Task Scheduling Schemes of Cloud Computing: A Review

Kishan Chandra Chaurasia, Sandeep Kumar Singh

Department of CSE, SHEAT Group of Institutions, Babatpur, Varanasi (U.P), India

Abstract - Load balancing in cloud computing is a vital technique used to distribute workloads and computing resources efficiently across multiple servers, ensuring optimal resource utilization, minimized response time, and improved system performance. In cloud environments, where resources are dynamically allocated and accessed via the internet, load balancing plays a crucial role in managing traffic, enhancing reliability, and ensuring fault tolerance. Cloud service providers use load balancers to automatically detect and redirect traffic from overloaded servers to those with available capacity, thereby maintaining system stability and user satisfaction. Load balancing techniques can be broadly classified into static and dynamic methods. Static methods rely on pre-defined rules, while dynamic methods adapt in real-time to changing workloads. Effective load balancing contributes to scalability, elasticity, and cost-efficiency, which are foundational principles of cloud computing. Moreover, it ensures service availability even during system failures or maintenance periods by rerouting requests seamlessly. This paper explores various load balancing strategies, compares their effectiveness in different cloud environments, and proposes improvements leveraging AI and predictive analytics. The study underscores the significance of load balancing as an enabler of high-performance, resilient, and user-centric cloud computing infrastructures.

Keywords - Load balancing, Cloud, Computing, Optimization

I. INTRODUCTION

Cloud computing has emerged as a leading innovation in recent years due to significant technological advancements. Initially, it was based on a surplus resource selling model, where users were charged for accessing specific amounts of computing and storage resources. This model has since evolved into a pay-as-you-go approach, enabling users to access configurable computing resources on demand through a network. This strategy benefits service providers by allowing them to rent out their resources for short durations. Meanwhile, businesses and individuals can reduce costs associated with establishing internal data centers by leasing virtual resources, resulting in substantial savings [1][2]. The primary goal of cloud computing is to deliver services through virtualized resources. Each service offered by a provider can be viewed as a task, and the process of allocating and executing these services for users constitutes task management. With the expansion of cloud computing, the demand for processing a larger volume of tasks has increased. Consequently, efficient task allocation and scheduling are critical for enhancing overall cloud performance and ensuring quality of service. Although numerous task scheduling algorithms have been developed, many still face limitations that require further investigation and improvement.

1.1. Cloud Computing Service Models

CC is currently a prominent area of research, with specialized organizations providing extensive computing resources, development environments, and services, as depicted in Figure 1. The categorization of Cloud computing is done into 3 classes according to the services it offers in various fields [3][4]. A wide range of facilities, known as XaaS, encompassing testing, security, collaboration, metamodeling, and more, are obtained from it. The cloud has swiftly become a hub for the proliferation of the "as a service" model, with numerous catch phrases associated with it.

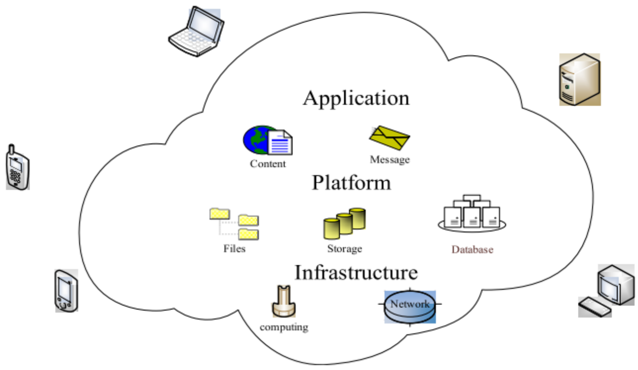

Figure 1: Cloud Computing Service Models

Cloud computing offers three major services that are discussed as follow:

a. Platform as a Service (PaaS): It refers to designers who implement their own software solutions. The fresh applications can be developed easily using already constituted models. These models also streamline maintenance as well as the production deployment of these applications during execution time. These models involve eliminating the need to buy separately as a unit of middleware or service-providing software or applications built through it [5][6].

