416	1415	34

$\bar{x}$ for all Algorithms is 774, as the total number of loads in all experiments is equal.

The following Table 8 shows the load distribution between virtual machines for every Algorithm in data center one.

Table 8. Algorithms task allocation for all virtual machines in datacenter two for experiment three.

Data Center two				
VM's	Round Robin	ESCE	Throttled	Proposed algorithm
VM 1	502	2322	2327	486
VM 2	501	100	151	493
VM 3	501	75	24	498
VM 4	501	7	4	527
VM 5	501	2	0	502
σ	0	911	914.6	13.96

Algorithms with lower standard deviation values show better load distribution. The proposed algorithm shows fair load distribution between its virtual machines according to the above values. The total number of tasks is fairly shared between the virtual machines according to their vmWeights. Round Robin algorithm performs better than all algorithms, but it does not consider the properties of virtual machines such as bandwidth and their capacities.

6. Conclusion and Recommendations

The first experiment evaluated resource utilization and load distribution between the virtual machines configured with two data centers and five userbases with 25000 requests per user for each user base. It showed better performance than other load-balancing algorithms in resource utilization, which helps avoid overutilization and underutilization. ESCE almost showed similar performance with throttled even though it could be managed to assign loads to all Virtual machines; almost the last virtual machines stayed idle while the first four or five virtual machines handled the tasks. The third Algorithm, Round Robin, allocated load to every virtual machine regardless of its load capacity. This is a drawback in the Round Robin algorithm as some low-configured virtual machines are assigned equal loads with other virtual machines. Generally, HLBA checks the capacity of Virtual machines before allocating a task to them, which is a drawback in all other load-balancing algorithms discussed in this study.

In the experiment section, we tested and evaluated the efficiency and performance of the proposed load-balancing Algorithm and compared it with other load-balancing algorithms. The comparison is between Round Robin, Throttled, and ESCE algorithms. The proposed algorithm has improved more than other load-balancing algorithms in resource utilization and average response time.

The second experiment evaluated response time and compared it with the other load-balancing algorithms. In the experiment, the proposed algorithm showed better response time than the other four algorithms; Throttled load balancing algorithm, Round Robin load balancing algorithm, and ESCE (Equally shared Current Execution) load balancing algorithm. Round Robin algorithm allocates tasks to all virtual machines without considering the capacity of the Virtual machines. The proposed Algorithm improved this drawback by comparing the virtual machines' vmWeight. vmWeight is the ratio of bandwidth to the number of tasks allocated.

The third experiment also evaluated the load distribution/ resource utilization using standard derivation. According to the results, the proposed algorithm performed well.

Below are some recommendations and future works to bring better efficiency and performance out of this Algorithm.

- Working on data center response time, which is relatively not good
- Focusing on other load balancing parameters such as energy consumption and including Works that focus on security, point of failure, fault

tolerance, and deadline awareness.

References

[1] S. S. Sefati, M. Mousavinasab, and R. Zareh Farkhady, "Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation," J. Supercomput., vol. 78, no. 1, pp. 18–42, Jan. 2022, doi: 10.1007/S11227-021-03810-8/FIGURES/14.

[2] M. Gopala and K. Sriram, "CHALLENGES OF CLOUD COMPUTE LOAD BALANCING ALGORITHMS," Accessed: Jun. 12, 2022. [Online]. Available: www.irjmets.com.

[3] N. Kannan, Y. Mobarak, and F. Alharbi, "Application of cloud computing for economic load dispatch and unit commitment computations of the power system network," Adv. Intell. Syst. Comput., vol. 1108 AISC, pp. 1179–1189, 2020, doi: 10.1007/978-3-030-37218-7_124.

[4] S. M. G. Kashikolaei, A. A. R. Hosseinabadi, B. Saemi, M. B. Shareh, A. K. Sangaiah, and G. Bin Bian, "An enhancement of task scheduling in cloud computing based on imperialist competitive algorithm and firefly algorithm," J. Supercomput., vol. 76, no. 8, pp. 6302–6329, 2020, doi: 10.1007/s11227-019-02816-7.

[5] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: A review," J. King Saud Univ. - Comput. Inf. Sci., Mar. 2021, doi: 10.1016/J.JKSUCI.2021.02.007.

[6] S. Afzal and G. Kavitha, "Load balancing in cloud computing – A hierarchical taxonomical classification," J. Cloud Comput., vol. 8, no. 1, 2019, doi: 10.1186/s13677-019-0146-7.

[7] M. Nanjappan and P. Albert, "Hybrid-based novel approach for resource scheduling using MCFCM and PSO in cloud computing environment," Concurr. Comput. Pract. Exp., vol. 34, no. 7, p. e5517, Mar. 2022, doi: 10.1002/CPE.5517.

[8] V. Richhariya, R. Dubey, and R. Siddiqui, "ORIENTAL JOURNAL OF Hybrid Approach for Load Balancing in Cloud Computing," Orient. J. Comput. Sci. Technol., vol. Vol. 8, no. 3, pp. 241–246, 2015.

[9] A. Alkhatib, S. Alzu'bi, A. A. Alkhatib, and T. Sawalha, "Load Balancing Techniques in Software-Defined Cloud Computing: an overview," ieeexplore.ieee.org, doi: 10.1109/SDS49854.2020.9143874.

[10] J. Bhatia, T. Patel, … H. T.-… S. on C., and undefined 2012, "HTV dynamic load balancing algorithm for virtual machine instances in cloud," ieeexplore.ieee.org, Accessed: Feb. 12, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6481229/?casa_token=P_h-elG6PQ0AAAAA:EYSjp67iqG1oju7hnbsUFRjB3WR9a9Xutyg4aHQ07PgzjUwxYB9yUrbaZzd3WeJ9x-jDhTJZBA.

