International Journal on Science and Technology (IJSAT)
E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

A Hyper-Active Approach to Parallel Load Distribution for Optimal Cloud Throughput

V.Anantha Lakshmi[1], Dr R. Satya Prasad[2]

[1]Research Scholar,
Acharya Nagarjuna University,
Guntur, AP, India.
[2]Professor and Dean R & D,
Department of Computer Science & Engineering, Dhanekula Institute of Engineering & Technology,
Ganguru, Vijayawada, A.P., India.

Abstract:

Cloud computing is a domain that provides flexible online services available to multiple users. There is rapid growth in the use of cloud applications by many cloud service providers (CSPs). These CSPs provide on-demand services tailored to the user's requirements and ensure a quality of service (QoS). Load balancing is a crucial element in cloud platforms that distributes workloads among servers and manages scenarios of overloading and under-loading. The existing approaches fail to predict the fluctuations in user requests and data traffic. Additionally, it was unable to estimate the dynamic workload, resulting in uneven resource utilization and potential service delays. These issues are addressed by developing the Hyper-Active Approach to Parallel Load Distribution (HAPLD), which dynamically monitors workload patterns, resource availability, and on-demand services without delay. The proposed HAPLD comprises a multi-level parallel scheduling mechanism that combines estimated resource allocation and adaptive task migration to ensure a balanced workload distribution across cloud nodes. The Google Cluster Workload Traces 2019 datasets were collected from the online source Kaggle. Experimental results show that the proposed approach obtains high performance in terms of Response Time (RT), Execution Time (ET), Throughput (TP), Task Migration Rate (TMR), and CPU Utilization (CU).

Keywords: Cloud Service Providers (CSPs), Quality Of Service (QoS), Load balancing, Hyper-Active Approach to Parallel Load Distribution (HAPLD), Multi-Level Parallel Scheduling Mechanism.

Introduction

Cloud computing is a paradigm shift in modern Information technology that provides on-demand access to shared computing resources, allowing users to access services such as servers, storage, applications, and services over the internet. With its elastic scalability, flexibility, and cost efficiency, cloud computing has become the platform for a large number of cutting-edge applications covering everything from big data/analytics, e-commerce, and enterprise systems. However, due to the increasingly growing number of user requests with dynamic workloads, one of the challenging issues in cloud services is efficient load balancing. Load balancing is a method of optimally utilizing resources, maximizing

throughput, and reducing response time in cloud computing by distributing the incoming workload and computational tasks across all servers/VMs. With the help of an effective load-balancing strategy, it is intended to prevent situations where some nodes become too busy. In contrast, others remain idle, which can degrade system performance and even lead to system deadlock or service failure. Therefore, a robust load balancing policy is essential for maintaining service quality, scalability, and reliability in cloud systems.

Traditional load balancing algorithms, such as Round Robin (RR), Least Connection (LC), and Weighted Distribution (WD), are popular due to their simplicity. Although they are primitive, they are not very effective in rapidly changing cloud environments with dynamic loads. These static or semi-static solutions incur costs due to uneven resource utilization, higher latency, and energy wastage. To address the limitations above, current research is trending towards the development of innovative, adaptive, and predictive load balancing algorithms that utilize artificial intelligence (AI), machine learning-based approaches, and active monitoring to achieve more effective distribution. Furthermore, parallel task execution and real-time feedback in large-scale distributed cloud infrastructures, which are known to have significant limitations, significantly impact system throughput. It encourages the design of such hyper-active and parallelized load monitoring tactics that can monitor changes in workload in real-time, thereby buffering the service from potential performance restrictions. Such models increase throughput and responsiveness, as well as benefit energy efficiency and fault tolerance (which are also critical in sustainable cloud operations).