b. Software as a Service (SaaS): Software as a Service framework of cloud computing hosts an application and delivers it as a service to users who use the service online. There is no need for the customers to manage or support the software when it is hosted externally. However, the client is not impacted when deciding to alter the application by the hosted SP (Service Provider). The idea is to use ready-to-use software and it can be integrated with other models easily. The provider deals with all repairs, updates and keeps the framework running simultaneously.

c. Infrastructure as a Service (IaaS): IaaS is a form of CC model that distributes virtualized computing resources to customer online. This is a robust component of CC with SaaS and PaaS. It is fully preconditioned and maintained on web [7][8]. Its use can save customers from the cost and complexity that they have to face while buying and handling their personal physical servers. Its each resource is provided as a separate service unit and the deployment of merely specific resource is essential for users. The supplier of cloud service is responsible for managing its framework and enables users to focus on installation, configuration and management of their software.

1.2 Cloud Computing Deployment Models

Another way of categorizing cloud computing is through deployment models that are absolutely independent of the service models. The service model can be deployed in different ways, while the deployment model does not affect the service model directly. Nevertheless, the deployment models do affect the security and privacy of the data in the cloud. For example, a cloud-only on-premises or hybrid solution provides an opportunity to safeguard data with restricted access. The deployment model is versatile enough to accommodate multiple service models, and on the other hand, a service model can be deployed over different deployment models. Figure 2 illustrates a general view of cloud computing deployment models [9][10].

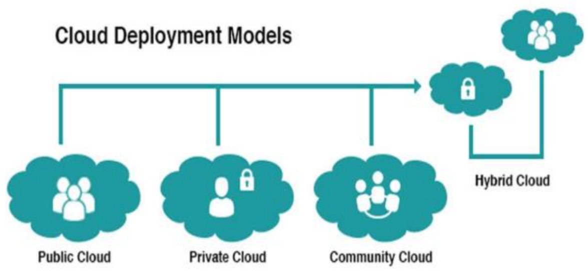

Figure 2: Cloud Computing Deployment Models

Following are different cloud computing frameworks:

a. Public Cloud: In the public cloud model, access to cloud resources is open publically or to great organizations via the internet. Customers don't own the underlying infrastructure; instead, it is managed by a cloud service provider. Services can be provided for free, through subscription, or on a pay-as-you-go basis. Customers have no visibility into the physical location of the cloud services. Multiple companies share the same infrastructure, but their data and applications are logically separated, ensuring services are available only to authorized users. Compared to other systems, these models are least secure because they bear the responsibility of securing all applications. Data stored in public clouds are also vulnerable to malicious activities. Some of these models are Amazon EC2, Google App Engine, and more.

b. Private Cloud: It is a system where the structure is provisioned and accommodated on a dedicated field within the DC of user, exclusively for the use by single company. Access to this service is limited to the company itself, not shared with others. Typically, the company plays a role of CSP to its interior units of profession, offering the potentials of CC deprived of requiring them for establishing their own set up. Private clouds can be hosted within the company's premises or in a DC of a third party. Private clouds are often rendered higher security in contrast to existing clouds as they have internal and restricted nature [11][12].

c. Community Cloud: This system allows numerous companies to share a cloud set-up which form a specific community with common concerns, like healthcare, regulatory obedience, and mission-critical operations. The community cloud aims to provide the assistances of a private cloud, such as to share set-up expenses and pay according to the deployment, while maintaining an extra layer of privacy, safety, and policy agreement which is typically related to earlier clouds. Community clouds can be deployed on-site or in a third-party DC and one of the partner companies focuses on managing them.

d. Hybrid Cloud: In this, the public cloud is put together with private one, connecting them through standardized or proprietary technology to enable the seamless transfer of data and applications. Such a model focuses on integrating services and data via dissimilar types of clouds for developing a cohesive and automatic computation setting. It allows governments to attain the scalable and cost-effective features of public clouds and keep certain applications and data within the confines of their private intranet, enhancing security. It provides the flexibility for businesses to use third-party cloud providers either partially or entirely, tailoring their computing resources to specific needs.

