A Survey on Load Balancing Algorithms in Cloud Computing Systems

Suchandan Ganguly
Dept. of CSE, Swami Vivekananda University Kolkata, India
gangulysuchandan8@gmail.com

*Abhijit Paul
Dept. of Computer Science Swami Vivekananda University Kolkata, India
a paul84@rediffmail.com

Ranjan Kumar Mondal
Dept. of CSE
Swami Vivekananda University Kolkata, India
ranjankm@svu.ac.in

*Corresponding author Email ID: a_paul84@rediffmail.com

***Abstract*—Cloud computing is one of today's fastest-growing technologies. People use it to handle, tweak, and pull data from all sorts of distributed apps—right over the internet. The perks are hard to ignore, so adoption keeps surging day by day. That boom, though, brings a real headache: optimizing resource use. Enter load balancing. Researchers have rolled out algorithms galore in papers to tackle it. This paper describes a survey of different load balancing algorithms in cloud computing, along with their objectives, parameters used, optimization algorithm used, implementation and future scope. A comparative study is also depicted in Table I.**

***Index Terms*—Cloud Computing, Load Balancing, AWS, Genetic Algorithm, HBFQL, PSO.**

I. INTRODUCTION

John McCarthy, a pioneering computer scientist, first floated the idea of cloud computing back in 1961. Fast forward to 1999: Marc Benioff and Parker Harris, along with their team, launched Salesforce.com—a trailblazing cloud-based CRM platform. The early 2000s saw Microsoft leaning into SaaS, delivering software apps over the cloud to users everywhere.

Then came 2002. Amazon rolled out AWS, a full-fledged cloud platform for computation and storage. By 2007, heavyweights like Google and IBM, plus various universities, ramped up broad research into cloud computing.
At its core, cloud computing makes resources available as-a-service. Users tap into storage and power via the internet. Pay only for what you use—no more, no less. We can store files, images, documents, etc. The following are some of the characteristics of cloud computing [2]-

- On-demand self-service- It means that the user can easily use the computing resources as and when required without the help of any human beings.
- Scalability- This can be achieved by increasing or de- creasing the number of resources in the cloud.
-

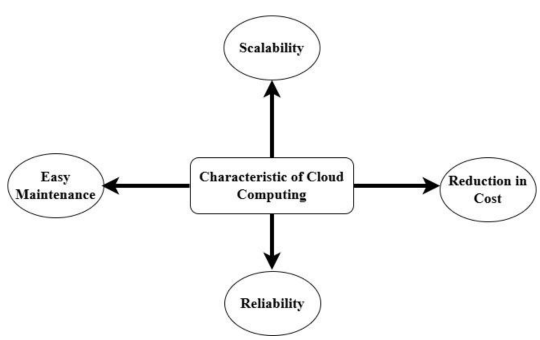

Fig. 1. Characteristic of Cloud Computing

- Reduction in Cost- As in cloud computing, users have to pay only that portion of whatever they are using.
- Resource pooling- It is a process that combines computing resources such as hardware, software, bandwidth. It is then assigned to the users for access.
- Reliability- In spite of relying on one single instance of a physical server, the virtual partition is considered here. So, if one server goes down, then virtual servers will take it up.
- Easy Maintenance- Maintaining any server is easy. If any server goes down it can be rectified within a limited time span.
- Rapid elasticity- It refers to the cloud's capability to ascend accordingly which will fulfill the demand.
- Measured service- It is the capability to command and control computing resource as per the usage.

The importance of this work is to provides information for the new researchers who want to continue work on designing new load balancing algorithms. The goals of this paper are mentioned below:-

- A study of the existing load balancing algorithms.
- Providing an idea of different optimization algorithm and simulation techniques used.
- Highlighting the results of the existing reviewed papers.
- Providing information about the ideas used in the existing reviewed papers.

