A Comparative Study of Traditional, Dynamic and Elastic Load Balancing Algorithms in Cloud Computing

Soumen Swarnakar
Faculty
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Lagnajita Mandal
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Deepanwita Sarkar
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Sayan Bhattacharya
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Sourav Chakraborty
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Md. Faiz Ansari
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

Rishav Datta
Student
Department of Information Technology
Netaji Subhash Engineering College
Kolkata, India

ABSTRACT
Elastic Load Balancing (ELB) in cloud computing is essential for managing dynamic workloads by distributing traffic across multiple computing resources to ensure high availability, scalability, and performance. Various ELB algorithms have been developed to meet the evolving demands of cloud environments. Traditional methods such as Round Robin and Least Connection offer basic traffic distribution, while more advanced strategies like resource-based, weighted, and latency-aware algorithms provide intelligent routing based on server health, capacity, and network conditions. Recent advancements incorporate AI and machine learning to enable predictive and adaptive load balancing, allowing systems to automatically respond to traffic fluctuations and optimize resource usage in real-time. Additionally, geolocation-based routing enhances user experience by directing requests to the nearest or fastest nodes, particularly valuable in edge computing and global service delivery. These algorithms are often integrated with infrastructure-as-code tools and DevOps workflows for enhanced automation and observability. As cloud applications become more complex and distributed, ELB algorithms continue to evolve, focusing on greater intelligence, flexibility, and cross-platform compatibility to support real-time applications, IoT systems, and serverless architectures. Unlike traditional load balancers that rely on hardware and static rules, ELB dynamically adjusts its capacity based on real-time traffic changes. In this paper a study on cloud load balancing has been discussed along with comparative study with dynamic and elastic load balancing. Future prospect of ELB has also been included in this research article.

Keywords
Cloud Computing, Load Balancing, Static Load Balancing, Dynamic Load Balancing, Elastic Load balancing.

1. INTRODUCTION
Cloud computing has revolutionized IT infrastructure by providing scalable, on demand computational power, storage, and services, enabling businesses to optimize their operations and reduce costs. However, the rapid growth and increasing complexity of cloud environments have introduced challenges in maintaining efficient resource utilization, system reliability, and performance consistency. Load balancing plays a vital role in overcoming these challenges by distributing workloads across multiple servers, preventing system overloads, and enhancing overall efficiency. Elastic load balancing, in particular, offers dynamic adjustment to fluctuating workloads, ensuring seamless scalability and high availability. This paper examines various elastic load balancing techniques, including their methodologies, advantages, and limitations. It also analyzes their impact on cloud environments, particularly in terms of resource optimization, fault tolerance, and system responsiveness. By reviewing key strategies and their effectiveness, the paper aims to provide valuable insights into the evolving landscape of load balancing in cloud computing. ELB operates across multiple Availability Zones to offer high fault tolerance and minimal latency. It comes in three main types: Application Load Balancer (ALB) for Layer 7 content-based routing, Network Load Balancer (NLB) for high-performance Layer 4 processing and Gateway Load Balancer (GLB) for integrating virtual security appliances like firewalls.

This article has been divided into nine sections. Related work is discussed in Section 2. Different types of cloud load balancing algorithms have been discussed in section 3.Section 4 describes some comparative study on already proposed cloud load balancing algorithms, whereas section 5 discusses a comparative study on some conventional cloud load balancing algorithms. Section 6 describes a comparative study on traditional, dynamic and elastic cloud load balancing algorithms whereas section 7 discusses on functionality and benefits of elastic load balancing. Future work has been discussed in Section 8 whereas conclusion has been drawn in section 9.

2. RELATED WORK

Several research studies have explored load balancing techniques in cloud computing, emphasizing different methodologies to enhance system performance, scalability, and reliability.

Swarnakar et al. (2023) [1] introduced a multi-agent-based virtual machine (VM) migration technique for dynamic load balancing. Their method uses autonomous agents to monitor system performance and make real-time VM migration decisions, effectively balancing the load while minimizing migration overhead. This approach is well-suited for cloud environments with fluctuating workloads.

Swarnakar et al. (2020 [2]) presented an improved dynamic load balancing approach that considers system performance, resource availability, and network conditions. This method enhances resource utilization, reduces server overload risks, and improves system responsiveness and fault tolerance

Agarwal and Raj (2022) [3] proposed a static load balancing algorithm aimed at optimizing resource allocation and minimizing response time. While their scheduling mechanism efficiently distributes tasks across virtual machines, it lacks adaptability to dynamic workload variations.