1.3 Characteristics Of Cloud Computing

The NIST definition of cloud computing lists the following five features as necessary to cloud computing. A succinct illustration of the key elements of cloud computing is shown in Figure 3.

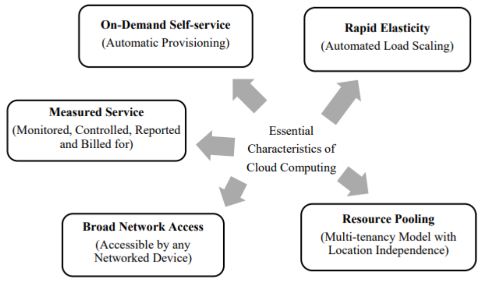

Figure 3: Essential Characteristics of Cloud Computing.

a. On-demand self-service: This is defined in terms of users who, independently of each service provider, are able to unilaterally supply the necessary computer power automatically and without supervision or human involvement. Network storage or server time might be considered computational capabilities.

b. Broad network access: This is stated in terms of computing power that is accessible through a standardized and standard mechanism that is put in place to encourage the usage of heterogeneous platforms, which can be very thin or very thick. This computing power is made available via the internet or network. Workstation PCs, laptops, tablets, and cellphones are a few examples of the platforms [13][14].

c. Resource pooling: Using a multi-tenant approach, the provider pools its computer resources to serve numerous users, dynamically allocating and reassigning various physical and virtual resources based on user demand. The customer may be able to select location at a higher level of abstraction (e.g., country, state, or data center), but they often have no control or knowledge over the precise location of the resources offered. This gives the customer a perception of location-independence.

d. Rapid elasticity: In this instance, the delivered and released elasticity is used to explain computational capabilities. In certain situations, the release could be automatic in order to truly expand both internally and outside. Additionally, this scalability is employed to match consumer demand. From the user's point of view, the capabilities that are made available frequently seem limitless in terms of how much and how quickly they can be used [15][16].

e. Measured service: The cloud system in this instance is automatically optimized and regulated by the leverage of a metering capability that is utilized as an abstraction when it is deemed suitable for a particular kind of service. The resource can also be used for much more, such as reporting, monitoring, and controlling, as it further increases transparency for both the user and the supplier with regard to used services. In this instance, storage, processing, bandwidth, and active user accounts are a few examples of services.

II. LITERATURE REVIEW

2.1 Load Balancing using Metaheuristic Techniques in Cloud Computing

M. Gamal, et.al (2019) presented a hybrid metaheuristics method in which osmotic behavior was integrated with bio-inspired algorithm to balance the load [17]. The osmotic behavior employed the VMs (virtual machines) for their migration in cloud infrastructures. Afterward, this method focused on constructing an OH_BAC (osmotic hybrid artificial bee and ant colony) so that the load was balanced. The simulation outcomes revealed that the constructed algorithm was applicable to mitigate the energy usage and the number of VMs migrations in comparison with the traditional methods.

K. Sekaran, et.al (2019) suggested a new meta-heuristic algorithm known as DFA (dominant firefly algorithm) to optimize the tasks related to balance the load multiple VMs (virtual machines) in the Cloud [18]. Hence, the response efficacy of Cloud servers was maximized due to which the accuracy of m-learning systems was also increased. This algorithm was capable of dealing with the issue of load imbalance and enhancing the experiences of users of m-learning. The suggested algorithm performed well concerning throughput and response time.

S. Pang, et.al (2019) designed an EDA-GA (estimation of distribution algorithm-genetic algorithm) in which EDA and GA algorithms were put together [19]. Primarily, a certain scale of feasible solutions was created using the probability model of EDA. Subsequently, the crossover and mutation operations of Genetic Algorithm were assisted in extending the search range of solutions. Finally, the tasks were allocated to the VMs (virtual machines) via the OSS (optimal scheduling strategy). The designed algorithm converges the data quickly and offered good searching potential. The experimental results obtained on CloudSim indicated that the designed algorithm was useful for mitigating the time to complete the task and enhance the effectiveness of balancing the load.