The remaining part of the paper is organized as follows. Section II provides information about load balancing techniques. A literature review is given in Section III. Section IV provides a brief description of the discussion of results of the literature. Finally, in Section V, we present the conclusion of our survey.

II. LOAD BALANCING

A. LOAD BALANCING TECHNIQUES

Load balancing is a technique to distribute the resources among VMs in evenly manner [3]. It is used to allocate the load equally and to use the resources very well so that no VMs are overloaded or underloaded. Using this property, the time taken to send and receive data can be decreased. It also increases throughput and decreases response time.

Load balancing algorithms, can be divided into three categories: static, dynamic, and nature inspired.

- Static Load Balancing Algorithm- These types of al- gorithms do not take into consideration the previous state of a node. Examples of some of these types of algorithms are Max-Min, Min-Min, OLB and Round Robin etc.
- Dynamic Load Balancing Algorithm- These types of algorithms first checks the previous state of a node during the distribution of load. Examples of some of these types of algorithms are Active clustering, Least Connection, Equally Spread Current Execution etc.
- Nature inspired Algorithm- These types of algorithms are based on biological processes or activities based on human nature. Examples of some of these types of algorithms are Ant Colony Optimization, Particle Swarm Optimization, Genetic Algorithm etc.

Static algorithms are again can be classified into two groups. They are optimal and sub-optimal [4].

- In the case of an optimal controller, the data center decides information about the tasks and resources. The load balancer can only do a maximum allocation in a descent time.
- If, for any reason, the load balancer is unable to calculate an optimal decision, a sub-optimal allocation is used for calculation.

Dynamic algorithms are classified into two types one is distributed and another is non-distributed [4].

- In the case of a distributed mechanism, all the nodes perform the dynamic load balancing algorithm. The load balancing job was shared between the nodes. Non-distributed mechanism allocate request into the resources inside a centralized system

Non-distributed algorithms are further classified into two categories one is centralized and another is semi distributed.

- In the centralized category, the central node is only responsible for the execution of the load balancing algorithm. The remaining nodes only communicate with the central node.
- In semi distributed category, the existing nodes are divided into clusters and each cluster is centralized by itself.

B. LOAD BALANCING METRICS

The following are some of the metrics used in load balnacing algorithms.

- Makespan- It is the amount of time to process all the tasks.
- Response Time- It is the time taken by the server to operate with the incoming requests and send a response.
- Throughput- It is the amount of data delivered from one point to another within a certain time. It is generally denoted in Megabits per second (Mbps) or Gigabits per second (Gbps).
- Energy consumption- It is the energy which is required to run and allot the loads in between the resources.
- Transfer time-The time taken to move the resources for execution from one node to another [5].
- CPU utilization- The amount of load that can be handled by the core of the CPU's of the resources.
- Memory usage- The amount of primary memory used by the loads in the resources.
- Task Prioritization- This is mainly the sequence of task to be executed or processed by the virtual machines.
- Bandwidth- the speed at which information is moved between nodes.
- Execution Time- The CPU's execution time for a request in a node.
- Completion Time- how long it takes the CPU of a node to finish a request.
- Utilization Degree- The measure to which a resource is used in context to its maximum capacity.

III. LITERATURE SURVEY

The various concepts and approaches that numerous writers have put forth for efficient load balancing in cloud computing are described here.

[6] focuses on developing a novel load balancing method that uses learning automata. They also focus on