Sharma et al. (2021) [4]in a paper published in IEEE Transactions on Cloud Computing, introduced a machine learning-enhanced ELB system that predicts traffic trends and adjusts resource provisioning accordingly. Their research proved superior to traditional threshold-based models, achieving lower latency and better throughput. These works demonstrate that elastic strategies not only scale effectively but also boost security, performance, and cost savings.

.**Shahid et al. (2020)[5]** developed a fault-tolerant load balancing technique that combines predictive analytics with dynamic scheduling. Their hybrid approach—merging static and dynamic strategies—showed improved resource utilization and better prevention of system failures caused by uneven task distribution.

Tadapaneni (2020) [6] conducted a survey of various load balancing strategies, focusing on the strengths and weaknesses of reactive versus proactive techniques. The study highlighted that self-adaptive algorithms, inspired by natural systems like honeybee foraging and ant colony optimization, offer enhanced flexibility and fault tolerance in cloud environments.

Manjula and Sundaram (2020) [7]proposed a dynamic load balancing algorithm using machine learning to predict workload patterns. Their technique significantly improved throughput and response time, though it required additional computational resources for predictive processing.

Chaudhury et al. (2020)[8] addressed limitations in traditional static load balancing by introducing an enhanced round-robin algorithm. Their version improved task fairness and reduced waiting times but faced challenges with resource-intensive workloads.

Upadhyay et al. (2018)[9]explored clustering-based load optimization methods, demonstrating that clustering enhances response time and reduces system overhead, making it effective for large-scale distributed cloud systems.

Mishra et al. (2018) [10] provided a comprehensive review of load balancing techniques, categorizing them into static and dynamic approaches. Their study emphasized that dynamic techniques are better suited for cloud environments due to their ability to adapt to real-time changes in workload distribution.

A foundational analysis by **M. Singh and S. Chana (2016) [11],** examined the transition from static to elastic load balancing in cloud infrastructure. Their findings underscored how elasticity promotes real-time resource tuning, enhances quality of service, and avoids inefficient resource usage. They highlighted the necessity of elastic methods to fulfill SLAs and ensure reliability in unpredictable environments.

Agarwal and Jain (2016) [12] evaluated the impact of combining ELB with Auto Scaling, showing that this architecture significantly enhances application availability and cost-efficiency compared to static provisioning.

3. CLOUD LOAD BALANCING ALGORITHMS

Load balancing in cloud computing helps distribute tasks across multiple processors to use resources efficiently. To manage this, a good scheduling algorithm is needed. It ensures tasks are assigned properly to reduce execution time and improve resource use. Different load balancing algorithms help share the load across the network. The main goals of load balancing are:

a. Increase network performance.

b. Reduce system workload.

c. Ensure reliable service.

d. Allow the system to scale easily.

e. Use resources efficiently.

f. Handle faults effectively.

The general structure of cloud load balancing algorithm is shown below in figure 1.

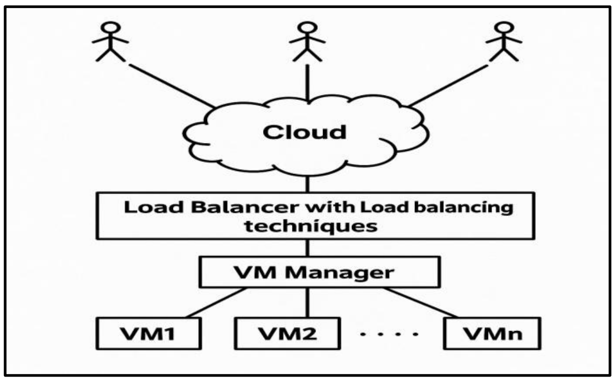

Fig. 1. General Structure of Load balancing in Cloud Environment

Cloud Computing Load Balancing can be divided into two categories as given below.

- Static load balancing

- Dynamic Load Balancing

a. Static Load Balancing

Static load balancing uses prior information about all the characteristics of the task, the computing resources, and communication network memory are known and provided. Static load balancing algorithms are non-preemptive types.

