Performance Evaluation of Intelligent Load Balancing Algorithms Using Cloud Analyst

Hemant Kumar Singh[1], Dr. Shashi Kant Gupta[2]
[1]Postdoctoral Research Fellow, Lincoln University College, Malaysia
[2]Adjunct Professor, Lincoln University College, Malaysia
[1]hemantbib@gmail.com, [2]shashigupta@lincoln.edu.my

Abstract: Cloud computing has transformed the approach businesses usage their IT assets by providing flexible and scalable services. As the number of cloud users increases, efficient use of resources becomes very important. Load balancing helps in distributing incoming user requests among multiple servers so that performance is improved and system availability is maintained.
This paper discusses different load balancing techniques and studies their effect on performance, scalability, and cost. Cyclic Request Allocation Method, Capacity-Aware Request Control Strategy and Dynamic Balanced Execution Load Distribution Technique are simulated using the Cloud Analyst tool. These algorithms are tested under three Service Allocation and Routing Strategies that are Closest Data Center, Optimized Response Time, and Reconfigure Dynamically with Load Balancer. The evolution is focused on overall response latency and whole data transfer cost.
Keywords: Load Balancing Algorithms, Cloud Analyst, Data Center

1.0 Introduction

Load balancing stands critical mechanism applied in cloud computing infrastructure to proficiently allocate incoming network traffic across multiple servers, guaranteeing best resource uses, maximizing throughput and stopping any individual server from failing due to excessive demand [1]. Load balancing involves the distribution of incoming network traffic or workload across multiple servers or resources. It ensures that no single server bears an excessive burden, thereby improving performance, responsiveness, and reliability. Load balancing is implemented at diverse layers of the network stack [2, 3]

- **Transport Layer (L4 Load Balancing):** It operates at the network transport layer (e.g. TCP/UDP), directing traffic based on network-level information such as IP addresses and ports.
- **Application Layer (L7 Load Balancing):** This type of load balancing operates at the application layer (e.g., HTTP/HTTPS) and can make more sophisticated decisions based on application-specific data, such as URL paths or headers.

1.1 Significance of load balancing in cloud computing

In cloud computing, load balancing plays a crucial role for several reasons [3,4]:
High Availability: Load balancers distribute traffic among multiple servers. If one server fails or becomes overloaded, the load balancer redirects traffic to other healthy servers, ensuring uninterrupted service.
Scalability: Cloud services often need to scale up or down based on demand. Load balancers help in scaling by distributing incoming traffic evenly across available resources, making it easier to add or remove servers as needed.
Performance Optimization: By efficiently distributing traffic, load balancers prevent any single server from being overwhelmed. This ensures optimal performance for users accessing applications or services hosted in the cloud.
Fault Tolerance: Load balancers can detect and reroute traffic away from failed or unhealthy servers, reducing the impact of potential hardware or software failures.

Global Load Balancing: In scenarios where services are deployed across multiple regions or data centers, global load balancing ensures that traffic is directed to the closest or most available data center, optimizing latency and improving user experience.

2.0 Cloud Load Balancing techniques:

2.1 Round Robin:

In computer science, the Round Robin scheduling algorithm operates based on a time quantum—a fixed interval that determines how long each node or process can execute its operations. The time quantum is a critical factor in load balancing algorithms, influencing the efficiency of the system. However, when the time quantum is significantly large, the algorithm's efficiency becomes akin to that of the First Come First Serve (FCFS) algorithm, rendering Round Robin less advantageous in such scenarios. Consequently, determining the appropriate time quantum poses a significant challenge for algorithm designers[5, 6].

Round Robin, despite its simplicity, presents several drawbacks. One major concern revolves around the determination of the time quantum, which imposes an additional burden on the scheduler. Moreover, this algorithm is associated with a high rate of context switching, consequently increasing the turnaround time for processes and leading to lower throughput. These limitations underscore the need for careful consideration and optimization when implementing Round Robin scheduling in practical systems.