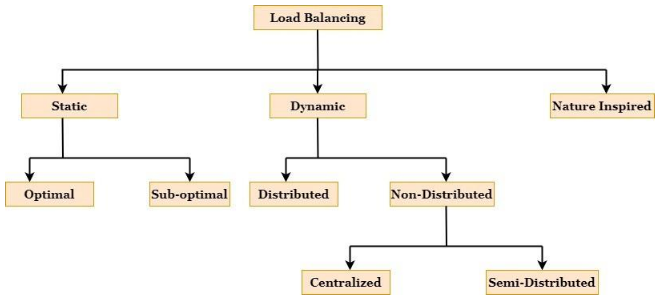

Fig. 2. Types of Load Balancing Algorithm

dynamically dividing real time requests to edge and cloud servers. Here the system learned from the previous experience and accordingly changes the workloads and network condition. Maximum work allocation was ensured by this procedure. A Service Time Measurement standard was added to the newly suggested system. It is employed to determine the performance of servers. Decisions about the distribution of tasks are also made using it. Reinforcement learning has been used in this instance to reduce the average time. In this paper the S environment model has been used. Learning automata was implemented using the current server and available server. The current server represents to the server currently processing a task, and the available server denotes the list of probable servers which will be used for processing the tasks. In future integrating learning automata can be incorporate with energy aware and deep learning methods which will increase the adaptability of the algorithm. Fault models for mobile devices and edge servers also can be introduced which will better robustness and scalability.

The objective of the paper of [7] is to build a metaheuristic load balancing technique suitable for cloud environments of dynamic and heterogeneous jobs and resources. They also emphasizes on to decrease the energy usage and makespan which will be basically a multi objective technique for users and service providers. The capability of the new technique has been verified using various tasks and resources. Here Rock Hyrax Algorithm is used as optimization algorithm which focuses on the calculation of upper bound on the required load of a server. There are two phases: one is to calculate server capacity and the other is to balance the load by the Rock Hyrax algorithm. In future to better the prediction of degree of accuracy of workload machine learning tools will be used. It will also propose a technique for dynamically adjustment to new conditions. Also in future it will optimize task using heterogeneous resource aware technique and also develop a technique that will protect the privacy.

The goal of [8] is to improve performance, speed, and throughput while also accelerating task execution. Additionally, it focuses on reducing runtime and makespan. The Q-learning machine learning algorithm and the honey bee foraging algorithm serve as the foundation for this hybrid algorithm, which essentially combines dynamic and reinforcement learning algorithms. It can be used in a variety of settings. This study uses the Q-learning machine learning method to determine the Q-value of the virtual machines (VMs) and the Honey Bee algorithm to determine the VMs' state.

The focus of [9] is on the proper distribution of stress and optimizing task scheduling. It also emphasizes on building a load balancing algorithm which considers different QoS parameters which conflict often. Another objective of this paper is to increase the working functions of VMs in different resources and workloads situation and to improve the clustering mechanism. Here an advance deep learning mechanism is used by using CNNs and RNNs together. A clustering technique is used which uses RL and advanced hybrid optimization technique. A Multi-Objective Hybrid Optimization model is used for scheduling the tasks. In the future, a modifiable technique will be developed to dynamically adjust the parameters of deep learning, reinforcement learning, and hybrid optimization. Also a self-adjusting technique will be developed that will refine self performance depending on workload properties.

The objective of [10] is to automate task scheduling and optimize the use of VMs that are in the best positions. It also aims to develop a robust decision-making algorithm with a feedback controller that reduces VM migration while increasing fault tolerance. Here, a new Intelligent Weighted filtering-based PSO technique is employed to decrease computation time for task scheduling and resource allocation. It utilizes a multi-objective PSO algorithm with Pareto dominance, which enhances service quality, throughput, scalability, and reduces response time. Additionally, a Double Deep Q proximal model with a feedback controller has been introduced to prevent violations of the service-level agreement. To avoid single-point failure, a conditional GAN feedback controller has been introduced. The Siamese Hamiltonian Markov Neural Network determines the status of the VMs, which is overloaded, underloaded, or balanced. The focus of [11] is to obtain fast converge and global optimization. It also emphasizes improving system efficiency and resource allocation. Here GWO is used to find out the best location called alpha. PSO uses this alpha value. This is used to develop an objective function that will be helpful in balancing the load in the cloud system. In future quality parameters will be calculated along with new and modified GWO has been developed. A new version of GWO and PSO will be combined together that will give better outcome.

