International Journal of Distributed and Cloud Computing
12 (1) 2024, 08-16
http://www.publishingindia.com/ijdcc

A Comparative Assessment of Load Balancing Techniques in Cloud Computing: A Review

John Abiodun Oladunjoye[1] and Egwom Onyinyechi Jessica[2*]
[1]Senior Lecturer, Department of Computer Science, Federal University, Wukari, Nigeria.
Email: oladunjoye.abbey@yahoo.com
[2]Assistant Lecturer, Department of Computer Science, Federal University, Wukari, Nigeria.
Email: egwom@fuwukari.edu.ng
[*]Correspondence Author

Abstract: Cloud computing is a resilient framework that enables users and organizations to procure services tailored to their needs. This model encompasses many services, including storage, deployment platforms, seamless access to web services, and more. However, load balancing poses a common challenge in the cloud, impacting the ability to uphold application performance aligned with the required quality of service (QoS) expected by enterprises from cloud providers. Achieving an equitable distribution of workloads among servers proves challenging for cloud providers. An effective load-balancing technique should optimize resource utilization in virtual machines, ensuring high user satisfaction. This paper presents a comprehensive review of various load balancing techniques in terms of static and dynamic and the overall performance of these algorithms using some popular performance metrics. Based on the performance measures, Cuckoo Search and Firefly was the most effective algorithm. This research makes a contribution by describing current methodologies and by helping researchers identify research issues connected to load balancing, to decrease response times and prevent server failures.

Keywords: Algorithms, Cloud computing, Load balancing, Performance metrics, Resource management, Virtual machine.

I. Introduction

Cloud computing technology is undergoing significant growth in network technology, propelled by advancements in communication technology, widespread Internet usage, and its capacity to tackle large-scale challenges [1], [2]. The term cloud computing according to S. Kaur, and V. Pandey [3] refers to the distribution of various computing services over the internet (cloud), encompassing servers, databases, networks, and software. Cloud computing facilitates the development of new applications and services, data backup and recovery, hosting of websites and blogs, streaming of audio and video, and the analysis of data patterns for predictive purposes [4]. Cloud users can leverage hardware and software applications as resources through the Internet.

The adoption of cloud computing has brought about a significant shift in how businesses perceive their available resources. These computers are dynamically allocated and presented as a unified computing resource or resources, guided by negotiated service-level agreements between the service provider and consumers [5]. Cloud computing also referred to as distributed computing, introduces innovative perspectives for both service providers and end consumers. To accommodate these evolving

aspects, novel design strategies are implemented, communicating concepts through three service delivery models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [6] as shown in Fig. 1.

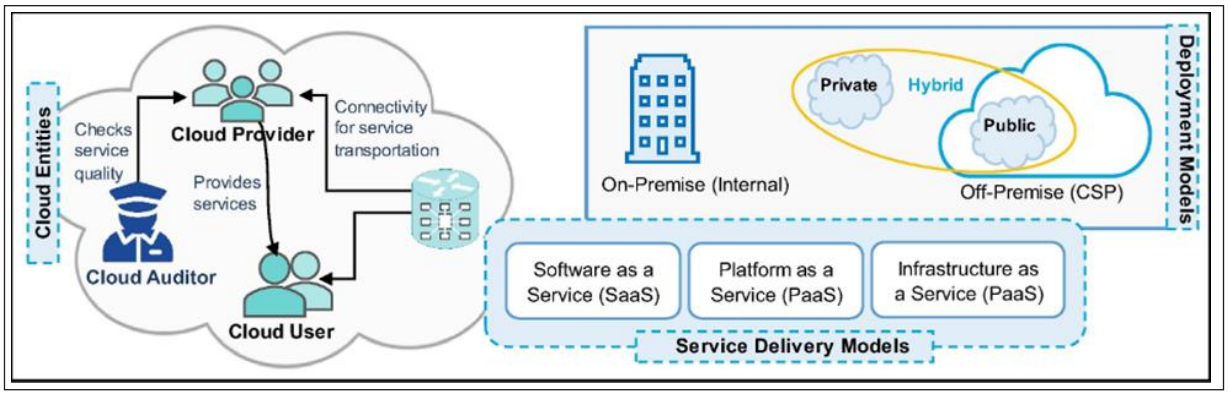

Fig. 1: A Cloud Computing Model [5]

