How to Cite:
Awasthi, V., Tiwari, A., Jayaram, V., Shukla, S. K., Pant, K., & Abhilash, K. S. (2022). Load balancing technique based on hybrid resource utilization in cloud computing. *International Journal of Health Sciences*, *6*(S5), 2082–2093. https://doi.org/10.53730/ijhs.v6nS5.9079

Load balancing technique based on hybrid resource utilization in cloud computing

Dr Vishal Awasthi
Asst Prof, Department Of ECE, School Of Engineering (UIET), C.S.J.M. University, Kanpur
Email: awasthiv@Rediffmail.com

Dr Ajay Tiwari
Asst Prof, Department Of ECE, School Of Engineering (UIET), C.S.J.M. University, Kanpur
Email: ajy_gkp23@rediffmail.com

Jayaram v
Research Scholar, Dept of Mechanical Engineering, Noorul Islam center for Higher Education
Tamilnadu
Email: jayaramvijayan@gmail.com

Dr. Surendra Kumar Shukla
Associate Professor, Department of Computer Science & Engineering, Graphic Era Deemed to be University, Dehradun, Uttarakhand, India, 248002
Email: surendrakshukla21@gmail.com

Dr. Kumud Pant
Designation: Associate Professor, Department of Biotechnology, Graphic Era Deemed to be University, Dehradun, Uttarakhand, India, 248002
Email: pant.kumud@gmail.com

Dr. Abhilash. KS
Managing Director, EduCorp Centre for Research and Advanced Studies Pvt. Ltd.
Thiruvananthapuram, Kerala
Email: dr.abhilashks@gmail.com

***Abstract*---**Cloud computing uses the internet to supply dynamic services including memory, data, bandwidth and applications. Work schedules have an influence on cloud service reliability and performance. A proper provisioning method is required for a systematic resource allocation, which comprises of large virtual resources. Depending on the present state of the system, load balancing solutions can be distinguished as dynamic or static.

Dynamic or static load balancing solutions can be employed to increase server response time or to raise load balancing factors for quicker and more efficient resource utilization. To decrease the load across resources and maximize CPU usage, a hybrid load balancing technique is developed. In the cloud, we have a finite quantity of resources that must be efficiently managed in order to fulfill tasks. Requests are transmitted to a cloud server, which assigns work via quadratic probing. During load balancing, the load is shifted from heavy-weighted servers to lighter-weighted servers, enhancing CPU usage. The suggested methodology's performance was assessed using average mean response time, make-span, average make-span, and average resource utilization. The Load Balancing Algorithm (LBA) is created with the primary purpose of reducing job completion time and increasing the average resource utilization ratio. Individual Virtual Machine (VM) loads were analyzed after tasks were sorted in decreasing order. If incoming tasks place a greater burden on the VMs than its current capacity, additional VMs are produced. The proposed load balancing approach surpasses existing Load Balancing Decision Algorithm (LBDA) in terms of and resource utilization. The proposed method has a shorter average make-span and average mean reaction time than LBDA.

***Keywords*---**Cloud Computing, Load Balancing, Hybrid Resource Utilization, Dynamic Load Balancing, Virtual Machine.

Introduction

Cloud computing has recently received a lot of interest, and it's now widely recognized as a novel method for organizing data and increasing data availability globally over the internet. The integration of main computing and storage services enables network services to provide dynamic and flexible virtualized resources. It also handles computer resources including data centers and servers. Customers must be provided with services that meet their needs. Some servers are overburdened, while others are under loaded, due to the diversion of services in cloud computing. As a result, a load balancing approach is essential to change the load of server and to improve resource usage. All cloud services are presented as web services, and these services adhere to Simple Object Access Protocol (SOAP) and other organizational standards. Resource pooling, on-demand self-service, measured services/pay as you go, rapid elasticity, and broad network connectivity are the five basic cloud features. Cloud deployment models specify how services are delivered to consumers in the cloud. Traditional computing requires businesses to pay for the resources they use, cloud computing allows services to be scaled up or down as needed. Cloud computing provides international access to business-oriented computer services that are paid for on a subscription basis [1]. A cloud contains a limitless amount of resources, scheduling tactics are critical for getting the most out of those resources by properly employing them. Cloud computing is a more advanced variant of grid and cluster computing. The way the cloud's overall structure is logically tied to other technologies that overlap with it. Cloud Computing got its name from the

fact that clouds are employed as computation components [2]. These data and apps operate on a global scale and may be accessed from anywhere.