Static load balancing assigns tasks based on system resources like memory, processing power, and storage without checking the system's current status. This makes it easy to implement but unsuitable for large or distributed systems since it may lead to an uneven workload. It also cannot detect new servers joining during runtime. A method called "dynamic exchange" helps distribute workloads across available machines. In static load balancing, tasks are divided into equal units (tokens) and assigned to processors, but once allocated, the load remains unchanged. This method works well for small, fixed systems but struggles in dynamic environments where workloads change. Static load balancing schedules tasks based on estimated completion time, giving priority to shorter tasks while longer tasks wait. However, some tasks may be delayed for too long. A common method, round-robin scheduling, processes tasks in a fixed sequence, making it useful for web servers but less effective for cloud environments with diverse workloads.

b. Dynamic Load Balancing

Static load balancing methods do not work well in systems that change dynamically and need real-time resource updates. Since they do not consider the system's current state, dynamic load balancing is required for cloud computing.

Dynamic load balancing strategies can be grouped into:

● **Overall Load Balancing** – Balances workload across all resources.

● **Natural Phenomena-Based Load Balancing** – Inspired by natural processes.
● **Combination Load Balancing** – Uses multiple techniques together.
● **Agent-Based Task Distribution** – Uses software agents to assign tasks.
● **General Task Distribution** – Allocates tasks efficiently.
● **Cluster-Based Task Distribution** – Balances load within groups of resources.
Dynamic load balancing adapts based on the system's current state. It works well in cloud environments, handling resource changes efficiently without needing prior system knowledge.

The following are prominent elastic load balancing algorithms proposed in research articles:

- **Round Robin Algorithm:** Round Robin is the simplest algorithm to work in a circular fashion. With this nature of the algorithmic rule computer hardware allocates a time quanta or time slice to execute a task on each node. Once a VM is assigned a task it moves to the end of the list. Round Robin provides higher performance than FCFS. If the time slice is simply too huge then the round robin behaves like FCFS and if the time slice is simply too short then it'll increase context modification inside the round robin algorithm. At some point of time some nodes may remain idle while others are heavily loaded.

- **Weighted Round Robin:** It is the changed version of Round Robin. Tasks are assigned according to the capacity of the VM the higher VM will get the more number of tasks. Server can be allocated weight, an integer value which represents the processing power of a VM.

- **Throttled Load Balancing:** A table is generated in this algorithm that includes the virtual machines as well as the existing state (available/busy). If a specific task is allocated to a virtual machine, a request is made to the control unit within the data center, which will look for the ideal VM suit with respect to their abilities to achieve the required task. The load balancer will send -1 back to the data center if an appropriate VM is not available.

The standard throttled algorithm always starts searching for an idle VM from the beginning of the table, causing some VMs to remain unused. To improve this, researchers proposed modified versions:

i. **Modified Throttled Algorithm** – Instead of starting from the first VM each time, it selects the next VM in line, ensuring better resource usage.

ii. **Efficient Throttled Algorithm** – Combines Throttled, Equally Spread Current Execution (ESCE), and Round Robin methods for better performance using a data structure to store VM information.

iii. **Hashmap-Based Throttled Algorithm** – Uses a Hashmap index to quickly find and allocate VMs, making it faster than the basic throttled method.

iv. **Divide-and-Conquer Throttled Algorithm (DCBT)** – A hybrid approach that divides jobs equally among VMs, optimizing resource usage and reducing task execution time.

These improvements make **VM selection more efficient** and help in better load balancing compared to the original throttled algorithm.

● **Active Monitoring Load Balancing:** It is a type of dynamic load technology. This technology obtains information relevant to each VM and to the number of requests that are presently allocated to each of them. The Data Center Controller (DCC) scans the VM index table after receiving a new request to determine the VM that is least loaded or idle. First-come first serve concept is employed by this algorithm to allocate load to the VM that has the smallest index number for more than two servers. The VM ID is sent back by the AMLB algorithm to the DCC which then sends the request to the VM represented by that ID. The AMLB is informed about the new allocation by the DCC and it is sent the cloudlet. Once the task is completed, the information is sent to the DCC and the VM index table is reduced. When a new request is received, it goes over the table again using load balancer and then the process allocation occurs. Figure 4 presents an illustration of AMLB.

● **Honey Bee Foraging Algorithm:** The working of this algorithm is influenced by the way bees behave when looking for honey, because its main objective is to divide the workload on the VM, considering the lack of excessive resource utilization and lack of under-usage of resources. This algorithm works by choosing a VM that fulfills two main requirements. Fewer tasks are allocated to this VM compared to those assigned to other VM machines. The time taken by VM to process falls within the average processing time taken by all other VMs.