The focus of the article by [12] is to build a useful load balancing and task scheduling technique that will increase throughput, reliability, and resource usage. In this paper the concept of hunting technique of dingo has been used which resembles with tasks and VMs resembles as prey. To find out the optimal allocation of incoming tasks DOA technique is used. It uses the advantages of whale optimization for betterment of the exploitation phase of DOA. It uses the fitness function to implement load balancing. In future an African Vulture Optimization-based load balancing approach will be established.

Building a resource handling method based on the RBAC algorithm is the aim of the paper [13]. Three layers make up the new method: the edge layer, the fog layer, and the IOT sensor layer. The IoT sensor layer is made up of a variety of sensors that are employed for various metrics. The fog layer handles load balancing, whereas the edge layer does on-device computations.

Building an efficient load balancing and computation offloading (LBCO) strategy is the goal of [14]. It also focuses on creating a robust security layer and optimizing energy and time demands. For security, an advanced encryption standard cryptographic technology is used along with an encryption and decryption key based on an ECG signal. The model proposed works well for multiuser, multitask, multitier, mobile-edge cloud computing systems. Looking ahead, deep learning techniques could be employed to model, formulate the problem, and find solutions. Additionally, a method will be developed to allow Mobile Device Users (MDU) to seamlessly move between different BSs. Depending on user behavior, an AI-based system could be designed to automate security-related processes.

The objective of the paper of [15] is to propose a new task scheduling approach, which will better the makespan and load balancing. Balancer Genetic Algorithm (BGA) is proposed as an optimization approach which is used for task scheduling. It takes a number of tasks in a batch and then allocates them to VMs. Here, two different objectives have been chosen, which will better coordinate the task. In future Service Level Agreements can be added in which tasks have unequal priorities.

The focus of [16] is to develop an energy framework which can access the information from IoE cloud network and to form group of same type of sensors in the IoE framework. The new technique named EECloudIoE merges the Internet of Everything and Cloud which provides real-time services. In future a mechanism for routing the packets will be developed which could minimize the throughput and request response time.

By distributing the load among virtual machines (VMs) and increasing their throughput, the article by [17] aims to improve the machine's performance. It also focuses on maintaining the balance between task priorities by maximizing task waiting times. Three goal functions are proposed in this case: the first measures the disparity between each host's load and the cloud's average load; the second is based on the total energy consumption; and the third considers the number of jobs sent to different cloud processing units. The final fitness function is calculated using the weighted sum of each individual fitness function. Here, modified particle swarm optimization (MPSO) and an improved Q-learning algorithm are combined to create a new method called QMPSO. In the future, a dynamic load balancing approach will be applied to interrelated tasks.

The goal of [18] is to provide a method for distributing the workload evenly across virtual machines based on their computational capacities. One interesting approach to scheduling is the resource-aware load balancing algorithm (RALBA). In this method, a batch dynamic scheduling heuristic is used, which works in two simple steps: first, workload is shared out once the virtual machines' (VMs') capabilities are understood. Then, in the second step, any remaining tasks are assigned to VMs to achieve the Earliest Finish Time (EFT). This method is designed to handle autonomous, non-preemptive jobs effectively. There are two main types of schedulers: Fill and Spill. Moving forward, future work will aim to enhance RALBA's features to better accommodate slower computing resources and tackle resource challenges with a fault-tolerant scheduling approach.

The article by [19] aims to give planned jobs the quickest completion times while also enhancing system performance and service quality. The suggested method is divided into two stages: load balancing and work scheduling. Here, a new algorithm known as Random Make Genetic Optimizer (RMGO) is created by combining three methods. For load balancing, GA is utilized. The goal is to increase data storage availability, power consumption, green computing, resource selection, and data replication in the future. This will improve system availability and response time.