Load balancing is an essential study subject in cloud computing for providing Quality-of-service (QoS). With an effective load balancing approach that decreases server response time, maximizes resource consumption, reduces request rejection, and supports system competency, customers' requests may be satisfied to the maximum degree feasible [3]. As a result, researchers should use an optimum load balancing strategy with the most parameters. A proper provisioning method is required for a systematic resource provisioning in the cloud, which comprises of large virtual resources [4]. Depending on the present state of system, load balancing techniques can be characterized as dynamic or static settings. In a static load balancing setting, researchers must already be familiar with the system's processing speed, node capacity, memory storage, performance capabilities, and other statistical data. Since the working environment is static, it is impossible to adjust user demand or assign additional burden when executing apps or providing requests [5]. The benefit of employing static load balancing is that it is simple to set up and is recommended for homogeneous cloud environments. Resources are flexible and configurable on a heterogeneous cloud in a dynamic environment [6]. The dynamic environment of the cloud may adapt in real time according to the needs of users. Load balancing systems may be split into three types based on the geographical distribution of nodes. Decision-making and other scheduling methods, as well as load balancing, will be centralized if a cloud is constructed on a centralized server [7]. In this case, the central server is in charge of deciding whether to use a static or dynamic method. It improves the speed of other servers or nodes, but it is not fault tolerant and adds overhead. Load balancing across a large area is known as distributed load balancing. Multiple nodes are built instead of a single node to make choices and monitor the load more effectively and precisely. Each node is now equally responsible for maintaining the knowledge table. In a static environment, the load is distributed evenly and efficiently, while in a dynamic context, the burden is re-distributed. In a dispersed context, the probability of failure is relatively low. As a result, the system may be described as fault-tolerant and dependable. Since the system is dependable and fault-tolerant, none of the nodes are overwhelmed [8]. In hierarchical load balancing, the master-slave approach/model is employed, since it employs different tiers in load balancing choices in the cloud. It is a tree-based data structure in which the root/parent node balances or controls each node. The data/information compiled by the parent node can be used for scheduling and allocation. In this strategy, all of the root nodes are responsible for load distribution.

In a heterogeneous cloud server, the problem of optimal resource usage and load imbalance has been highlighted. The goal of this method is to build up efficient resource provisioning and load management across a large number of cloud resources while reducing response time and time to market. The goal is to enable effective resource provisioning and load management across a large number of cloud resources while keeping response time and time to market to a minimum.

Saber *et al* [8] proposed a Heterogeneous Initialized Load Balancing (HILB) approach for implementing an effective task scheduling procedure that improves

the make-span while allowing appropriate load variation. This strategy can be used on both homogeneous and heterogeneous cloud resources. Radhamani *et al* [9] proposed Hawks Optimization and Pigeon-inspired Optimization algorithms to construct an efficient load balancing strategy. The incoming request was appropriately balanced utilizing the Hawks approach after assessing the overloaded and under loaded Virtual Machines (VM). This approach is implemented in the JAVA IDE, which is incorporated into the cloud simulation framework, and performance is evaluated. Muteeh *et al* [10] proposed a Multi-resource Load Balancing Algorithm (MrLBA). The suggested approach aims for a quick make-time and low cost. According to the findings, by maintaining a balanced load across resources, MrLBA proficiently utilize existing resources.