● **Min-Min Algorithm:** This algorithm is easy to use and works at a faster pace. In addition, it improves performance and consists of a series of tasks. The time taken to execute the task is computed and allocated to VMs on the basis of the smallest completion time for the existing tasks. The process will continue till it is ensured that each task has been allocated to the VM . Because of the existence of a greater number of smaller tasks, this algorithm performs better compared to if there were bigger tasks. However, this will lead to starvation because of giving priority to smaller tasks and deferring the bigger tasks.

● **Max-Min Algorithm:** This algorithm is quite similar to the Min-Min Load Balancing, based on the calculation time. In this algorithm, all existing tasks are sent to the system, after which the calculation is carried out for determining the least time to complete each of the given tasks. The selected task then has the maximum time to be completed and will be allocated to the relevant machine. A comparison of the performance of this algorithm with the Min-Min algorithm shows that the Max-Min algorithm is better because there is just one large task in the set, which means that the Max-Min algorithm will carry out the shorter tasks alongside the larger task .

4. COMPARATIVE STUDY OF SOME EXISTING LOAD BALANCING ALREADY PROPOSED ALGORITHMS

Algorithm	Approach	Description	Strong Points	Weak points
Multi Agent Based Dynamic Load Balancing (MADLB) [1]	Dynamic	MADLB is a technique that helps balance workloads in cloud computing. It uses multiple agents that communicate with each other to monitor server loads, share information, make decisions to move tasks or distribute loads.	**Decentralization** – By distributing the decision-making among multiple agents, MADLB eliminates bottlenecks and improves fault tolerance. **Real-time Adaptation** – Capable of adjusting load distribution dynamically as workloads change. **Improved Scalability** – Suitable for large-scale distributed systems. **Enhanced Resource Utilization** – Agents work together to ensure balanced utilization of computational resources.	**Communication Overhead** – Inter-agent communication can lead to increased network traffic, especially in large systems. **Complex Implementation** – Designing cooperative agents with robust decision-making logic can be challenging. **Synchronization Issues** – Ensuring consistency and avoiding conflicts during task migration may require sophisticated synchronization mechanisms. **Resource Consumption by Agents** – Agents themselves consume computational resources, potentially affecting system performance.
Improved Dynamic Load Balancing Approach in Cloud Computing (IDLBA) [2]	Dynamic	IDLBA is an algorithm that: shares tasks across computers (VMs) in the cloud, monitors computer resources (CPU, memory etc.) in real-time predicts future workload and balances tasks proactively, moves tasks to underused computers to avoid overload and makes decisions based on current data for efficient task scheduling	**Dynamic Resource Management** – Continuously adjusts task allocation based on real-time data. **Integrated Monitoring** – Uses real-time system data (CPU, memory, etc.) to make informed decisions. **Improved Performance** – Reduces task response time and improves resource utilization. **Prevents Overload** – Predictive model helps avoid VM bottlenecks before they happen.	**Centralized Scheduling** – While adaptive, relying on a central system for decisions could become a bottleneck or point of failure. **Overhead of Monitoring** – Continuous system monitoring may consume additional resources. **Scalability Constraints** – May face issues in very large cloud environments due to central control. **No Multi-agent Collaboration** – Unlike MADLB, this algorithm doesn't support distributed intelligence.

5. COMPARATIVE STUDY OF SOME CONVENTIONAL CLOUD LOAD BALANCING EXISTING ALGORITHMS

Algorithm	**Approach**	**Description**	**Advantages**	**Disadvantages**
Round Robin algorithm	Static	Request is allocated for a fixed period of time	Equal distribution of workload	Process is notKnown in advance. For a larger task context switching increases.
Weighted Round Robin	Static	According to the processing capacity of VM weight is assigned.	Optimal resource utilization	Processing time not taken into consideration
Throttled LB Algorithm	Reactive	Maintain state of VM busy or idle	Evenly distribution of load	Does not consider current state of the VM
Active Monitoring Load Balancing	Reactive	Least loaded VM is allocated with the request	Existing load istaken intoconsideration	VM processingpower is not considered
Honey Bee Foraging	Dynamic	Distributed load balancing for self-organization	Well suited forheterogeneousenvironment	Increase in resource not Increases efficiency
Max Min Algorithm	Static	Job with higher execution time executed first.	Shorter makespan as compared toMin- Min	Shorter jobs have to wait
Min-Min Algorithm	Static	Select task with least execution time	Simple to execute	Does not consider existing load