The focus of [20] is to build a combination of resource scheduling and load balancing algorithm and also increasing the utilization of VMs and improving latency time and decreasing processing time. Here, we put together a fuzzy-based model to handle multidimensional resource scheduling—aiming straight for better efficiency. It pairs with a load balancing algorithm that wards off underuse and overloads. Looking ahead, the plan calls for a privacy-aware version, folding in preservation alongside the other key factors.

The objective of the paper of [21] is to build a technique which will consider the two parameters-server processing power and computer loading that will able to deal with excessive computational requirements. The proposed technique can be used in both virtual web servers and physical servers. Cloud load balancing technique is used which is consists of 5 layers.

The focus of the article by [22] is to build an energy- cost aware load balancing technique and to implement a technique that will enhance the throughput. Here Krill Load balancing technique is used to enhance the throughput. A modified dynamic energy-aware based on cloudlet mobile cloud computing model (MDECM) was implemented. The Krill herd load balancing is dynamic in nature.

The focus of [23] is to achieve efficient resource utilization and to decrease the makespan. Genetic Algorithm (GA) was used here as a soft computing technique. It is based on the natural selection mechanism. It can be used for large search spaces and difficult objective functions. In the future, different types of crossover and selection approaches can be explored.

Maximizing the optimization of available resources is the main goal of [24]. In this case, the load balancing mechanism is solely carried out by the central node. One solution candidate is assigned to a group of potential successors using a candidate generator. Valid solutions are categorized using an evaluation criterion. Different soft computing techniques can be employed in the future to achieve even greater results.

The focus of [25] is to build an independent agent-based load balancing Algorithm (A2LB).It supports dynamic load balancing. This approach ramps up resource utilization and throughput while keeping response times in check. A2LB breaks it down with three key agents: the Load Agent, Channel Agent, and Migration Agent. The Load and Channel Agents stay put—they're static. But the Migration Agent? It's like an ant on the move, a mobile agent designed to scout the best paths, just like ants do in nature.

The Load Agent handles info policies and tracks every detail about a data center. It logs VM specs in a handy fitness table. Meanwhile, the Channel Agent oversees transfers, selections, and location rules. When it's go-time, the Channel Agent fires off the Migration Agent. That agent's real work kicks in as it hops to other data centers, chats with their Load Agents, and checks VM status across the board.

IV. DISCUSSIONS

According to the study [6], the suggested algorithm outperforms other current methods in dynamic edge-cloud systems with regard to energy efficiency, adaptability, scalability, and task-handling strategies. According to the article [7], the suggested method has reduced the makespan by 10% to 15% and the overall energy consumption at data centers by 8% to 13%. The ant colony optimization approach and the shortest job first have been used to compare performance in the paper [8]. The task was distributed swiftly, as can be observed. The suggested method in the article [9] can manage better resource use, which also raises CPU consumption and reduces makespan. Compared to other current models like PSO, TOPSIS-PSO, and C-MODLB, the suggested model in the article [10] has a lower transmission time of 1.7 s, a shorter makespan of 22 s, a high resource utilization of 78%, a low load fairness of 1.00%, and a higher number of migrations of 23%. According to the article [11], the overall response time has been reduced to 12%. It has been also observed that the proposed GWO- PSO algorithm better PSO by 97.253%. According to the publication [12], the computational cost is reduced by 16.54%, 18.92%, and 21.26%, while the computational complexity is reduced by 15.16% and 18.92%. According to [13], as compared to alternative methods, an average improvement of 29.85%, 16.82%, 13.63%, and 16.55% was achieved in terms of CPU utilization, memory usage, delay, and jitter, respectively. In the paper [14] it has been proved that about 68.2% of system consumption was saved with additional security layers compared to without