Kodli *et al* [11] implemented Hybrid Max-Min Genetic Algorithm (HMMGA) that drastically improves workload performance disparities in the cloud. HMMGA increased resource utilization by 10% to 40% as compared to the max-min model. Gundu *et al* [12] proposed that an instance of software or hardware can be employed to balance load. A hybrid algorithm is one that blends many algorithms into one. Make-span is 292.46, waiting time is 0.01, and burst time is 0.23 in the algorithm. Kaur *et al* [13] proposed hybrid Heuristic-Metaheuristic approaches for load balancing optimization. The suggested framework is based on a metaheuristic algorithm that combines heuristic approaches. Two crucial measures such as make-span and cost were used to evaluate the framework's performance.

Subalakshmi *et al* [14] proposed an advanced hybrid technique combining both throttle and uniformly extend modern execution algorithms. The Enhanced Hybrid approach stores a list of requests that have been assigned. These algorithms are used in this study to provide a hybrid load balancing solution. The proposed strategy is put to the test using the CloudSim simulator, and the findings show that it outperforms earlier methods on similar objectives. Lawanyashri *et al* [15] proposed Energy-aware Hybrid Fruitfly Optimization. The recommended approach is used in cloud computing to accomplish optimal supply deployment while minimizing energy utilization and expenditures. This strategy outperforms existing load balancing strategies in terms of efficiency.

Methodology

Cloud computing refers to the on-demand availability of computer system resources like data storage and processing power without the user having to handle them directly. Large clouds commonly divide functions across several locations, each of which is a data centre. Despite the advantages of QoS, load balancing is one of the most difficult research topics in cloud computing. A proper provisioning method is required for systematic resource provisioning in the cloud, which comprises large virtual resources. A cloud is a collection of several servers that are geographically scattered. The cluster must not be overloaded in order to provide smooth cloud services. Researchers devised a new way while examining this issue. While calculating the cluster's dimension size, the maximum number of VMs/resources and their execution rates were established to fulfill tail sojourn duration and reduce response time. Execution and response times, as well as load distribution, should be maintained to a minimum.

Proposed Cloud Model

To decrease the load across resources and maximize CPU usage, a hybrid load balancing technique is presented in the method. We have a limited amount of resources in the cloud, which must be handled efficiently to do tasks efficiently. The suggested system model is organized into four sections, as follows: Cloud server requests are issued, which utilizes quadratic probing assigning the work, distributes demand from servers, and improves CPU usage while balancing load. The architecture of proposed load balancing system for cloud computing is depicted in Figure 1.