It is important to have a reliable and consistent network connection to deliver services to clients, utilizing these cloud-based platforms effectively. Efficient and scalable characteristics of cloud computing may be attained by implementing effective cloud resource management [5], [7]. As a result, a lot of researchers have increased their focus on the load balancing of these resources in cloud computing. Load balancing refers to redistributing the overall workload among the individual nodes within a collective system which aims to optimize resource utilization and enhance job response time, while simultaneously addressing the issue of certain nodes being overloaded while others remain underutilized [5].

The problem of load balancing remains a significant challenge within cloud computing. Hence, the need for this research to provide a review of load-balancing techniques using different performance metrics. The purpose of this study is to conduct a comprehensive examination of load-balancing algorithms in the context of cloud computing. This study focuses on evaluating several load-balancing approaches utilizing multiple performance metrics providing an in-depth review of the latest advancements in load balancing and encouraging future investigation in this field. The rest of the research is organised as follows: Section II: discusses the concept of load balancing. In Section III: the various methods of load balancing, including static and dynamic load balancing algorithms, are discussed. Section IV: analyzed and showed a performance comparison of the algorithms based on certain specified performance metrics. Section V: provides the study's conclusion.

II. Concept of Load Balancing

Load balancing is a crucial strategy used to evenly spread the burden among different resources participating in computing operations inside a network, resulting in improved performance [8]. This workload includes memory utilization, CPU capacity, and network traffic. Within the domain of cloud computing, load balancing becomes a crucial issue, requiring the fair allocation of tasks among the available resources to achieve shorter response times [9]. This guarantees that the Virtual Machines (VMs) in the system are never overloaded at any one moment. Load balancing is seen as the coordination of a collection of servers engaged in the same service, executing identical tasks to enhance service quality by managing fluctuating customer loads over time [10].

According to E. J. Ghomi, A. M. Rahmani, and N. N. Qader [11] The fundamental goal of load balancing is to improve the availability, throughput, and dependability of a system, while also ensuring system stability, optimizing resource usage, and providing fault-tolerant capabilities. Careful incident handling is required since the chance of failures grows with the number of servers and high availability refers to the capability of maintaining uninterrupted service even in the presence of many faults occurring

simultaneously [11]. Load balancing aims to achieve optimal utilization of resources, which is a crucial objective for enhancing the efficiency of the cloud model. Attaining a high level of data processing capacity, which is crucial for a system that performs well, can only be accomplished when the tasks and available resources are fairly allocated throughout various nodes with the primary objectives encompassing attaining a minimal reaction time and averting bottlenecks [12], [13]. Within the domain of cloud computing, efficient load balancing enables data centers to effectively manage issues such as excessive or insufficient workload distribution among virtual machines as shown in Fig. 2. To keep up the performance of cloud applications an efficient Load Balancer must be provided an Unequal load distribution have many disadvantage one of them been Task Scheduling. Without proper task scheduling, the resources will not be efficiently utilized [9].