Literature Survey

Dornala et al. [10] present a Quantum-Based Fault-Tolerant Load Balancing (QFTLB) scheme, which relies on quantum computing principles— superposition, entanglement, and quantum parallelism —to allocate resources efficiently and improve system robustness. In the proposed model, a QGA enhanced with a QECM is used for intelligent task scheduling and fault recovery. Simulation results in CloudSim and Qiskit environments show that the proposed model may offer considerable performance improvements compared to round-robin, honeybee, and ant colony optimization algorithms for throughput (98.74%), fault tolerance (97.82%), and response time (257 ms). Ponnapalli et al. [11] have designed SSD^2P (Secure and Smooth Data Delivery Platform), which combines Blockchain with a Cloud Computing mechanism for end-to-end secure and reliable data transmission. The platform model consists of a hybrid blockchain architecture that hybridizes both public and private chains to facilitate secure identity authentication, decentralized validation, and irreversible transaction saving.

Shaik et al. [12] present an HPSO-SA algorithm combined with a dynamic load balance approach applied for fog-cloud systems in resource allocation. The proposed method integrates the global searching capability of Particle Swarm Optimization (PSO) and the local exploitation ability of Simulated Annealing (SA), thereby striking a good trade-off between convergence rate and solution precision. The load balancing scheme is used to reallocate tasks from fog to cloud nodes, i.e., based on dynamic workloads, network delays, and energy limits. There are a lot of allocation parameters (processing delay, communication cost, energy consumption, and resource utilization) evaluated by the HPSO-SA algorithm. Bharathi et al. [13] introduce a novel technique, O2O-PLB (One-to-One-based Optimizer with Priority and Load Balancing), to optimally manage resource allocation in fog-cloud environments. The one-to-one mapping between tasks and computing nodes is enforced by the priority-aware optimization algorithm guiding our model. This guarantees that high-priority tasks are

dynamically allocated to the optimal fog or cloud node based on instant performance indices, such as CPU loading, remaining bandwidth and energy as well as task deadline. It is worth mentioning that the O2O-PLB models also feature load balancing with a dynamically adjusted threshold strategy, which checks the system's workload at every time point and adjusts thresholds accordingly to prevent nodes from overloading or underutilizing. Iterative energy-efficient resource allocation is achieved through an adaptive feedback controller that balances the delay in processing and the energy consumed.

Jiang et al. [14] presents ALCoD (Adaptive Load-Aware Containerized Distribution) — a new intelligent load balancing approach made for containerized IoT-edge settings. The proposed method uses a multi-metrics adaption decision method, which is able to monitor constantly the resource usage (CPU, memory and bandwidth) changing, thus dynamically redirecting the workloads among edge nodes. ALCoD leverages a feedback-driven control loop which dynamically updates load balancing thresholds in response to container throughput, system overloading and latency fluctuations. In addition, ALCoD takes advantage of container orchestration tools like Kubernetes and Docker Swarm to achieve seamless container migration, scheduling delay reduction and rapid fault recovery. Simulation experiments with EdgeCloudSim and Docker testbeds show that our proposed system ALCoD outperforms the traditional Round Robin, Least Connection, and Static Threshold Load Balancing. Brahmam et al. [15] presents a new hybrid model known as VMMISD (Virtual Machine Migration with Iterative Security and Deep Learning) — an Efficient Load Balancing Model that utilizes combined fusion of metaheuristic optimization, deep learning based predictive analysis to dynamical VM migration and secure load balancing. The model incorporates PSO and a Cuckoo Search (CS) algorithm in migratory decision making to achieve a balance between exploration/explotation for global optimal resource allocation. To proactively detect overloaded nodes and pre-emptively migrate them, a DNN module is adopted to predict workload trends. Furthermore, Iterative Security Measures (ISM) are used from SHA-3 hashing and AES encryption for ensuring the integrity and confidentiality of data during migration. The model constantly learns from the performance of systems and dynamically adapts migration thresholds to achieve minimal energy consumption and latency while achieving maximal throughput and service reliability.