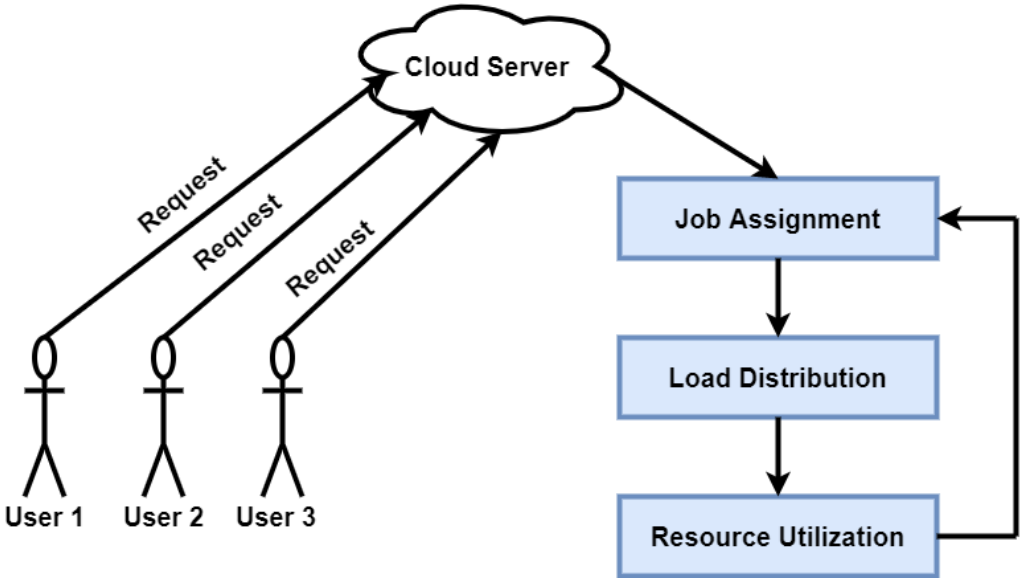

Figure 1: Proposed Load Balancing System Model

First and foremost, server receives requests from various users connected to the cloud. The server assigns job to the VMs and the loads are distributed evenly across various VMs. A comparison study of numerous work assignment techniques was carried out as displayed in Table 1.

Table 1: Job Assignment Scheme Comparison

Tasks	Sequential	Linear Hashing	Quadratic Hashing
10	0.73	1.07	0.72
50	1.47	2.44	1.24
100	2.42	3.93	2.13
200	3.6	4.26	2.92
400	5.9	6.80	4.6

According to the table above, hashing with a quadratic probing strategy was chosen for the task assignment since it generated better results than other strategies. The calculation of various parameters utilized in the proposed model is important for the performance analysis.

The execution time of i^{th} task on j^{th} resource (E_{ij}) can be computed using Eqn.(1). Here we consider T_i as the current task and V_j as the current VM.

$$E_{ij} = \frac{(T_i)_{length}}{(V_j)_{size}} \tag{1}$$

The waiting time of i^{th} task on j^{th} resource can be computed using Eqn. (2). The make-span of j^{th} resource after assigning i^{th} task is computed using Eqn. (3).

$$W_{ij} = E_{ij}(T_{ij})_{finish\ time} \tag{2}$$

$$MS_{ij} = W_{ij} + E_{ij} \tag{3}$$

The total time required for the total make span is computed using Eqn. (4). The computation of average make span is expressed using Eqn.(5). The total resource utilization is computed using the Eqn.6.

$$T_{MS} = MS_{00} + MS_{11} + \text{---------} MS_{n-1\,m-1} \tag{4}$$

$$MS_{avg} = \frac{T_{MS}}{m} \tag{5}$$

$$RU_j = \frac{UT_j}{MS_{ij}} * 100 \tag{6}$$

Hybrid Load Balancing Algorithm:

Step 1: Make i = 0 and j=0.
Step 2: While (i≠ n-1)
Allocate the i^{th} resource to the j^{th} task.
Compute AVMS as the factor for load balancing
i = i + 1
Step 3: Choose the VM for T_i
Step 4: Estimate MS_{ij}
If MS_{ij}> AVMS
Compute RU_j
If RU_j < 100, Allocate T_i to VM_j
Else, find the subsequent resource with an RU < 100
Else, Allocate T_i to VM_j
i = i + 1
Step 5: End of Loop
Step 6: Stop

Experimental Results

The performance of the suggested approach was evaluated in terms of average resource usage, average make-span, average mean response time and make span in this research. If an algorithm has a short make-span, a short average mean response time, and high average resource utilization, it is said to be efficient.

$$AverageMean\,\mathrm{Re}\,sponseTime = \frac{\sum_{j=0}^{m}\mathrm{Avg}(\mathrm{RT_j})}{m} \quad (7)$$

$$Avg(RT_i) = \frac{\sum_{i=0}^{n} RT_i}{n} \quad (8)$$

$$RT_i = FT_i - ST_i \quad (9)$$

Where, finish time is denoted as FT_i and submission time is denoted as ST_i for the i^{th} task. The term "maximum resource utilization" refers to the fact that resources should not remain idle. They must be occupied with some calculations.