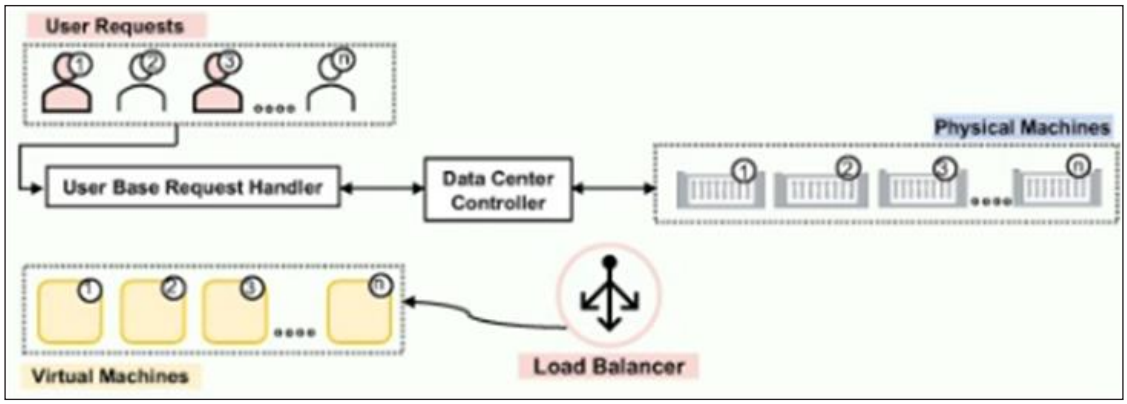

Fig. 2: Workflow in Cloud Computing [9]

The implementation of load-balancing techniques faces significant problems due to the influence of different physical and logical elements on the success of the strategy. Challenges arise from several factors, including the dispersion of graphical nodes, the migration of virtual machines, the complexity of algorithms, the presence of heterogeneous nodes, the risk of single points of failure, and the scalability of load balancers [11], [12], [13]. To address these issues, several load-balancing techniques have been proposed in the domain of cloud computing.

III. Classification of Load Balancing Techniques

The categorization of load balancing (LB) algorithms is contingent upon the operational state of the system, resulting in classifications as static, dynamic or hybrid. In essence, these classifications delineate whether the load-balancing approach employed is unchanging, adaptive to real-time fluctuations, or a combination of both static and dynamic elements [7] as shown in Fig. 3.

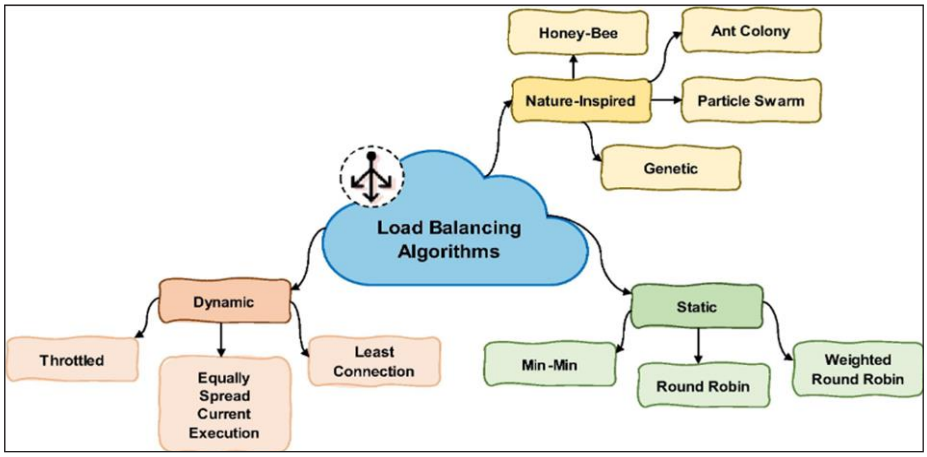

Fig. 3: Types of Load Balancing Algorithm

A. Static Algorithm

Static algorithms obtain information pertaining to the system and identify the existing resources for use before commencing thereby maintaining a consistent distribution of workloads regardless of changing conditions [4]. Static algorithms gather information about the system and identify available resources before starting. They evenly distribute workloads, regardless of changing conditions, using existing virtual machines (VMs) until a task is completed [10]. This works well when VM capabilities are similar. However, static algorithms face a challenge because they rely on fixed information and struggle to adapt to dynamic load changes in VMs effectively [14]. They are not flexible and cannot accommodate dynamic changes in attributes. These algorithms, also known as offline algorithms, need to know VM information in advance [4]. Despite their limitations, static algorithms generally achieve better overall performance than dynamic algorithms. It's important to note that when static load-balancing algorithms assign tasks to nodes, they don't consider the previous task's state and functionality of the node.

Some popular static load-balancing algorithms are presented in this section.