[11] B. H. Shanthan and L. Arockiam, "Resource Based Load Balanced Min Min Algorithm(RBLMM) for static Meta task Scheduling in Cloud," Accessed: Jun. 13, 2022. [Online]. Available: http://www.ijetjournal.org.

[12] S. Eswaran and M. Rajakannu, "Multiservice Load Balancing with Hybrid Particle Swarm Optimization in Cloud-Based Multimedia Storage System with QoS Provision," Mob. Networks Appl., vol. 22, no. 4, pp. 760–770, Aug. 2017, doi: 10.1007/S11036-017-0840-Y/FIGURES/5.

[13] G. Ramadhan, T. W. Purboyo, R. Latuconsina, and A. R. Robin, "Experimental Model for Load Balancing in Cloud Computing Using Throttled Algorithm," Int. J. Appl. Eng. Res., vol. 13, no. 2, pp. 1139–1143, 2018, [Online]. Available: https://www.ripublication.com/ijaer18/ijaerv13n2_42.pdf.

[14] A. Alkhatib, S. Alzu'bi, A. A. Alkhatib, A. Alsabbagh, R. Maraqa, and S. Alzubi, "Load balancing techniques in cloud computing: Extensive review," researchgate.net, vol. 6, no. 2, pp. 860–870, 2021, doi: 10.25046/aj060299.

[15] R. Somani and J. Ojha, "A Hybrid Approach for VM Load Balancing in Cloud Using CloudSim," Int. J. Sci. Eng. Technol. Res., vol. 3, no. 6, pp. 1734–1739, 2014.

[16] D. Patel, "International Journal of Modern Trends in Engineering and Research Efficient Throttled Load Balancing Algorithm in Cloud Environment," Int. J. Mod. Trends Eng. Res., vol. 02, no. 03, pp. 463–481, 2015.

[17] K. Kaur, "Equally Spread Current Execution Load Algorithm - A Novel Approach for Improving Data Centre' s Performance in Cloud Computing," Int. J. Futur. Revolut. Comput. Sci. Commun. Eng., vol. Volume: 4, no. August, pp. 10–12, 2018.

[18] S. A. Narale and P. K. Butey, "Throttled Load Balancing Scheduling Policy Assist to Reduce Grand Total Cost and Data Center Processing Time in Cloud Environment Using Cloud Analyst," Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2018, no. Icicct, pp. 1464–1467, 2018, doi: 10.1109/ICICCT.2018.8473062.

[19] M. U. Sapra1 and A. Sharma2, "Wisely Randomized Weighted Throttled Load-Balancing Algorithm for Cloud," ijerm.in, vol. 7, no. 5, 2020, Accessed: Feb. 22, 2022. [Online]. Available: https://www.ijerm.in/tmp/Vol_7/IJERM_11070.pdf.

[20] R. S. Sajjan, "Load Balancing and its Algorithms in Cloud Computing : A Survey International Journal of Computer Sciences and Engineering Open Access Load Balancing and its Algorithms in Cloud Computing : A Survey," no. January 2017, 2019.

[21] A. Singh, S. Bhat, R. Raju, R. D.-A. Comput, and undefined 2017, "Survey on various load balancing techniques in cloud computing," academia.edu, vol. 7, no. 2, pp. 28–34, 2017, doi: 10.5923/j.ac.20170702.04.

[22] H. Chandra, H. B.-I. J. of Computer, and undefined 2017, "A survey of load balancing algorithms in cloud computing," researchgate.net, Accessed: Feb. 22, 2022. [Online]. Available: https://www.researchgate.net/profile/Harish-Sharma-5/publication/357458654_A_SURVEY_OF_LOAD_BALANCING_ALGORITHMS_IN_CLOUD_COMPUTING/links/61cf19dfb6b5667157ba8c08/A-SURVEY-OF-LOAD-BALANCING-ALGORITHMS-IN-CLOUD-COMPUTING.pdf.

[23] S. Garg, D. V. Gupta, and R. K. Dwivedi, "Enhanced Active Monitoring Load Balancing algorithm for Virtual Machines in cloud computing," Proc. 5th Int. Conf. Syst. Model. Adv. Res. Trends, SMART 2016, pp. 339–344, Apr. 2017, doi: 10.1109/SYSMART.2016.7894546.

[24] A. F. Pathan and Sneha.N, "Simulation of Load Balancing Algorithms using Cloud Analyst," Int. J. Eng. Res. Technol., vol. 2, no. 13, Jul. 2018, doi: 10.17577/IJERTCONV2IS13084.

[25] J. Gustedt, E. Jeannot, and M. Quinson, "EXPERIMENTAL METHODOLOGIES FOR LARGE-SCALE SYSTEMS: A SURVEY," http://dx.doi.org/10.1142/S0129626409000304, vol. 19, no. 3, pp. 399–418, Nov. 2011, doi: 10.1142/S0129626409000304.

[26] 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," Softw. - Pract. Exp., vol. 41, no. 1, pp. 23–50, Jan. 2011, doi: 10.1002/spe.995.

[27] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, "CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications," in Proceedings - International Conference on Advanced Information Networking and Applications, AINA, 2010, pp. 446–452, doi: 10.1109/AINA.2010.32.

[28] 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," Softw. - Pract. Exp., vol. 41, no. 1, pp. 23–50, Jan. 2011, doi: 10.1002/spe.995.

[29] A. Pradhan and S. K. Bisoy, "A novel load balancing technique for cloud computing platform based on PSO," J. King Saud Univ. - Comput. Inf. Sci., Oct. 2020, doi: 10.1016/J.JKSUCI.2020.10.016.