The approach was simulated using CloudSim, and the results were compared to those of other methods. During the investigation, the similar criteria are taken into account as per previous algorithms. In LBDA, tasks were allocated according to the criteria of VM such as Under load, Balanced, Overload and Optimally Balanced. Then the summation of tasks completion time with the load of VM was estimated as time for completion. The tasks having lower completion time are selected and moved to different VMs for reducing make-span and improving the usage of VM. The findings of the experiments and research for the comparison of MS_{avg} and $Avg(RT_i)$ between the proposed algorithm and LBDA are presented in Tables 2 and 4.

Table 2. Average Make-span Comparison

(Tasks/VM)	LBDA	Proposed Algorithm
100/6	361	160.97
150/8	428	206.03
200/10	431	278.35
250/12	434	303.68
300/14	459	318.60

Table 2 shows that the suggested method outperforms LBDA in terms of minimal average make-span. Figure 2 shows a graphical depiction of the results for a better understanding of the findings.

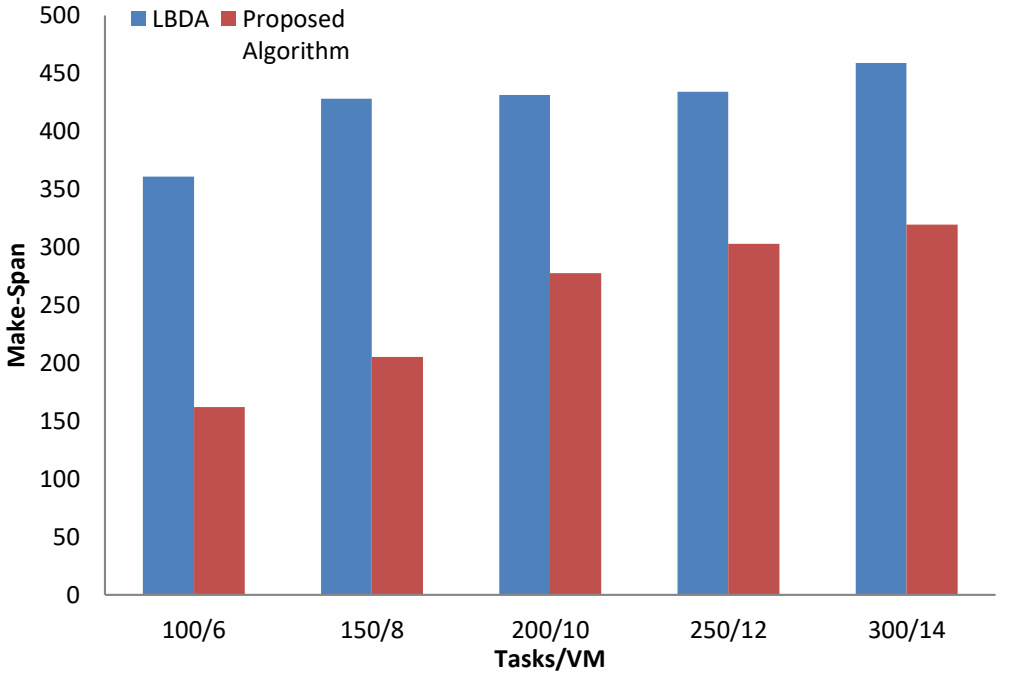

Figure 2: Comparison of Average Make-span

Table 3 shows a comparative analysis between the LBDA with the suggested algorithm in terms of $Avg(RT_i)$.

Table 3: Average Mean Response Time Comparison

(Tasks/VM)	LBDA	Proposed Algorithm
100/6	279	72.92
150/8	302	78.23
200/10	315	83.60
250/12	328	114.30
300/14	334	124.17

Table 3 shows that, when compared to LBDA, the suggested algorithm has a shorter average mean reaction time. Figure 3 depicts a graphical depiction of the results in the same way considered in accordance with $Avg(RT_i)$. Similar settings were considered for simulation during the conduct of experiment.