U. K. Jena, et.al (2020) introduced a new approach to balance the load among the VMs (virtual machines) for which a hybrid technique called QMPSO (modified Particle swarm optimization-improved Q-learning) was implemented [20]. Both the algorithms were integrated for adjusting the velocity of the MPSO. This approach emphasized on maximizing the efficacy of the machine. The waiting time of tasks was optimized such that the load was balanced among the VMs, the throughput of VMs was increased and the balance was maintained among priorities of tasks. The results depicted that the introduced approach was robust as compared to the traditional techniques.

2.1 Comparison Table

Author	Year	Technique Used	Findings
M. Gamal, et.al	2019	OH_BAC (osmotic hybrid artificial bee and ant colony)	The simulation outcomes revealed that the constructed algorithm was applicable to mitigate the energy usage and the number of VMs migrations in comparison with the traditional methods.
K. Sekaran, et.al	2019	DFA (dominant firefly algorithm)	This algorithm was capable of dealing with the issue of load imbalance and enhancing the experiences of users of m-learning and offered better throughput and response time.
S. Pang, et.al	2019	EDA-GA (estimation of distribution algorithm-genetic algorithm)	The experimental results obtained on CloudSim indicated that the designed algorithm was useful for mitigating the time to complete the task and enhance the effectiveness of balancing the load.
U. K. Jena, et.al	2020	QMPSO (modified Particle swarm optimization-improved Q-learning)	The results depicted that the introduced approach was robust as compared to the traditional techniques.

2.2 Load Balancing using Load Balancing Algorithms in Cloud Computing

H. Shen, et.al (2020) established a RIAL (Resource Intensity Aware Load balancing) technique in order to balance the load [21]. This technique was employed to assign diverse weights to dissimilar resources in accordance with their utilization intensity in the PM (physical machine) for alleviating the time and cost so that the load was balance and future load imbalance was avoided. An extension of RIAL was also put forward with 3 additional algorithms. The results of experiments validated that the established technique performed more effectively in comparison with the existing techniques.

S. A. Javadi, et.al (2022) presented an IAL (interference-aware load balancing) known as DIAL whose implementation was directly done in cloud users [22]. This algorithm concentrated on inferring the demand for contended resources on the physical hosts which were not visible to users. The load was shifted from the compromised VMs using the estimates of the co-located load. The results acquired on OpenStack and AWS exhibited the adaptability of the presented algorithm for diminishing the tail latencies up to 70% in comparison with the conventional solutions.

A. Pradhan, et.al (2020) projected a LBMPSO (load balancing technique using modified Particle Swarm Optimization task scheduling) for scheduling the tasks over the available cloud resources which resulted in lessening the makespan and improving the resource usage [23]. The proper information was utilized among the tasks and resources within the datacenter. The CloudSim tool was applied to simulate the projected algorithm. The results of simulation indicated that the projected algorithm was more effective over other algorithms as it assisted in mitigating the makespan and maximizing the resource usage.

J. Mercy Faustina, et.al (2019) developed a SGA_LB (Self-Governing Agent Based Load Balancing Algorithm) in which an AMA (autonomous migration agent) was utilized to balance the dynamic workload [24]. The current behavior of the system was taken in account for balancing the load among the nodes. The CPU load, memory utilization and network load were considered as the attributes of nodes. The results confirmed that the developed algorithm performed well, and proved reliable as well as fault tolerant. Moreover, the developed algorithm was effective to balance the entire load.

2.2 Comparison Table

Author	Year	Technique Used	Findings
H. Shen, et.al	2020	RIAL (Resource Intensity Aware Load balancing) technique	The results of experiments validated that the established technique performed more effectively in comparison with the existing techniques.
S. A. Javadi, et.al	2022	DIAL	The results acquired on OpenStack and AWS exhibited the adaptability of the presented algorithm for diminishing the tail latencies up to 70% in comparison with the conventional solutions.
A. Pradhan, et.al	2020	a LBMPSO (load balancing technique using modified Particle Swarm Optimization task scheduling)	The results of simulation indicated that the projected algorithm was more effective over other algorithms as it assisted in mitigating the makespan and maximizing the resource usage.
J. Mercy Faustina, et.al	2019	SGA_LB (Self-Governing Agent Based Load Balancing Algorithm)	The results confirmed that the developed algorithm performed well, and proved reliable as well as fault tolerant. Moreover, the developed algorithm was effective to balance the entire load.