- *Round Robin Algorithm:* Round Robin, a static load balancing algorithm, operates without considering preceding states. Its simplicity lies in the utilization of the Round Robin Method for distributing tasks [10]. The algorithm randomly chooses the first node and then allocates jobs evenly to all other nodes using the Round Robin Method as seen in Fig. 4. An essential advantage of Round Robin is its independence from inter-process communication [15]. This algorithm provides faster response in the case of equal workload distribution among processes However, due to the lack of information about processors' running times, certain tasks may experience heavy loads.
- *Min-Min Load Balancing Algorithm:* This algorithm is user-friendly and operates quickly. It enhances performance by completing tasks in a series [16]. It calculates the time for each task and assigns it to virtual machines (VMs) based on the shortest completion time of existing tasks. This continues until all tasks are assigned to VMs. The algorithm performs well due to numerous smaller tasks, but there's a risk of neglecting larger tasks, leading to starvation as smaller tasks are prioritized over bigger ones [6].

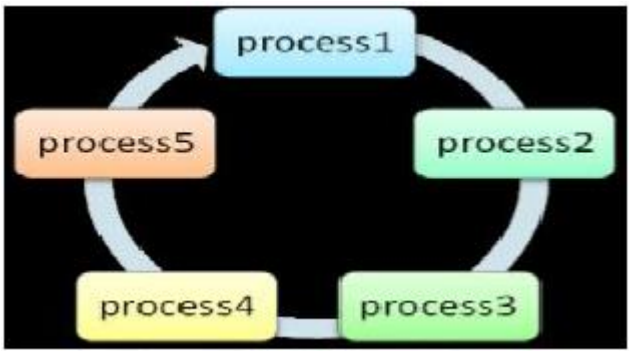

Fig. 4: Round Robin Load Balancing Algorithm

- *Max Min Load Balancing Algorithm:* This approach is similar to Min-Min load balancing, with a specific emphasis on the computation time of jobs [13]. Under this methodology, all tasks are given to the system, and the algorithm computes the duration required to complete each job. The machine is allocated the duty with the minimum time needed. The Max-Min method outperforms the Min-Min algorithm due to its ability to handle a larger job in addition to smaller tasks inside the set [9]. The Max-Min algorithm effectively manages jobs of varying durations, regardless of their length.
- *Opportunistic Load Balancing Algorithm:* The basic principles of this algorithm prevent it from comprehending the VM's activity level. Therefore, it ensures that every node in the system is actively working by randomly assigning those tasks [14]. Things go slowly, though, because it doesn't account for how long things take to complete. Consequently, it fails to provide reliable outcomes when attempting to distribute the labor evenly [14].

B. Dynamic Algorithm

Dynamic algorithms are a type of algorithm that differs from static algorithms because they exhibit

flexibility [9], [4]. These algorithms work based on rescheduling the tasks allocated to them over the available VM while carrying out the work. The main benefit is that task selection is influenced by real-time situations, leading to improved system performance [4]. Dynamic algorithms can be implemented in two ways:

- *Distributed System:* In this setup, all nodes in the system communicate with each other, and each node participates in executing a load-balancing algorithm. The responsibility of load balancing is shared among all nodes [17].
- *Non-Distributed System:* This can be centralized or semi-distributed. In a centralized system, a central node manages the load balancing for the entire system, and other nodes interact with this central node [18]. In a semi-distributed system, each cluster has a node that performs load balancing for the entire system. If the central node of a cluster fails, it affects only that cluster's functionality [18].

Some of the well-known dynamic algorithms are:

- *Throttled Load Balancer (TLB):* This algorithm generates a table containing virtual machines and their current states (available or busy). When a specific task is assigned to a virtual machine, a request is sent to the data center's control unit. This unit seeks the most suitable VM based on its capabilities for the given task [7]. The Table Allocator (TA) maintains a list, known as the index table or allocation table, containing all VMs and their corresponding states (available, busy, or idle). If an available VM with sufficient space is found, the task is allocated to that VM. If no suitable VM is available, TA returns −1, and the request is queued for prompt processing [5]. The load balancer's role is to locate a suitable virtual machine upon receiving a client request. While TA outperforms the Round Robin algorithm, it lacks consideration for more advanced load balancing requirements, such as processing time [5].
- *Ant Colony Optimization:* This ACO-based algorithm efficiently distributes workloads among cloud nodes. Drawing inspiration from ant behavior, it seeks alternative pathways when encountering obstacles, establishing new routes between nodes [18]. The algorithm begins by initializing a table, facilitating data flow, and reaching the required nodes' threshold level. As the flow passes through nodes, the algorithm evaluates each one. If a node is underloaded, it utilizes the highest Trailing pheromone (TP) to find the route for the underloaded node [18]. The table is updated until the node reaches its threshold limit. However, upon reaching the limit, the algorithm uses a Foraging Pheromone (FP) to explore new sources, updating the table until it finds an underloaded node, at which point it reallocates resources. This process repeats until the entire process is completed.
- *Honey Bee Algorithm:* This algorithm operates based on the behavior of bees searching for honey. Its primary goal is to distribute the workload on virtual machines (VMs) efficiently, avoiding both excessive resource use and underutilization [19]. The algorithm selects a VM that meets two key criteria: it receives fewer tasks compared to other VMs, and its processing time falls within the average processing time of all VMs [19].
- *Load Balancing Technique Based on Cuckoo Search and Firefly:* The cuckoo search algorithm draws inspiration from the nesting behavior of cuckoo birds. Similar to how these birds lay their eggs in the nests of other birds, the algorithm carries forward the best-quality eggs to the next generation and abandons the worst nests [20]. In the context of cloud computing, the Cuckoo Search with Firefly, a hybrid algorithm, is employed for load balancing. This algorithm efficiently schedules tasks, assigning overloaded tasks to under-loaded virtual machines (VMs). It quickly identifies the best VM, enhancing load balance efficiency and preventing task imbalances across the system [21]. The load balancing (LB) technique initiates by scheduling tasks using the round-robin method. It then calculates the capacity and load of each VM, identifying overloaded and under-loaded tasks through Cuckoo Search

with Firefly. This method ensures task migration from overloaded VMs to under-loaded VMs, preventing overall system task imbalances [21]. Notably, the approach boasts a high interchange rate and requires fewer tuning parameters compared to existing methods.

- *First Come First Serve Algorithm:* This algorithm operates by assigning new tasks to resources with the shortest waiting time, meaning those resources handle the fewest tasks. It follows a sequential execution approach, starting with the first task and progressing through the queue until completion [16]. The algorithm is chosen because it allocates tasks to virtual machines without taking into account the specifications of available virtual machines or the time required for tasks in the queue when assigning new tasks [22]. To determine the most effective load-balancing algorithm for optimal performance in cloud applications, various popular metrics are employed to measure their effectiveness.

IV. Performance Comparison of Load Balancing Algorithm

This section presents the key metrics commonly examined by authors when assessing the latest advancements in load-balancing algorithms. These metrics play a crucial role in shaping and creating effective load-balancing algorithms. They serve as indicators of an algorithm's performance in cloud applications. Researchers [16], [23], [9], [13], [5] utilize certain parameters to assess their proposed algorithms, emphasizing the need for adjustments to prevent imbalances in the cloud computing environment. From the review, the popular performance metrics that will be used in this research are:

- *Resource Utilization (RU):* The utilization of system resources such as memory and CPU is a measure of Resource Utilization (RU) within the cloud Data Center. RU gauges the extent to which resources are employed. As service demand grows, maintaining a high RU becomes crucial. A maximum RU is necessary to ensure optimal performance for the load-balancing algorithm.
- *Complexity:* An algorithm should exhibit optimal performance even when faced with unforeseen circumstances. This implies that irrespective of an upsurge in tasks and load, the algorithm should maintain scalability, ensuring consistently high performance for the load-balancing algorithm.
- *Throughput (TP):* It denotes the count of job requests completed and successfully processed per unit of time within the virtual machine. It signifies the volume of data moving from one location to another. High throughput is indicative of effective load-balancing algorithm performance.
- *Response Time (RT):* The duration the algorithm requires to address a task, encompassing waiting, transmission, and service times, is called response time. It signifies the time necessary to answer a user's inquiry, and an effective load-balancing algorithm aims for the minimum response time.
- *Speed:* The total time needed to finish all tasks and distribute resources to users within the system is known as the total completion time. This metric is critical in the scheduling process within the cloud environment, providing a measure of the time required to process a specific set of tasks. A good load-balancing algorithm aims to minimise this metric, ensuring efficient task completion.
- *Network Overhead (NO):* It is the level of additional work generated during the execution of the load balancing algorithm, which may arise from a significant number of task migrations and inter-process communication. An evenly distributed system load leads to minimal overhead caused by the load-balancing algorithm.
- *Fault Tolerance (FT):* A robust load-balancing system should function effectively even in the event of failures in some components. This implies that if one virtual machine (VM) is overloaded, another available VM should be