2.2 Equally spread current execution load

The approach to load balancing known as Equally Spread Current Execution Load involves the load balancer keeping track of a comprehensive list containing all virtual machines along with their availability statuses. When a client sends a request to the load balancer, it scans through this list of virtual machines. If it identifies a virtual machine that aligns with the specific requirements of the client's request, the request is then directed towards that particular virtual machine [7] [8].

This algorithm operates on the fundamental principle of ensuring an even distribution of the workload across all virtual machines. Achieving this equitable distribution necessitates a comprehensive understanding of the current allocation of workload on each virtual machine. To accomplish this, the algorithm maintains a current allocation table, facilitating the equal distribution of the workload among all virtual machines. By doing so, it aims to optimize throughput.

However, there are certain drawbacks associated with this approach:

It is susceptible to centered failure, wherein failures or issues centralized around specific points can disrupt the system. Additionally, this method lacks a fault tolerance feature, which means it may not be resilient against failures or errors within the system [10].

2.3 Throttled Load balancing mechanism

The Throttled Load Balancing mechanism relies on maintaining records wherein the load balancer holds a table containing the indices of virtual machines and their respective operational states, distinguishing between availability and occupancy. The algorithm begins with a client sending a request to the data center, seeking a suitable virtual machine for a specific task. Subsequently, the data center engages the load balancer to facilitate the allocation of the virtual machine. The load balancer proceeds by systematically scanning the table, starting from the top and continuing until it either identifies the first unoccupied virtual machine or completes a full scan of the table. Upon finding an available virtual machine, the data center then directs the request to the specific virtual machine identified by its unique identifier [4,9].

2.4 Weighted Round Robin

Weighted Round Robin (WRR) load balancing is commonly used in cloud environments to distribute incoming traffic among multiple servers or instances based on their assigned weights. In a cloud setup, load balancers play a crucial role in managing and distributing the incoming requests among various resources to optimize performance and reliability.

This method ensures that more powerful or capable resources handle a larger share of the workload compared to less powerful ones. It's particularly useful in scenarios where servers have different processing capacities or capabilities [11].

For example, in a load balancer configuration, if Server A has a weight of 3 and Server B has a weight of 1, Server A will receive three times more requests than Server B in a single round-robin cycle. Implementing WRR can help optimize resource utilization, improve system performance, and ensure that more capable resources contribute more to the overall task processing [12].

2.5 Least Connection Method

The Least Connection Method is a technique used in cloud-based load balancing to allocate incoming requests to servers based on the number of active connections. In this method, the load balancer maintains information about the current connections each server is handling.

When a new request arrives, the load balancer analyzes the number of active connections on each server and directs the request to the server with the fewest ongoing connections at that moment. The rationale behind this approach is to evenly distribute the workload among the servers by sending incoming requests to the server that currently has the least number of connections.

This method aims to optimize resource utilization and prevent any single server from becoming overloaded, thereby ensuring a more balanced distribution of traffic across the server pool. As servers manage varying loads due to fluctuating traffic, the Least Connection Method helps maintain a more equitable distribution of connections among the available servers in a cloud-based environment.

2.6 IP Hashing

IP Hashing load balancing is a technique used in distributed systems, particularly in load balancing scenarios. This method involves the utilization of the source IP address of incoming requests to determine the destination server. The load balancer applies a hashing algorithm to the source IP address of the client to generate a hash value.

The hash value is then used to map the client's IP address to a specific server from a pool of available servers. The goal is to ensure that all requests coming from a particular client IP address are consistently directed to the same server. This approach helps maintain session persistence or affinity, ensuring that interactions from a specific client are handled by the same server throughout their session.

IP Hashing load balancing is beneficial for applications that require continuous communication or data retention between the client and the server. By consistently directing traffic from the same source IP to the same server, it facilitates the preservation of session-related information and minimizes disruptions or data loss that might occur when requests from the same client are distributed across multiple servers [9, 11].