6. COMPARATIVE STUDY OF TRADITIONAL, DYNAMIC AND ELASTIC LOAD BALANCING ALGORITHMS

Feature	**Traditional Load Balancing**	**Dynamic Load Balancing**	**Elastic Load Balancing (Cloud)**
Scalability	Limited to physical or static resources	Scales based on real-time data, but still bounded by infrastructure	Auto-scales with demand using virtual resources
Deployment	On-premise, hardware-based	Software-defined, can be hybrid	Fully integrated with cloud platforms (AWS, Azure, GCP)
Flexibility	Static configurations	Adjusts with policies or metrics	Elastic, reactive, policy + AI driven
Cost	High upfront cost, maintenance	Lower than traditional, but still infra-dependent	Pay-as-you-go, no infra management
Availability	Single point of failure risks	Better with redundancy	Highly available, geo-redundant
Integration	Manual or semi-automated	Needs external monitoring tools	Built-in with CI/CD pipelines, monitoring, auto scaling

7. FUNCTIONALITY AND BENEFITS OF ELASTIC LOAD BALANCING

ELB architecture is closely connected with monitoring and orchestration tools in the cloud. It conducts automatic health assessments of its registered targets and redirects traffic away from failing ones without human involvement. In combination with Auto Scaling Groups, ELB dynamically adjusts resource levels to maintain responsiveness and efficient resource usage. Unlike basic algorithms like round-robin, which rotate requests without considering system state, ELB leverages intelligent routing based on real-time data like latency, resource usage, and instance health. This leads to faster responses and higher application uptime. Figure 2 illustrates

Cloud Elastic Load Balancing distributing traffic from multiple clients to EC2 instances. Clients (Client 1 to Client n) connect to a central Elastic Load Balancer via bi-directional arrows.The Load Balancer routes requests to EC2 instances based on availability and traffic.

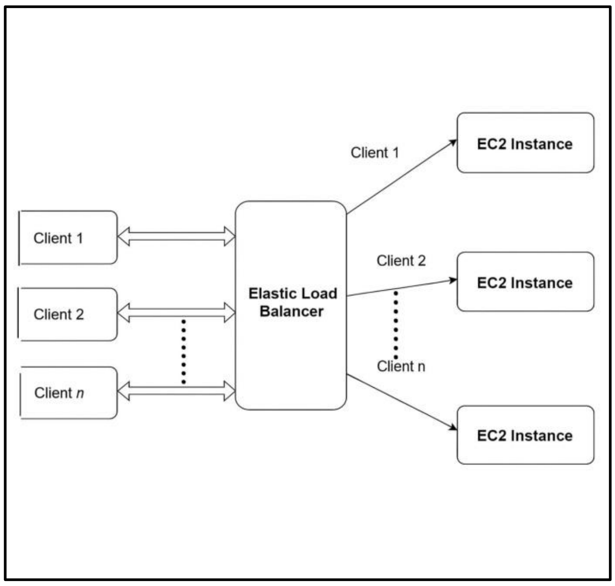

Fig. 2. AWS Elastic Load balancing

8. FUTURE PROSPECT

Cloud elastic load balancing remains a crucial area for research and innovation, with several opportunities for future advancements. One key direction is the integration of AI and machine learning to develop intelligent load-balancing algorithms that predict workload patterns and optimize resource allocation dynamically. Another important area is energy-efficient load balancing, where strategies can be designed to minimize power consumption in cloud data centers while maintaining performance. The future of Elastic Load Balancing (ELB) in cloud computing is marked by advanced automation and AI integration, enabling predictive traffic management and self-healing capabilities to maintain service continuity. As edge computing and multi-cloud environments expand, ELBs will evolve to intelligently route traffic based on not just server load, but also network latency, user proximity, and resource availability across geographically distributed systems. With the rise of serverless and event-driven architectures, ELBs are expected to support ephemeral compute resources like AWS Lambda, dynamically routing based on event characteristics. Seamless integration with DevOps pipelines and infrastructure-as-code tools (e.g., Terraform, Pulumi) will further enhance observability and performance optimization. These advancements position ELBs to handle increasingly complex use cases—from real-time applications like streaming and gaming to large-scale IoT ecosystems—through adaptive, scalable, and efficient load distribution.