capable of carrying out tasks. Ensuring high fault tolerance is crucial for the optimal performance of the load-balancing algorithm.

- *Migration Time (MT):* This represents the overall time required to transfer a task from one virtual machine (VM) to another, and this migration process should occur without impacting the system's availability. The efficiency of this process is closely tied to the virtualization concept in the cloud. A low migration time is crucial for optimal performance in a load-balancing algorithm. Table I represents the performance of the Load Balancing Algorithms reviewed in this research.

Table I: Performance of the Load Balancing Algorithms

Performance Metrics	Load Balancing Algorithms								
	Round robin	MIN-MIN	MIN-MAX	Opportunistic	Cuckoo Search and Firefly	Throttled Algorithm	Ant Colony	Honey Bee	First Come First Serve
Throughput	Low	Moderate	High	Low	High	High	Low	Low	Moderate
Complexity	Low	Low	Moderate	Low	Moderate	Moderate	Moderate	High	Moderate
Response time	Low	Low	Low	Moderate	Moderate	High	High	High	High
Speed	Low	Low	High	Low	High	High	Low	Moderate	Moderate
Network overhead	Low	High	Moderate	Low	Moderate	Low	High	High	Low
Fault Tolerance	Low	Low	High	Low	High	High	High	Moderate	Low
Migration time	Low	Moderate	Moderate	Low	High	High	Low	Low	Low
Resource utilization	High	Moderate	High	Low	High	High	Moderate	High	High

V. Conclusion

Load balancing holds significance in cloud computing as it improves the distribution of workloads and optimises resource utilisation, consequently reducing the overall system response time. This research delves into the categorization of load-balancing algorithms, providing an overview of existing algorithms designed to balance loads in cloud computing and their performance. These performance metrics are crucial factors for designing a robust algorithm capable of thriving in a dynamic cloud environment. The Max-Min algorithm came in second place among the algorithms this study examined, after Cuckoo Search and Firefly which is the best algorithm. This research contributes by aiding researchers in identifying load balancing-related research challenges, particularly in further minimizing response times and preventing server failures, and by offering a summary of existing techniques. Future researchers can build upon this foundation by exploring the development of more dynamic, intelligent algorithms addressing fault tolerance issues to further elevate the efficiency of cloud computing.

References

[1] R. Boutaba, Q. Zhang, and M. F. Zhani, "Virtual machine migration in cloud computing environments," in *Advances in Systems Analysis, Software Engineering, and High-Performance Computing Book Series*, 2014, pp. 383-408, doi: https://doi.org/10.4018/978-1-4666-4522-6.ch017.

[2] U. B. Usman, F. U. Ambursa, E. J. Onyiyechi, and T. J. Jaremiah, "Virtual machine migration techniques in cloud computing environment: A review of load balancing algorithms," *International Journal of Artificial &*

Computational Intelligence, vol. 1, no. 3, Sep. 2020.

[3] S. Kaur, and V. Pandey, "A survey of virtual machine migration techniques in cloud computing," *International Institute for Science, Technology, and Education (IISTE) Computer Engineering and Intelligent Systems,* vol. 6, no. 7, pp. 28-34, 2015, ISSN: 2222-1719 (Paper), ISSN: 2222-2863 (Online).

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

[5] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 7, pp. 3910-3933, Jul. 2022, doi: https://doi.org/10.1016/j.jksuci.2021.02.007.