2.7 Content-Based Routing

Content-Based Routing (CBR) in load balancing is a method that involves directing incoming requests to specific servers based on the content or characteristics of the data being transmitted. Instead of relying solely on factors like IP addresses or connection counts, CBR examines the content of the incoming data packets to determine their destinations.

This technique typically involves parsing or inspecting the content of the incoming requests, extracting

relevant information such as specific keywords, headers, or attributes. Based on this extracted content,

the load balancer employs rules or policies to determine which server within the pool is best suited to handle the request.

CBR enables the load balancer to make more granular and informed decisions about how to distribute the workload among servers. For instance, in a scenario where certain types of requests require specific processing or resources, CBR can route these requests to specialized servers optimized for handling such content. This approach optimizes resource utilization and improves the efficiency of the overall system by ensuring that requests are directed to servers that are best equipped to handle them based on their content characteristics.

2.8 Adaptive Load Balancing algorithm: It continuously monitor system conditions and adjust their strategies in real-time. They might consider factors like server health, network latency, or application performance to make informed decisions about workload distribution.

These dynamic algorithms enhance system performance, scalability, and reliability by efficiently distributing tasks across resources, ensuring optimal utilization and responsiveness even in fluctuating and unpredictable environments [10, 11].

3.0 Feature analysis of some load balancing algorithms [12, 13]

Comparing load balancing algorithms requires assessing their effectiveness in distributing workloads efficiently across available resources and their ability to handle faults or failures within a system. Several key algorithms often undergo comparative analysis based on these criteria:

Table-1

S. No.	Load Balancing Algorithms	Features
1.	**Round Robin**	Known for its simplicity, it cyclically allocates requests among servers. While it guarantees fairness, it might not be optimal for varying server capacities or workload demands.
2.	**Least Connections**	Efficiently allocates tasks to servers with the fewest active connections, encouraging balanced resource use. However, it might not consider server capacity or response time.
3.	**Weighted Round Robin**	Assigns weights to servers based on their abilities, aiming for better workload delivery. It is effective when servers have different capabilities but might not dynamically adapt to varying conditions.
4.	**IP Hashing**	This technique sends user requests to servers based on the user's IP address, so the same user always connects to the same server. It helps in keeping the session active but sometimes it can overload a server if many users from one IP address send several requests.
5.	**Adaptive Load Balancing**	These algorithms constantly monitor system health and adapt to varying conditions, reallocating workloads dynamically. They offer better fault tolerance by proactively responding to failures or variations in load.

4.0 Performance Metrics and Assessment:

The effectiveness of a load balancing approach is dignified by how efficiently it allocates traffic among available resources while maintaining high performance, availability and scalability. The following factors are commonly used to evaluate load balancing performance [14, 15]:

1. Response Time Analysis:

The effectiveness of a load balancing approach is measured by how efficiently it allocates traffic among available resources while maintaining high performance, availability and scalability. The following parameters are commonly used to evaluate load balancing performance.

2. Resource Utilization Metrics:

Server/Resource Usage: It Evaluates resource utilization metrics like CPU, memory, and network utilization across load balanced servers. Ensure balanced utilization to prevent underutilization or overloading.

3. Traffic Distribution Analysis:

Request Count: It analyzes the dispersal of incoming requests across backend servers or instances. Check if the load balancer uniformly allocates traffic according to well-defined algorithms or policies.

4. Health Check Monitoring:

Backend Health Status: It monitors the health status of backend resources as well as evaluates how efficiently the load balancer identifies and sends traffic away from nonworking or failed servers.

5. Failure Recovery Evaluation:

Failure Simulation: It simulates server failures or unreachability situations to assess the load balancer's capability to reroute traffic to healthy servers promptly and proficiently.

6. Scalability Assessment:

Auto-Scaling Performance: It tests the load balancer's capability to scale resources based on traffic variations and assesses how well it accommodates improved load and allocates traffic across dynamically scaled resources.

7. Load Testing:

Stress Testing: It Performs load tests by simulating peak traffic situations to define the load balancer's performance under heavy load conditions. It Measures response times and observe how well it succeeds to manage the load.