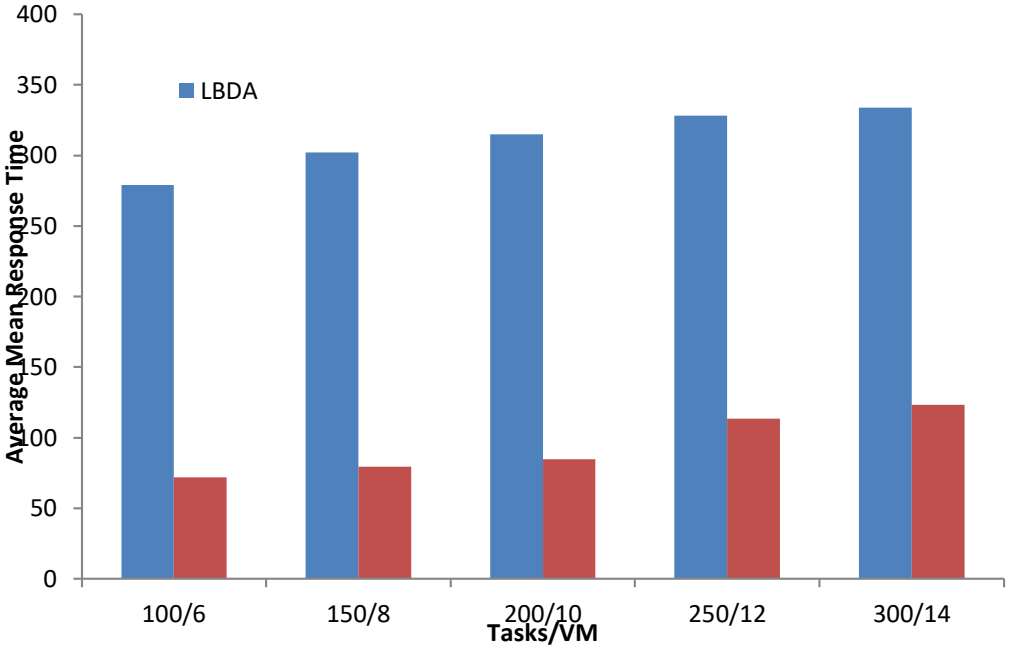

Figure 3: Average Mean Response Time Comparison

Figure 4 depicts a graphical depiction of the results in the same way made in respect of make-span. Similar settings were considered for simulation during the conduct of experiment.

Table 4: Make-span Comparison

No. of Tasks	LBDA [6]	Proposed Algorithm
10	506	169.86
15	635	226.11
20	560	216.11
25	633	311.11
30	616	488.61
40	794	576.11
50	913	707.36