[6] P. Kumar, and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing," *ACM Computing Surveys*, vol. 51, no. 6, pp. 1-35, Feb. 2019, doi: https://doi.org/10.1145/3281010.

[7] S. Hamadah, "A survey: A comprehensive study of static, dynamic and hybrid load balancing algorithms," *International Journal of Computer Science and Information Technology & Security (IJCSITS)*, vol. 7, no. 2, pp. 27-26, Mar.-Apr. 2017, ISSN: 2249-9555.

[8] N. S. A. Chawla, "Efficient cost scheduling algorithm with load balancing in a cloud computing environment," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 3, no. 6, pp. 5138-5143, Jun. 2015, doi: https://doi.org/10.15680/ijircce.2015.0306026.

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

[10] R. S. Sajjan, and B. R. Yashwantrao, "Load balancing and its algorithms in cloud computing: A survey," *International Journal of Computer Sciences and Engineering,* vol. 5, no. 1, pp. 95-100, 2017.

[11] E. J. Ghomi, A. M. Rahmani, and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," *Journal of Network and Computer Applications*, vol. 88, pp. 50-71, Jun. 2017, doi: https://doi.org/10.1016/j.jnca.2017.04.007.

[12] S. Talwani, and J. Singla, "A comprehensive review of virtual machine migration techniques in cloud computing," *Social Science Research Network*, Jan. 2020, doi: https://doi.org/10.2139/ssrn.3564971.

[13] A. A. Alkhatib, A. Alsabbagh, R. Maraqa, and S. AlZu'bi, "Load balancing techniques in cloud computing: Extensive review," *Advances in Science, Technology and Engineering Systems Journal*, vol. 6, no. 2, pp. 860-870, Apr. 2021, doi: https://doi.org/10.25046/aj060299.

[14] W. Duan, X. Tang, J. Zhou, W. Jing, and G. Zhou, "Load balancing opportunistic routing for cognitive radio ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1-16, Dec. 2018, doi: https://doi.org/10.1155/2018/9412782.

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

[16] A. Thakur, and M. S. Goraya, "A taxonomic survey on load balancing in cloud," *Journal of Network and Computer Applications*, vol. 98, pp. 43-57, Nov. 2017, doi: https://doi.org/10.1016/j.jnca.2017.08.020.

[17] R. Panwar, and B. Mallick, "Load balancing in cloud computing using dynamic load management algorithm," *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, Oct. 2015, doi: https://doi.org/10.1109/icgciot.2015.7380567.

[18] N. R. Tadapaneni, "A survey of various load balancing algorithms in cloud computing," *Int.*

J. Sci. Advance Res. Technol., vol. 6, no. 4, pp. 484-487, 2020.

[19] W. Hashem, H. Nashaat, and R. Rizk, "Honey bee-based load balancing in cloud computing," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 12, pp. 5694-5711, Dec. 2017, doi: https://doi.org/10.3837/tiis.2017.12.001.

[20] M. Yakhchi, S. M. Ghafari, S. Yakhchi, M. Fazeli, and A. Patooghi, "Proposing a load balancing method based on Cuckoo Optimization Algorithm for energy management in cloud computing infrastructures," *International Conference on Modeling, Simulation, and Applied Optimization*, May 2015, doi: https://doi.org/10.1109/icmsao.2015.7152209.

[21] K. K. Kumar, T. Ragunathan, D. Vasumathi, and P. V. S. R. B. Prasad, "An efficient load balancing technique based on Cuckoo search and Firefly algorithm in cloud," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 3, pp. 422-432, Jun. 2020, doi: https://doi.org/10.22266/ijies2020.0630.38.

[22] P. Prajapati, and A. K. Sariya, "A review: Methods of load balancing on cloud computing," *Int. J. Res. Anal. Rev.*, vol. 6, 2019, doi: https://doi.org/10.6084/m9.doi.one.IJRAR19J2346.

[23] T. Tamilvizhi, and B. Parvathavarthini, "A novel method for adaptive fault tolerance during load balancing in cloud computing," *Cluster Comput.,* vol. 22, no. S5, pp. 10425-10438, Sep. 2019.