8. Logging and Monitoring Analysis:

- *Log Analysis*: It involves review load balancer logs and analyzes traffic patterns, errors and throughput to classify any bottlenecks or variations in load distribution.

9. Geographical Distribution Evaluation:

Geolocation Performance: It assesses the load balancer's competence in directing traffic based on geographic proximity. It measure latency for users in diverse regions to ensure best routing.

10. Cost-Efficiency Analysis:

Resource Cost: It calculates the cost-effectiveness of load balancing strategies. Consider resource utilization against the associated costs to guarantee effective use of resources.

11. Real User Monitoring (RUM):

User Experience Metrics: It implement real user monitoring tools to collect data on actual user understanding, including page load times, transaction performance and user communications affected by load balancing.

5.0 Performance Simulation:

Cloud Analyst tool is used for measuring the performance of three algorithms on the Windows 10 operating system using the Eclipse IDE and Java 8.

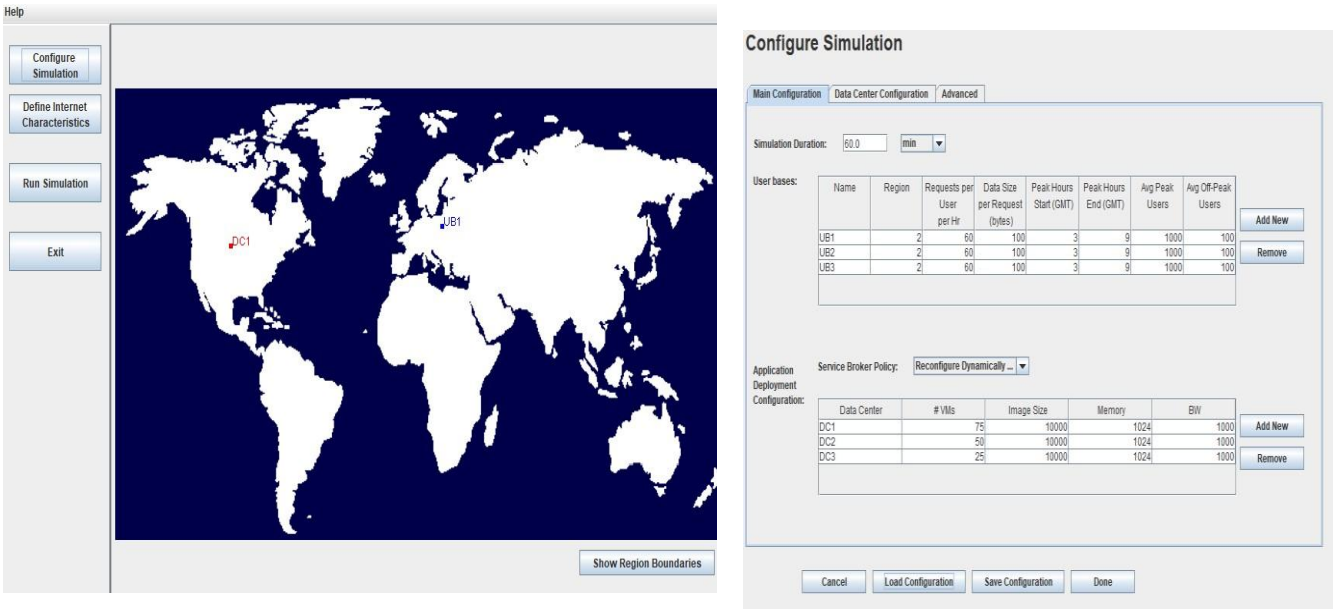


Figure 1: Cloud Analyst Simulator

CloudAnalyst is a strong large-scale cloud simulator that makes it easy to simulate a accurate cloud environment with a large-scale Internet load. It is successor of CloudSim based on the graphical interface and also gives a simulation result in a graphical form [16].