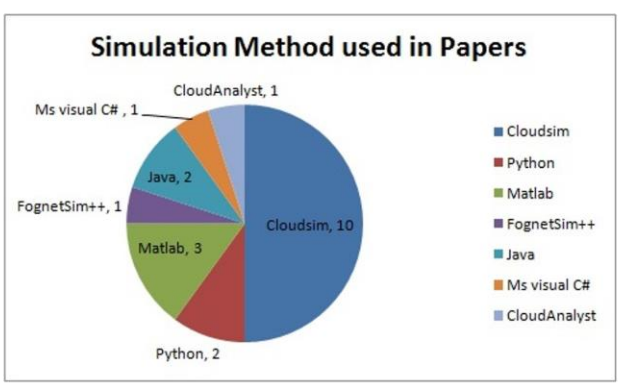

Fig. 3. Simulation Methods used in Papers

the security layer which is about 72.4%. In [15], it has been claimed that the proposed technique improves makespan by 27.3%, 71.9%, 40.5%, and 4.6%. The new technique in the article [16] increases lifetime and reduces traffic. In [17], a survey has been conducted; considering the number of tasks migrated, response time of tasks, delays in all tasks, idle time of tasks, and makespan before and after load balancing through modified PSO and improved Q-learning. It performs better than other algorithms. The proposed algorithm in [18] has achieved better resource utilization, makespan, and throughput. It also improves resource utilization by 7.21% compared to the Random Selection (RS) heuristic. According to [19], the suggested approach reduces makespan when compared to other job scheduling methods currently in use. The new approach improves resource scheduling efficiency by 7% and decreases response time by 35.5% in [20]. Additionally, the strategy proposed in [20] improves server performance. The method in [22] boosts throughput while lowering average turnaround time, energy usage, and latency. The new load balancing strategy in [23] meets client tasks' QoS requirements and performs better than some current approaches. The proposed method outperforms the Round Robin and First Come First Serve (FCFS) algorithms in [24]. Results in [25] show that the suggested algorithm shortens service time.

TABLE I
STATE-OF-ART COMPARISON

Author Name	Parameters Considered	Optimization Algorithm Used	Simulation	**Future Scope Mentioned**
Mondal et.al, 2012	Response time (RT)	Stochastic Hill Climbing which is a type of Hill Climbing algorithm	CloudAnalyst	Yes
Dasgupta et.al, 2013	Response time, Makespan	Optimization is done through GA	CloudSim based Visual Modeler	Yes
Singh et.al, 2015	Memory, CPU utilization, Response time	No optimization algorithm used concept based on agent based	Java	No
Chen et.al, 2016	CPU, RAM	Weighted Round Robin	The Programming Language used is MS Visual Studio 2010 C# and MS SQL Server 2005 database	No
Hasan & Mohammed, 2017	Speed, Task Cost and Weight	Krill herd optimization algorithm	JAVA platform	No
Priya et.al, 2018	Bandwidth, CPU utilization and Memory	Multidimensional Queuing Load Optimization algorithm	CloudSim	Yes
Mohamad et.al, 2018	Numbers of VMs, Cloudlets, hosts and Datacenters	Combination of Min-min, Max-min and Suffrage	CloudSim	Yes
Hussain et.al, 2018	Makespan, Throughput, and Resource utilization	Spill Scheduler	CloudSim	Yes
Jena et.al, 2020	Disk, memory, CPU and Bandwidth	Modified Particle Swarm Optimization (MPSO) algorithm	CloudSim3.0.3	No
RM et.al, 2020	Temperature, energy, Load, and Alive nodes	Wind Driven Optimization algorithm to create cluster. Firefly optimization algorithm to select optimal cluster heads.	Matlab R2015a version	Yes
Gulbaz et.al, 2021	Makespan, Average resource utilization ratio	Balancer Genetic Algorithm	CloudSim	Yes