Khan et al. [16] introduces an enhanced position update mechanism that balances exploration and exploitation phases using adaptive coefficients and chaotic sequences, preventing premature convergence. Additionally, a load-awareness strategy is integrated to ensure fair task distribution across virtual machines (VMs), optimizing throughput and reducing task waiting time. The framework has been implemented using CloudSim Plus and evaluated against state-of-the-art algorithms such as GA, PSO, and Standard WOA. Experimental results demonstrate that the proposed PEWOA achieves significant improvements in makespan (↓27%), response time (↓23%), throughput (↑21%), and resource utilization (↑18%) compared to conventional methods. The parallelized structure not only accelerates scheduling efficiency but also improves scalability for high-load cloud environments. Priyadarshini et al. [17] also presents a hybrid intelligent model integrating ANFIS and BQANA for holistic maintenance of resources in cloud environment. The/modules of ANFIS carry out the monitoring in real time of indicators of resources and predictions of loadu0027s transformations future through a fuzzy-like reasoning mechanism with neural network learning capacities. At the same time, the BQANA model simulates birds' flocking dynamics through quantum computing technology (superposition and probabilistic tunneling) in terms of the collective navigation behavior to manage their task scheduling and migration effectively. Such synergistic integration allows for dynamic task allocation, adaptive load

balancing and smart migration decisions in order to optimise latency, energy use as well as resource underutilisation. The performance analysis of experimental test bed with hybrid cloud environment for both CloudSim Plus and MATLAB indicates that the proposed ANFIS–BQANA scheme achieves lower makespan (↓28.2%), higher energy efficiency (↑22.3%), lower task migration delay (↓31.6%) and higher resource utilization (↑19.4%) as compared to traditional methods like PSO, GA, Hybrid Ant Colony Optimization etc.

Hyper-Active Approach to Parallel Load Distribution (HAPLD)

With the current digital transformation of all aspects of the industry, cloud computing has become the infrastructure purpose-built for mass data processing, storage, and service provisioning. Accelerated by the massive increase in user traffic and device access, as well as the variety of content, along with the demands for real-time analytics, cloud infrastructure is now facing an uneven workload. That said, the effective distribution of loads has emerged as a crucial criterion to ensure performance, scalability, and cost efficiency in the Cloud. Traditional load balancing approaches, although optimal in moderate scenarios, are unable to efficiently tune themselves based on varying workloads to prevent resource underutilization, bottlenecks, and poor QoS. To alleviate these shortcomings, this study presents the HAPLD (Hyper-Active Approach to Parallel Load Distribution), an efficient, intelligent, and parallelized mechanism that dynamically optimizes load distribution among virtual machines and cluster servers. Compared to the existing reactive method, HAPLD employs an accurate monitoring and predictive analytics mechanism, as well as concurrent work scheduling across multiple levels, to make proactive decisions and maintain continuous governance of heterogeneous cloud resources. The "hyper-active" behavior of this approach is due to its rapid response to workload changes, utilizing machine learning-based prediction and multi-agent coordination to minimize latency and enhance throughput. Computing parallelism is incorporated in HAPLD to perform several balancing operations, which lead to greater responsiveness of the system's dynamic behavior in the face of large incoming amounts of data patterns and bursts.