5.1. Features of Cloud Analyst

Main features of the Cloud Analyst Simulator are [17]:

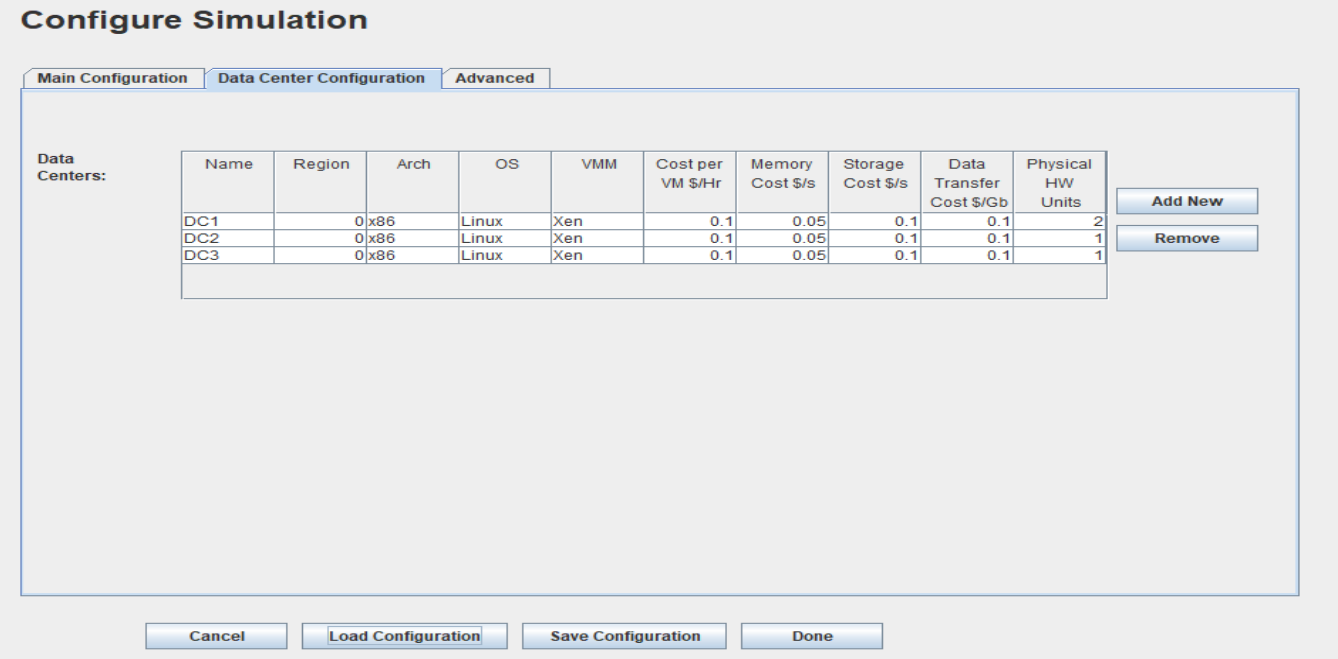
- **Region:** The Cloud Analyst toolkit splits the world into six regions and each region analogous to a continent: Australia, South America, North America, Asia, Africa, and Europe. This separation helps in simulating cloud deployments across diverse geographical areas.
- **User Base**: In Cloud Analyst, a group of users is considered as a single unit that is called User Base. These users are involved in the simulation procedure and responsible for creating traffic. This module helps in simulating user behavior and workload patterns on cloud services.
- **Data Center Controller (DCC)**: Each instance of CloudSim is associated with a one Data Center Controller (DCC). The DCC controls the behavior and supervision of the simulated data center within a specific region. It handles works like resource allocation, scheduling and communication with other modules.
- **Virtual Machine (VM) Load Balancer:** The Data Center Controller (DCC) starts communication with the VM Load Balancer (VMLB) to handle the spreading of virtual machine tasks. VMLB plays a critical role in defining how cloudlets are allocated to VMs. Presently, Cloud Analyst offers three VMLB options: Round Robin, Throttled and Active Monitoring Load Balancer.
- **Cloud Application Service Broker:** It has the responsibility of managing traffic flow among data centers and user bases and implements several policies. Three service broker policies are available within the Cloud Analyst simulator:
  - Optimizing response time
  - Selecting the closest data center
  - Dynamically reconfiguring resources.