2.3 Load Balancing using General Techniques in Cloud Computing

F. Tang, et.al (2018) investigated an innovative DLBS (dynamical load-balanced scheduling) algorithm to enhance the network throughput when the workload was balanced in dynamic manner [25]. First of all, diverse heuristic scheduling algorithms were presented on 2 OpenFlow network models in order to balance the data flows time. The results of experimentation revealed the supremacy of the investigated algorithm over others and this algorithm was capable of tackling the higher imbalance degree data flows.

M. Sohani, et.al (2021) recommended a PMHEFT (Predictive Priority-based Modified Heterogeneous Earliest Finish Time) algorithm with the objective of estimating the upcoming resource demands of application [26]. A predictive model was generated to provision the resource in a heterogamous system environment so that the demands of users were met. This algorithm had capacity for diminishing the makespan of a workflow application for which the load was balanced among all the VMs (virtual machines). The experimental outcomes proved that the recommended algorithm was effective with regard to makespan, and energy utilization in contrast to the traditional algorithms.

D. A. Shafiq, et.al (2021) introduced a new algorithmic scheme based on some performance measures (QoS, to prioritize the VMs, and distribute the resource) to optimize the resource and balance the load [18]. The new algorithmic approach tackled different concerns and the existent research gaps as per the obtained results. The introduced LB algorithmic scheme used 78% of resources in average than its counterpart in findings. This approach also performed superior to than its comparable approach in the context of Makespan and execution time.

S. Souravlas, et.al (2022) put forwarded a reasonable proposal in the context of additional workloads to very VM, task load balancing plan [27]. The main objective of this strategy was to improve the average response time and makespan of the architecture in cloud scenarios. The new framework used the load balancer (LBer) behaved like a central server. It adopted the presented impartial workload distribution strategy to distribute arriving tasks fairly and equally between virtual machines in the view of their existing location along with their processing capacity. The comparison of the new framework was done with current frameworks based on PSO and honey bee baiting scheme to balance load. In experimentation, the performance of new framework was better than the classic frameworks in terms of several universal performance indices.

2.3 Comparison Table

Author	Year	Technique Used	Findings
F. Tang, et.al	2018	DLBS (dynamical load-balanced scheduling) algorithm	The results of experimentation revealed the supremacy of the investigated algorithm over others and this algorithm was capable of tackling the higher imbalance degree data flows.
M. Sohani, et.al	2021	PMHEFT (Predictive Priority-based Modified Heterogeneous Earliest Finish Time) algorithm	The experimental outcomes proved that the recommended algorithm was effective with regard to makespan, and energy utilization in contrast to the traditional algorithms.
D. A. Shafiq, et.al	2021	New algorithmic approach	The introduced LB algorithmic scheme used 78% of resources in average than its counterpart in findings. This approach also performed superior to than its comparable approach in the context of Makespan and execution time.
S. Souravlas, et.al	2022	load balancer (LBer)	The performance of new framework was performed better than the classic frameworks in terms of several universal performance indices.

III. CONCLUSION

Load balancing is a cornerstone of efficient and resilient cloud computing, playing a critical role in ensuring the seamless delivery of services to users across diverse applications and platforms. By intelligently distributing workloads across multiple servers and dynamically adjusting to fluctuating traffic demands, load balancing enhances performance, reduces latency, and prevents system overload or failure. The comparative study of various algorithms and techniques reveals that no one-size-fits-all solution exists; instead, the choice of load balancing method depends on specific application needs, traffic patterns, and infrastructure complexity. Emerging technologies such as artificial intelligence, machine learning, and predictive analytics are opening new frontiers in adaptive load balancing, allowing systems to anticipate demand and optimize performance proactively.In conclusion, load balancing is not just a performance optimization tool—it is essential for maintaining the core attributes of cloud computing: scalability, availability, reliability, and cost-efficiency. Continued research and innovation in this field will be pivotal in meeting the growing demands of modern cloud-based systems and ensuring user satisfaction in increasingly complex and dynamic computing environments.