9. CONCLUSION

Cloud elastic load balancing is a crucial component of cloud computing that enables organizations to distribute incoming traffic across multiple servers to improve responsiveness, reliability, and scalability of applications. With its ability to detect and redirect traffic away from unhealthy servers, elastic load balancing also enhances application availability and fault tolerance. Overall, cloud elastic load balancing is an essential tool for organizations seeking to build highly available, scalable, and responsive applications in the cloud. Elastic Load Balancing (ELB) is a key component in modern cloud architecture that ensures efficient distribution of incoming traffic across multiple computing resources. It plays a vital role in enhancing application availability, performance, and fault tolerance. The primary benefit of ELB lies in its ability to automatically scale and adapt to changing traffic conditions, which helps maintain a seamless user experience even during peak loads.

By monitoring the health of resources and rerouting traffic away from failed instances, ELB minimizes downtime and ensures that only healthy targets handle requests. This results in better resource utilization, improved system reliability, and reduced operational overhead. Additionally, ELB supports different load balancing strategies (like round robin, least connections, and IP-hash) and protocols (HTTP, HTTPS, TCP, UDP), making it flexible for various application needs.

In real-world scenarios, ELB is widely used across industries. E-commerce platforms rely on it to manage traffic during high-demand periods. Streaming services use it to distribute content efficiently, ensuring smooth playback. Cloud providers like AWS and Azure incorporate ELB in their services to offer scalable and resilient infrastructures. It is also used in finance, healthcare, and mobile app backends to ensure secure and uninterrupted services.

In summary, Elastic Load Balancing offers a robust, automated solution to manage application traffic, making it indispensable for building scalable, high-performance, and fault-tolerant systems in today's digital world.

10. REFERENCES

[1] Soumen Swarnakar, Chandan Banerjee, Joydeep Basu, Debanjana Saha, "A Multi-Agent-Based VM Migration for Dynamic Load Balancing in Cloud Computing Cloud Environment", International Journal of Cloud Applications and Computing. 13. 1-14. 10.4018/IJCAC.320479, 2023.

[2] S. Swarnakar, R. Kumar, S. Krishn and C. Banerjee "Improved Dynamic Load Balancing Approach in Cloud Computing," IEEE 1st International Conference for Convergence in Engineering (ICCE), Kolkata, 2020, pp. 195-199, 2020, doi: 10.1109/ICCE50343.2020.9290602.

[3] Abhay Kumar Agarwal, Atul Raj, "A New Static Load Balancing Algorithm in Cloud Computing", International Journal of Computer Applications (0975 – 8887), Volume 132 – No.2, December, 2022.

[4] A. Sharma, R. Kumar, and P. Varma, "Intelligent Elastic Load Balancing Using Machine Learning in Cloud Environments," IEEE Transactions on Cloud Computing, vol. 9, no. 4, pp. 1250–1263, Dec. 2021. doi: 10.1109/TCC.2020.2973229.

[5] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, S. Musa, "A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach," IEEE Access, 8, 130500–130526, 2020, doi:10.1109/ACCESS.2020.3009184.

[6] N. R. Tadapaneni, "A Survey of Various Load Balancing Algorithms in Cloud Computing," Int. J. Sci. Adv. Res. Technol., vol. 6, 2020.

[7] Manjula K., S. Meenakshi Sundaram, Improved and Efficient Dynamic Load Balancing Algorithm in Cloud

Based Distributed System, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-5, January 2020.

[8] Chaudhury, K. S., Pattnaik, S., Moharana, H. S., & Pradhan, S., "Static load balancing algorithms in cloud computing: challenges and solutions", In the International Conference on Soft Computing and Signal Processing (pp. 259-265). Springer, Singapore, 2020.

[9] S. K. Upadhyay, A. Bhattacharya, S. Arya, T. Singh, "Load optimization in cloud computing using clustering: a survey," Int. Res. J. Eng. Technol, 5(4), 2455–2459, 2018.

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

[11] M. Singh and S. Chana, "Elastic Load Balancing in Cloud Computing: Algorithms, Techniques and Challenges," The Journal of Supercomputing, vol. 72, no. 8, pp. 3210–3240, Aug. 2016. doi: 10.1007/s11227-015-1507-0.

[12] Agarwal, S., & Jain, A., "Efficient Management of Elasticity in Cloud Applications Using Load Balancing." International Journal of Computer Applications, 145(7), 20-25,2016.