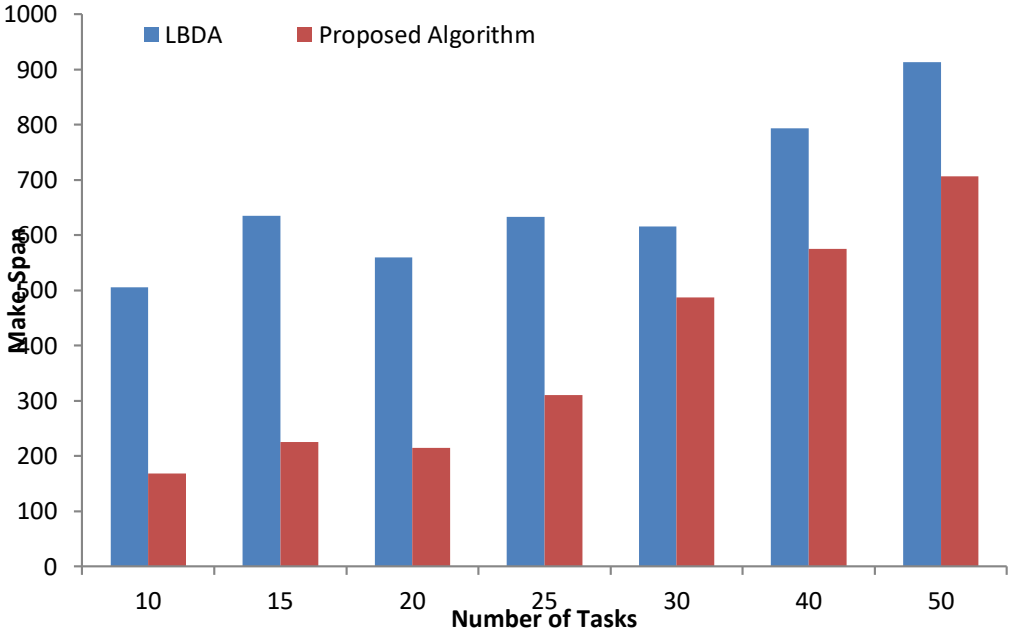

Figure 4: Comparison of Make-span.

The recommended method has a shorter make-span than LBDA, based on the conclusions of the investigation and Table 5. It may be concluded that the offered algorithms produce superior outcomes in terms of execution time. Figure 4 depicts a graphical representation of the table 6 analysis.

Table 5: Average Resource Utilization Rates in Comparison

Experiments (Tasks)	LBA [6]	Proposed Algorithm
10	74	77.43
15	86	85.34
20	76	79.24
25	67	78.44
30	72	81.19
40	89	88.83

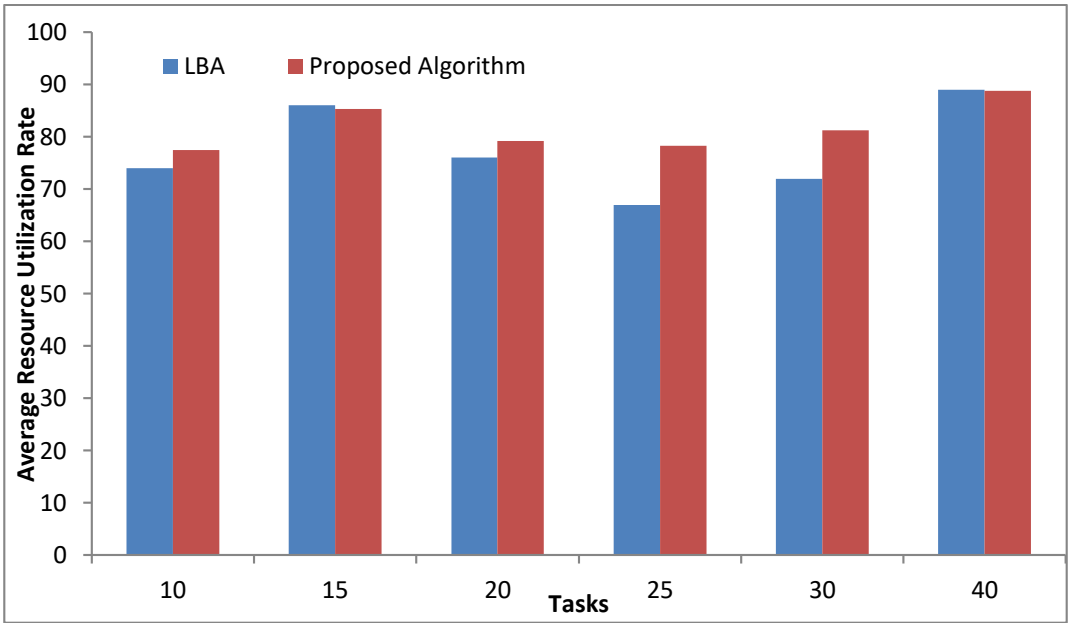

Figure 5: Comparison of average resource utilization

Figure 5 depicts a graphical depiction of Table 5 for customers' better understanding. In comparison to LBDA and LBA algorithms, the proposed approach has the shortest make-span, mean average response time, average make-span, and maximum mean RU_j , as shown in the tables.