Zhang et.al, 2021	Location, CPU cycle count, uplink data rate, and current user count	A secure LBCO algorithm	MATLAB	Yes
Kumar and Agrawal, 2022	CPU usage, Memory usage, Delay, Jitters	Optimal Scheduling Policy (SP)	FogNetSim++	No
Ramya, K.,& Ayothi, S., 2023	Response time, Completion time, Makespan, Utilization degree	Combination of Whale and Dingo optimization technique called as Hybrid dingo and whale optimization (HDWOA)	CloudSim	Yes
Reshan et.al, 2023	Response time, Throughput, Execution time	Combination of GWO and PSO	Matlab	Yes
Ghafir et.al, 2024	Workload departure, Demand capacity, CPU, memory, Bandwidth	Multi-objective Particle Swarm Optimization (PSO) algorithm with Pareto dominance	Python	Yes
Ahmad Raza Khan, 2024	Makespan, Energy consumption, CPU utilization, Memory usage and Task prioritization	Hybrid Lyrebird Falcon Optimization Algorithm	Python and Cloudsim	Yes
Adewale et.al, 2024	Algorithm runtime, Throughput and Makespan.	Honey bee foraging Q-learning (HBFQL) algorithm	CloudSim simulator with an Intel core i7 processor, 16 GB RAM, 2.8 GHz CPU and Windows 11	No
Singhal et.al, 2024	Makespan, Response time, Throughput and Energy efficiency.	Rock Hyrax algorithm based on Swarm optimization algorithm	CloudSim version 3.0.3 on Windows 7	Yes
Ghorbani et.al, 2025	Average service time, delay, network traffic, and number of successful task execution	No such optimization algorithm has been used	EdgeCloudSim, a specialized tool based on CloudSim	Yes

The figure-3 demonstrates that how different simulators are used in the literature review papers. Here different simulators like CloudSim, Python, Matlab, Java etc are used. From the figure-3 it can be concluded that CloudSim is used frequently as simulator than the other.

V. CONCLUSION

The use of cloud computing for load balancing plays a vital role. It provides a better way by which resource utilization and task scheduling can be done in a better way. Different authors previously proposed different types of load balancing algorithms till date. After thoroughly studying some of the articles it has been concluded that dynamic load balancing algorithms are much better than static one. The main emphasis is on to choose a good optimization technique. Basically, more often combination of more than one optimization technique has been used which performs better compared to others. The nature inspired algorithms when combined with each other, perform better way which has been already proved by the articles of many authors.

In future, other observations can be made from the latest articles to find out new types of optimization techniques which may outperform the existing ones.

REFERENCES

[1] Giordanelli, R., & Mastroianni, C. (2010). The cloud computing paradigm: Characteristics, opportunities and research issues. Istituto di