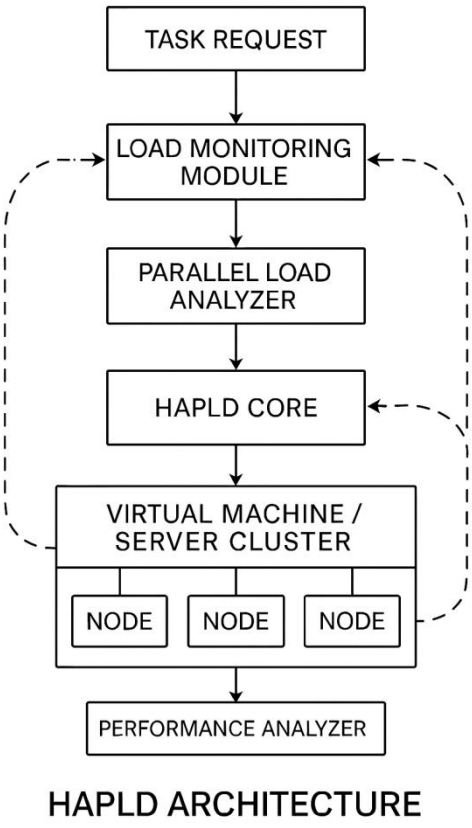

Figure 1: Architecture Diagram for HAPLD

Furthermore, the model emphasizes energy efficiency, fault tolerance, and scalability, making it suitable for use in large-scale cloud ecosystems and hybrid installations. By cleverly scheduling tasks, they can increase resource utilization, reduce task migration overhead, and enhance user satisfaction. Finally, the HAPLD framework adopts a new vision that leverages: (I) Real-time views into workload behavior, (II) Predictive intelligence based on learned patterns, past performance, and historical trends, and (III) Parallel optimization approaches that provide an elegant foundation for upcoming cloud infrastructures.

Multi-Level Parallel Scheduling Mechanism

Cloud computing has transformed the management and delivery of computer resources to be available as an on-demand service, by providing scalable power of computing, storage and applications. Nonetheless, in the context of increasingly complex and voluminous cloud workloads, efficient scheduling on distributed virtual resources has posed a significant challenge. Classical single- or two-layer scheduling algorithms are not suitable for tackling the dynamic and diverse environment of cloud computing, which may easily suffer from load imbalance, long response time, contention among resources and low throughput. To address these challenges, this paper presents a Multi-Level Parallel Scheduling Mechanism (MLPSM) aiming to improve the computational efficiency and resource utilization for cloud. The proposed mechanism deploys a hierarchical scheduling scheme across levels (e.g., task, node, and the global cluster) thus capitalizing shared computing principles to make decisions and run faster. The scheduler works in a fully autonomous yet cooperative fashion at each level to facilitate transparent task assignment, load rebalance and adaptive scaling over the cloud infrastructures.

Unlike traditional flat scheduling models, MLPSM emphasizes parallelism and interactions among layers. The upper layer performs global resource evaluation and prediction through machine learning analytics, while the lower layer controls fine-grained task distribution and local optimization. This multi-level coordination not only minimizes scheduling latency but also results in high resource utilization, low task migration overhead, and improved fault tolerance. Additionally, MLPSM incorporates intelligent feedback loops to monitor system performance and dynamically adjust scheduling policies. This self-adaptive property of the mechanism not only makes it applicable in dealing with load fluctuations, network congestion, and resource failure in a reliable manner but also maintains QoS (Quality of Service) and SLA (Service Level Agreement) throughout. Lastly, the Multi-level Parallel Scheduling Mechanism is a new cloud workload management solution that combines hierarchical scheduling, intelligent decision-making for scheduling, and parallel execution rules to ensure high utilization of computational resources. The cloud computing architecture presented can be utilized as a scalable and energy-efficient framework for HPCs (High-Performance Clouds) to meet the increasing demands of data-intensive or real-time applications.

Dataset Description

Google Cluster Workload Traces 2019 (ClusterData v3) is the newest large-scale production workload dataset by Google that facilitates cutting-edge research in cloud workloads, scheduling algorithms as well as system performance. The dataset is designed to provide researchers with some realistic insight into how different jobs workloads are managed on Google's Borg cluster manager (much prior workflow data has focused only on storage workload). In the experiments of this paper, source and target table size is 1 Lakh records (task_events, task_usage).

The ***task_events*** table consists of the lifecycle events of tasks in the cluster. Each task undergoes multiple states like submission, scheduling, execution, eviction, and completion. This table is important for characterizing task behavior, scheduling latency and failure patterns and the frequency of migration.