IV. REFERENCES

[1] Robert Adaikalaraj, C. Chandrasekar, “To improve the performance on disk load balancing in a cloud environment using improved Lion optimization with min-max algorithm”, Measurement: Sensors, 2023, Vol. 27, 100834, doi.org/10.1016/j.measen.2023.100834.

[2] K. Sekaran, M. S. Khan, R. Patan, A. H. Gandomi, P. V. Krishna and S. Kallam, "Improving the Response Time of M-Learning and Cloud Computing Environments Using a Dominant Firefly Approach," in IEEE Access, vol. 7, pp. 30203-30212, 2019

[3] S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing," in IEEE Access, vol. 7, pp. 146379-146389, 2019

[4] U. K. Jena, P. K. Das and M. R. Kabat, “Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment”, Journal of King Saud University - Computer and Information Sciences, vol. 7, no. 3, pp. 1159-1163, 1 Feb. 2020

[5] H. Shen and L. Chen, "A Resource Usage Intensity Aware Load Balancing Method for Virtual Machine Migration in Cloud Datacenters," in IEEE Transactions on Cloud Computing, vol. 8, no. 1, pp. 17-31, 1 Jan.-March 2020

[6] S. A. Javadi and A. Gandhi, "User-Centric Interference-Aware Load Balancing for Cloud-Deployed Applications," in IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 736-748, 1 Jan.-March 2022

[7] A Mrhari and Y. Hadi, "A Load Balancing Algorithm in Cloud Computing Based on Modified Particle Swarm Optimization and Game Theory," 2019 4th World Conference on Complex Systems (WCCS), Ouarzazate, Morocco, 2019, pp. 1-6, doi: 10.1109/ICoCS.2019.8930807.

[8] M. Menaka, K.S. Sendhil Kumar, “Supportive particle swarm optimization with time-conscious scheduling (SPSO-TCS) algorithm in cloud computing for optimized load balancing”, International Journal of Cognitive Computing in Engineering, 2024, Vol. 5, Pages 192-198, doi.org/10.1016/j.ijcce.2024.05.002.

[9] A. Pradhan and S. K. Bisoy, “A novel load balancing technique for cloud computing platform based on PSO”, Journal of King Saud University - Computer and Information Sciences, vol. 2, no. 45, pp. 672-680, 22 Oct. 2020

[10] J. Gao, "Simulation Design of Load Balancing Optimization for Cloud Computing Data Stream Storage Based on Big Data Algorithms," 2024 Asia-Pacific Conference on Software Engineering, Social Network Analysis and Intelligent Computing (SSAIC), New Delhi, India, 2024, pp. 931-936, doi: 10.1109/SSAIC61213.2024.00188.

[11] A. Francis Saviour Devaraj, Mohamed Elhoseny, S. Dhanasekaran, E. Laxmi Lydia, K. Shankar, “Hybridization of firefly and Improved Multi-Objective Particle Swarm Optimization algorithm for energy efficient load balancing in Cloud Computing environments”, Journal of Parallel and Distributed Computing, 2020, Vol. 142, Pages 36-45, doi.org/10.1016/j.jpdc.2020.03.022.

[12] B. Kruekaew and W. Kimpan, "Multi-Objective Task Scheduling Optimization for Load Balancing in Cloud Computing Environment Using Hybrid Artificial Bee Colony Algorithm with Reinforcement Learning," in IEEE Access, vol. 10, pp. 17803-17818, 2022, doi: 10.1109/ACCESS.2022.3149955.

[13] G. Muneeswari, Jhansi Bharathi Madavarapu, R. Ramani, C. Rajeshkumar, C. John Clement Singh, “GEP optimization for load balancing of virtual machines (LBVM) in cloud computing”, Measurement: Sensors, 2024, Vol. 33, 101076, doi.org/10.1016/j.measen.2024.101076.