Calcolo e Reti ad Alte Prestazioni (ICAR), 5, 11.
[2] Mahmood, Z. (2011, August). Cloud computing: Characteristics and deployment approaches. In 2011 IEEE 11th International Conference on Computer and Information Technology (pp. 121-126). IEEE.
[3] Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2022). Load balancing techniques in cloud computing environment: A review. Journal of King Saud University-Computer and Information Sciences, 34(7), 3910-3933.
[4] Ghomi, E. J., Rahmani, A. M., & Qader, N. N. (2017). Load-balancing algorithms in cloud computing: A survey. Journal of Network and Computer Applications, 88, 50-71.
[5] Khan, R. Z., & Ahmad, M. O. (2017). A survey on load balancing algorithms in cloud computing. International Journal of Autonomic Computing, 2(4), 366-383.
[6] Ghorbani, M., Khaledian, N., & Moazzami, S. (2025). ALBLA: an adaptive load balancing approach in edge-cloud networks utilizing learning automata. Computing, 107(1), 34.
[7] Singhal, S., Sharma, A., Verma, P. K., Kumar, M., Verma, S., Kaur, M., ... & Garc´ıa-Arenas, M. (2024). Energy efficient load balancing algorithm for cloud computing using rock hyrax optimization. IEEE Access.
[8] Adewale, A. A., Obiazi, O., Okokpujie, K., & Koto, O. (2024). Hy- bridization of the Q-learning and honey bee foraging algorithms for load balancing in cloud environments. International Journal of Electrical & Computer Engineering (2088-8708), 14(4).
[9] Khan, A. R. (2024). Dynamic load balancing in cloud computing: optimized RL-based clustering with multi-objective optimized task scheduling. Processes, 12(3), 519.
[10] Ghafir, S., Alam, M. A., Siddiqui, F., & Naaz, S. (2024). Load balanc- ing in cloud computing via intelligent PSO-based feedback controller. Sustainable computing: informatics and systems, 41, 100948.
[11] Al Reshan, M. S., Syed, D., Islam, N., Shaikh, A., Hamdi, M., Elmagzoub, M. A., ... & Talpur, K. H. (2023). A fast converging and globally optimized approach for load balancing in cloud computing. IEEE Access, 11, 11390-11404.
[12] Ramya, K., & Ayothi, S. (2023). Hybrid dingo and whale optimization algorithm-based optimal load balancing for cloud computing environ- ment. Transactions on Emerging Telecommunications Technologies, 34(5), e4760.
[13] Kumar, R., & Agrawal, N. (2022). RBAC-LBRM: an RBAC-based load balancing assisted efficient resource management framework for IoT- edge-fog network. IEEE Sens. Lett. 6 (8), 1–4, Art no. 5501104 (2022).
[14] Zhang, W. Z., Elgendy, I. A., Hammad, M., Iliyasu, A. M., Du, X., Guizani, M., & Abd El-Latif, A. A. (2020). Secure and optimized load balancing for multitier IoT and edge-cloud computing systems. IEEE Internet of Things Journal, 8(10), 8119-8132.
[15] Gulbaz, R., Siddiqui, A. B., Anjum, N., Alotaibi, A. A., Althobaiti, T., & Ramzan, N. (2021). Balancer genetic algorithm—A novel task scheduling optimization approach in cloud computing. Applied Sciences, 11(14), 6244.
[16] RM, S. P., Bhattacharya, S., Maddikunta, P. K. R., Somayaji, S. R. K., Lakshmanna, K., Kaluri, R., ... & Gadekallu, T. R. (2020). Load balancing of energy cloud using wind driven and firefly algorithms in internet of everything. Journal of parallel and distributed computing, 142, 16-26.
[17] Jena, U. K., Das, P. K., & Kabat, M. R. (2022). Hybridization of meta- heuristic algorithm for load balancing in cloud computing environment. Journal of King Saud University-Computer and Information Sciences, 34(6), 2332-2342.
[18] Hussain, A., Aleem, M., Khan, A., Iqbal, M. A., & Islam, M. A. (2018). RALBA: a computation-aware load balancing scheduler for cloud computing. Cluster Computing, 21, 1667-1680.
[19] Mohamad, Z., Mahmoud, A. A., Nik, W. N. S. W., Mohamed, M. A., & Deris, M. M. (2018). A genetic algorithm for optimal job scheduling and load balancing in cloud computing. International Journal of Engineering & Technology, 7(3), 290-294.
[20] Priya, V., Kumar, C. S., & Kannan, R. (2019). Resource scheduling algorithm with load balancing for cloud service provisioning. Applied Soft Computing, 76, 416-424.
[21] Chen, S. L., Chen, Y. Y., & Kuo, S. H. (2017). CLB: A novel load balancing architecture and algorithm for cloud services. Computers & Electrical Engineering, 58, 154-160.
[22] Hasan, R. A., & Mohammed, M. N. (2017). A krill herd behaviour in- spired load balancing of tasks in cloud computing. Studies in Informatics and Control, 26(4), 413-424.
[23] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. Procedia Technology, 10, 340-347.
[24] Mondal, B., Dasgupta, K., & Dutta, P. (2012). Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. Procedia Technology, 4, 783-789.
[25] Singh, A., Juneja, D., & Malhotra, M. (2015). Autonomous agent based load balancing algorithm in cloud computing. Procedia Computer Science, 45, 832-841.