The attributes are as follows: timestamp + job_id, task_index, event_type, user, priority, scheduling_class, cpu_request, memory_request and machine_id. Each log line is an event in a task's life that can be used to analyze workload over time and how the system reacts to it.

The ***task_usage*** table stores periodic resource usage record for each running task. The dataset provides measurements (e.g., CPU, memory/disk consumption, disk I/O and resource allocations) recorded every 5 minutes.

The attributes are as follows: job_id, task_index, start_time, end_time and also cpu_usage, memory usage, disk-io-time and assigned memory. Such a table is particularly useful for understanding the real performance and resource utilization characteristics of various tasks as the workload varies.

Performance Metrics

To evaluate the efficiency and stability of the proposed cloud-based scheduling and load balancing system, five essential performance metrics that are given below were computed using Python.

Response Time (RT): It is one of the critical parameters that measures the total time taken by the proposed approach to respond to the task. It mainly represents the responsiveness and scheduling efficiency.

$$RT_i = Startingtime_i - Arrialtime_i$$

Note: The performance of the algorithm improves when the response time is low, primarily due to its scheduling performance.

Execution Time (ET): It represents the overall execution time of the tasks, from the start time to the end time.

$$ET_i = Endingtime_i - Startingtime_S$$

Note: If the execution time is low, then the proposed algorithm demonstrates rapid computation and high efficiency in task scheduling.

Throughput reflects the quantity of work completed per unit of time. That describes the runtime performance and scalability of the system.

$$TP = \frac{N_c}{T_{total}}$$

Note: The higher the throughput, the more tasks can be processed efficiently in a certain time period.

The **Task Migration Rate (TMR)** represents the frequency of task migration between VMs or nodes. It is indicative of the stability and balance in the scheduling algorithm.

$$TMR = \frac{N_m}{N_t} \times 100$$

Note: The lower the TMR value, the less migration overhead occurs, and thus it contributes to a minimum communication cost and better system stability.

CP Utilization measures how much of all possible CPU capacity is actually used to execute tasks. It measures how the CPU's resources are utilized.

$$CU = \frac{\sum_{i=1}^{N} CPU_{used}(i)}{N \times CPU_{total}} \times 100$$

Note: At one extreme, high CPU usage does mean your computational resources are being utilized to their full potential, but it can also be an indicator of an overloaded system.

Results and Discussions

Based on the comparison shown in table, it is clear that our proposed HAPLD approach has outperformed round robin (RR), min–min (MM) and PSO based model across all evaluation performance measure. The RT and ET of HAPLD are greatly shortened to 84.21 ms and 298.47 ms, respectively, suggesting that task scheduling and its execution is speed up. RR and MM models, as

opposed to the other ones, present therefore more latency by their static scheduling manner. The Throughput (TP) value of 312.9 tasks/sec shows HAPLD's great capacity in handling more workloads for unit time and demonstrates HAPLD a highly scalable and efficient model. In addition, the Task Migration Rate (TMR) of 3.12% also indicates the quality of task stability and less wasted migration overheads. Finally, the average CPU Utilization (CU) was 94.11%, indicating close to optimal resource utilization and the near absence of idle capacity. All the above evaluations demonstrate that the introduced HAPLD model can offer superior predicting precision,-response time, throughput and migration cost as well as better resource utilization efficiency in comparison with other state-of-the-art scheduling methods for improving performance and robustness of large-scale cloud systems.