[14] K. A. Kumari, T. Soujanya, Z. Alsalami, I. Rohini and B. Dhandayuthapani V, "Fire Hawk Optimization based Multi-objective Dynamic Load Balancing in Cloud Computing," 2024 International Conference on Distributed Computing and

Optimization Techniques (ICDCOT), Bengaluru, India, 2024, pp. 1-4, doi: 10.1109/ICDCOT61034.2024.10516057.

[15] L Abualigah, Ahmad MohdAziz Hussein, Mohammad H. Almomani, Raed Abu Zitar, Hazem Migdady, Ahmed Ibrahim Alzahrani, Ayed Alwadain, "Improved synergistic swarm optimization algorithm to optimize task scheduling problems in cloud computing", Sustainable Computing: Informatics and Systems, 2024, Vol. 43, 101012, doi.org/10.1016/j.suscom.2024.101012.

[16] S. Singhal et al., "Energy Efficient Load Balancing Algorithm for Cloud Computing Using Rock Hyrax Optimization," in IEEE Access, vol. 12, pp. 48737-48749, 2024, doi: 10.1109/ACCESS.2024.3380159.

[17] M. Gamal, R. Rizk, H. Mahdi and B. E. Elnaghi, "Osmotic Bio-Inspired Load Balancing Algorithm in Cloud Computing," in IEEE Access, vol. 7, pp. 42735-42744, 2019

[18] K. Sekaran, M. S. Khan, R. Patan, A. H. Gandomi, P. V. Krishna and S. Kallam, "Improving the Response Time of M-Learning and Cloud Computing Environments Using a Dominant Firefly Approach," in IEEE Access, vol. 7, pp. 30203-30212, 2019

[19] S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing," in IEEE Access, vol. 7, pp. 146379-146389, 2019

[20] U. K. Jena, P. K. Das and M. R. Kabat, "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment", Journal of King Saud University - Computer and Information Sciences, vol. 7, no. 3, pp. 1159-1163, 1 Feb. 2020

[21] H. Shen and L. Chen, "A Resource Usage Intensity Aware Load Balancing Method for Virtual Machine Migration in Cloud Datacenters," in IEEE Transactions on Cloud Computing, vol. 8, no. 1, pp. 17-31, 1 Jan.-March 2020

[22] S. A. Javadi and A. Gandhi, "User-Centric Interference-Aware Load Balancing for Cloud-Deployed Applications," in IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 736-748, 1 Jan.-March 2022

[23] A. Pradhan and S. K. Bisoy, "A novel load balancing technique for cloud computing platform based on PSO", Journal of King Saud University - Computer and Information Sciences, vol. 2, no. 45, pp. 672-680, 22 Oct. 2020

[24] J. Mercy Faustina, B. Pavithra, S. Suchitra and P. Subbulakshmi, "Load Balancing in Cloud Environment using Self-Governing Agent," 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), 2019, pp. 480-483

[25] F. Tang, L. T. Yang, C. Tang, J. Li and M. Guo, "A Dynamical and Load-Balanced Flow Scheduling Approach for Big Data Centers in Clouds," in IEEE Transactions on Cloud Computing, vol. 6, no. 4, pp. 915-928, 1 Oct.-Dec. 2018

[26] M. Sohani and S. C. Jain, "A Predictive Priority-Based Dynamic Resource Provisioning Scheme With Load Balancing in Heterogeneous Cloud Computing," in IEEE Access, vol. 9, pp. 62653-62664, 2021

[27] D. A. Shafiq, N. Z. Jhanjhi, A. Abdullah and M. A. Alzain, "A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications," in IEEE Access, vol. 9, pp. 41731-41744, 2021

[28] S. Souravlas, S. D. Anastasiadou, N. Tantalaki and S. Katsavounis, "A Fair, Dynamic Load Balanced Task Distribution Strategy for Heterogeneous Cloud Platforms Based on Markov Process Modeling," in IEEE Access, vol. 10, pp. 26149-26162, 2022