Configuration:

Numerous configurations were used to calculate the performance of various algorithms across changing numbers of data centers and user bases. Table 2 outlines further fundamental configuration parameters.

Table-2

Parameter Name	Value
User Grouping Factor in User Base	1000
Request Grouping Factor	250
Executable instruction length/request	250



Round Robin, Equally Spread Current Execution Load and Throttled Load Balancing strategies are simulated by means of the Cloud Analyst Simulation tool on three diverse service broker strategies that are closest data center, Enhanced Response Time and Reconfigure Dynamically with Load Balancer. The Strategies were compared on two parameters i.e. over all response time and total data transfer cost. In conclusion, the Throttled load balancing algorithm proves higher performance.

Table-3 Simulation Performance Measurement

Service Broker Policy	Closest Data Center			Optimized Response Time			Recofigure dynamically with load balancer		
Load Balancing Policy	Round Robin			Round Robin			Round Robin		
	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)
Overall response time	390.92	267.33	599.14	364.77	268.21	608.27	412.64	282.22	686.46
Total data Transfer cost	91.96	11.76	304.01	64.74	11.76	280.76	113.6	13.79	363.01

Service Broker Policy	Closest Data Center			Optimized Response Time			Recofigure dynamically with load balancer		
Load Balancing Policy	Equally Spread Current Execution Load			Equally Spread Current Execution Load			Equally Spread Current Execution Load		
	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)
Overall response time	393.91	265.86	604.02	355.65	262.64	601.77	406.04	278.24	698.77
Total data Transfer cost	96.94	11.76	296.01	55.63	11.76	268.39	107.4	14.54	381.76

Service Broker Policy	Closest Data Center			Optimized Response Time			Recofigure dynamically with load balancer		
Load Balancing Policy	Throttled			Throttled			Throttled		
	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)	Avg(ms)	min(ms)	Max(ms)
Overall response time	292.33	252.17	631.28	374.82	268.21	624.27	423.13	274.24	695.65
Total data Transfer cost	93.5	11.76	309.27	75.05	11.88	289.76	123.94	13.79	376.64

6.0 Challenges and Future Directions:

Certainly! Load balancing continues to evolve to address emerging challenges and capitalize on future opportunities. Here are key challenges and potential future directions in load balancing:

Challenges:

1. Multi-Cloud and Hybrid Environments:

Challenge: Managing load balancing across multiple clouds or hybrid environments introduces complexities in interoperability, consistent strategies and traffic routing.

Future Focus: Developing standardized load balancing frameworks and tools that seamlessly operate across diverse cloud environments.

2. Dynamic Workload Variability:

Challenge: Handling random traffic patterns that includes sudden spikes or drops in demand needs load balancers to dynamically regulate resources.

Future Focus: Improving auto-scaling abilities and machine learning-driven load balancing for real-time adaptive resource distribution.

3. Security and Compliance:

Challenge: Guaranteeing load balancers obey security protocols and industry-specific guidelines

without cooperating performance.

Future Focus: Applying extra healthy security measures within load balancing mechanisms and maintaining compliance across various environments.

4. Edge Computing and IoT:

Challenge: Enhancing load balancing for edge computing as well as IoT devices, where latency, bandwidth and resource constraints are serious concern.

Future Focus: Creating focused load balancing solutions personalized for edge computing and integrating edge-aware routing and resource allocation strategies.

7.0 Conclusion

This paper presents a detailed overview of cloud load balancing methods, their implementation, and performance assessment and emerging guidelines in this area. By investigating different methods, the research paper highlights the role of effective load balancing in refining the competence of cloud environments. The results show that the Throttled load balancing algorithm performs better than other considered approaches. The scope of this work can be further extended by testing added service broker strategies and comparing more load balancing strategies. In future research, an intelligent architecture integrating big data and machine learning techniques will be proposed to further improve load balancing in cloud computing systems.