Conclusion

Cloud computing has gotten a lot of press recently, and it's now widely recognized as a unique way of organizing data and improving data availability around the globe via the internet. Organizations that work in the cloud are typically involved in the development, collaboration, bringing users to online services, record documentation, and a variety of other tasks. They must be able to handle a large number of concurrent requests while also engaging servers consistently and reliably. Maximizes customer's demands may be met to the greatest extent possible with an effective load balancing technique that minimizes server response time, minimizes resource utilization, reduces request rejection, and promotes system competence. To decrease the load across resources and maximize CPU usage, a hybrid load balancing technique is developed. The suggested technique is compared to the LBA. LBA's major purpose was to shorten the time it took to perform a task. More VMs are created if the burden of incoming jobs exceeds the capacity of the VMs. Experiments revealed that hybrid LBA had a lower MS_{avg} and $Avg(RT_i)$ than LBDA.

References

1. G. Breiter and M. Behrendt, “Life Cycle and Characteristics of Services in the World of Cloud Computing,” IBM Journal of Research and Development, vol. 53, pp. 3:1–3:8, July 2009.
2. J. Liu, X.-G. Luo, X.-M. Zhang, F. Zhang, and B.-N. Li, “Job Scheduling Model for Cloud Computing Based on Multi-Objective Genetic Algorithm,” International Journal of Computer Science Issues, vol. 10, p. 134, Jan. 2013.
3. Monika, D. P. Kumar, and D. S. Tyagi, “Survey on Various Scheduling Algorithmsin Cloud Computing,” International Journal of Scientific & Engineering Research, vol. 7, pp. 204–209, Dec. 2016.
4. R. Bansal and P. Gautam, “Extended Round RobinLoadBalancinginCloudComputing,” International Journal of Engineering and Computer Science,vol. 3, Aug. 2014.
5. M. Kumar and S. Sharma, “Dynamic Load Balancing Algorithm For Balancing The Workload Among Virtual Machine In Cloud Computing,” Procedia Computer Science, vol. 115, pp. 322–329, Dec. 2017.
6. P. Maheta and S. Dave, “Utilizing Round Robin Concept for Load Balancing Algorithm at Virtual Machine Level in Cloud Environment,” vol. 94, pp. 23–29, May 2014.
7. S. Kumar and S. Khurana, “Scheduling in Cloud Computing : A Review,” International Journal of Advanced Research in Computer Science, vol. 5, no. 1, pp. 79–81, 2014.
8. Saber, W., Moussa, W., Ghuniem, A. M., & Rizk, R. (2021). Hybrid load balance based on genetic algorithm in cloud environment. International Journal of Electrical & Computer Engineering (2088-8708), 11(3).
9. Annie Poornima Princess, G., & Radhamani, A. S. (2021). A hybrid meta-heuristic for optimal load balancing in cloud computing. Journal of Grid Computing, 19(2), 1-22.
10. Muteeh, A., Sardaraz, M., & Tahir, M. (2021). MrLBA: multi-resource load balancing algorithm for cloud computing using ant colony optimization. Cluster Computing, 24(4), 3135-3145.
11. Kodli, S., & Terdal, S. (2021). Hybrid max-min genetic algorithm for load balancing and task scheduling in cloud environment. Int J Intell Eng Syst., 14(1), 63-71.z
12. Gundu, S. R., & Anuradha, T. (2019). Improved hybrid algorithm approach based load balancing technique in cloud computing. Global journal of computer science and technology.
13. Kaur, A., & Kaur, B. (2019). Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment. Journal of King Saud University-Computer and Information Sciences.
14. Subalakshmi, S., & Malarvizhi, N. (2017). Enhanced hybrid approach for load balancing algorithms in cloud computing. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 2(2), 136-142.
15. Lawanyashri, M., Balusamy, B., & Subha, S. (2017). Energy-aware hybrid fruitfly optimization for load balancing in cloud environments for EHR applications. Informatics in Medicine Unlocked, 8, 42-50.