References:

[1] M. Agarwal, D.G.M.S. Srivastava "Cloud Computing: A Paradigm Shift in the Way of Computing", Int. J. Mod. Educ. Comput. Sci., 9 (12) (2017), pp. 38-48,

[2] D. Lowe, B. Galhotra "An overview of pricing models for using cloud services with analysis on Pay-Per-Use model" Int. J. Eng. Technol., 7 (3) (2018)

[3] "Shafiq, N.Z. Jhanjhi, Azween Abdullah et al
Load Balancing Techniques in Cloud Computing Environment: A Review " in journal of King Saud University - Computer and Information Sciences (2021)

[4] Buyya R, Vecchiola C, Selvi ST "Mastering cloud computing: foundations and applications programming" Morgan Kaufmann, USA, (2013)

[5] Mishra SK, Sahoo B, Parida " Load balancing in cloud computing: a big picture" in J King Saud Univ Comp Infor Sci:1–32(2018)

[6] Afzal, S., Kavitha, G. "Load balancing in cloud computing – A hierarchical taxonomical classification" in J Cloud Comp 8, 22 (2019)

[7] J. Noor, M. N. H. Shanto, J. J. Mondal, M. G. Hossain, S. Chellappan and A. B. M. A. Al Islam, "Orchestrating Image Retrieval and Storage Over a Cloud System," in *IEEE Transactions on Cloud Computing*, vol. 11, no. 2, pp. 1794-1806, 1 April-June 2023

[8] X. Wei and Y. Wang, "Popularity-Based Data Placement With Load Balancing in Edge Computing," in *IEEE Transactions on Cloud Computing*, vol. 11, no. 1, pp. 397-411, 1 Jan.-March 2023

[9] F. Jamal and T. Siddiqui, "Comparative Analysis of Load Balancing Techniques in Cloud Computing, Based on LB Metrices," 2021 5th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 2021, pp. 1-5

[10] Ashawa, M., Douglas, O., Osamor, J. et al. RETRACTED ARTICLE: Improving cloud efficiency through optimized resource allocation technique for load balancing using LSTM machine learning algorithm. J Cloud Comp 11, 87 (2022)

[11] H. Rai, S. K. Ojha and A. Nazarov, "Cloud Load Balancing Algorithm," in 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2020, pp. 861-865

[12] Pranjal Upadhyay, Krishna Kumar Sharma, Rishu Dwivedi, Pradeep Jha, "A Statistical Machine Learning Approach to Optimize Workload in Cloud Data Centre", 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), pp.276-280, 2023

[13] GUO Qiaoyu, HE Weizhen, ZENG Wei, XIAO Yuqiang, "Design and Implementation of High-Performance Software Load Balancer for Cloud Service", 2021 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE), pp.175-181, 2021

[14] Priyal Ghetiya, Prof. Dhaval Nimavat, "Enhancing Response Time of Cloud Resources Through Energy Efficient Cloud Scheduling Algorithm", International Journal of Scientific Research in Science, Engineering and Technology, pp.354, 2022.

[15] Adhikari, M., Amgoth, T., Srirama, S.N., " A survey on scheduling strategies for workflows in cloud environment and emerging trends" ACM Comput. Surv. (CSUR) 52 (4), 1–36.

[16] Hamdani, M., Aklouf, Y. and Bouarara, H. A. Improved fuzzy Load-Balancing Algorithm for Cloud Computing System. In Proceedings of the Proceedings of the 9th International Conference on Information Systems and Technologies (Cairo, Egypt, March 24 - 26). ACM, 2019.

[17] Rathore, J., Keswani, B. and Rathore, V. S. Analysis of Load Balancing Algorithms Using Cloud Analyst.Springer Singapore, City, 2019.