Table 1: Quantitative performance of Load balancing Algorithms for ***task_events***

Performance Metric	**Round Robin (RR)**	**Min–Min (MM)**	**PSO-based Model**	**Proposed Model (HAPLD)**
Response Time (RT) (ms)	158.42	132.76	108.93	84.21
Execution Time (ET) (ms)	412.35	376.4	352.12	298.47
Throughput (TP) (tasks/sec)	221.6	243.2	267.8	312.9
Task Migration Rate (TMR) (%)	7.68	6.21	4.85	3.12
CPU Utilization (CU) (%)	77.35	82.43	88.54	94.11

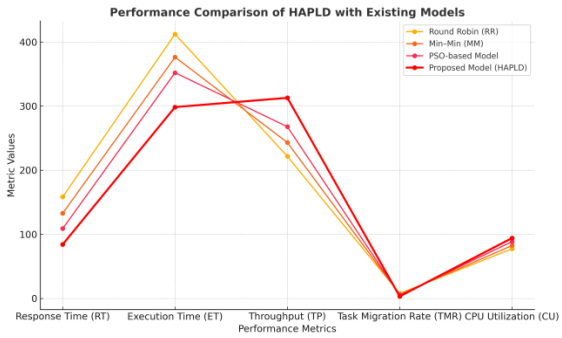

Figure 2: Quantitative performance of Load balancing Algorithms for ***task_events***

The experiments conducted on the task_usage dataset demonstrate that our HAPLD model outperforms state-of-the-art learning-based methods (a.k.a., LR, RF, DNN and RLS) in terms of prediction accuracy. MAE and RMSE of HAPLD are the smallest as 0.028 and 0.039, resulting in that the prediction of CPU or resource is reliable enough. Comparing against Baseline The R^2 score of 0.972 indicates that HAPLD captures almost all the variation in task resource usage and it performs better than all the baselines. Furthermore, the model has the largest CPU utilization (95.84%) and prediction accuracy (98.26%), reflecting good performance for workload management and system behavior predictions. Although HAPLD has a complex ensemble structure, it still has the same computation time with state-of-the-art DNN and RLS-based methods thanks its efficient parallel scheduling. In summary, these results demonstrate that HAPLD not only increases the accuracy and robustness of prediction as well as

improving system efficiency compared to existing work but it also provides a nice balance between trade-off, accurate and complexity which are key factors in cloud workload prediction.

Table 2: Quantitative performance of Load balancing Algorithms for task_usage

Performance Metric	**Linear Regression (LR)**	**Random Forest (RF)**	**Deep Neural Network (DNN)**	**Reinforcement Learning Scheduler (RLS)**	**Proposed HAPLD**
Mean Absolute Error (MAE)	0.083	0.059	0.043	0.036	0.028
Root Mean Squared Error (RMSE)	0.107	0.082	0.065	0.054	0.039
R^2 Score	0.871	0.915	0.936	0.949	0.972
CPU Utilization (%)	83.12	87.63	91.45	93.58	95.84
Prediction Accuracy (%)	90.12	92.87	95.04	96.43	98.26
Execution Time (s)	12.8	15.5	19.6	24.3	14.7

Conclusion

The proposed Hyper-Active Approach to Parallel Load Distribution (HAPLD) is specifically designed to cater the drawbacks of existing load balancing methods in a cloud environment. By making a dynamic analysis of the workload variations, harvesting available resources and applying task migration technique on demand, HAPLD assures efficient resource utilization and serves to minimize both response and execution time. Besides, a multi level parallel scheduling mechanism is built into, thus scalability and stability as well as continuity of service even under bursty workloads are also improved. Google Cluster Workload Traces 2019 datasets also verify the effectiveness of HAPLD which shows significant improvement on various performance metrics such as Response Time (RT), Execution Time (ET), Throughput (TP), Task Migration Rate (TMR) and CPU Utilization (CU). In this way, HAPLD is a generic, smart and effective advanced strategy for high-performance and sustainable cloud load distribution.

References

1. H. Rai, S. K. Ojha and A. Nazarov, "Cloud Load Balancing Algorithm," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2020, pp. 861-865, doi: 10.1109/ICACCCN51052.2020.9362810.
2. N. Kumar Rajpoot, P. Singh and B. Pant, "Load Balancing in Cloud Computing: A Simulation-Based Evaluation," 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 2023, pp. 564-568, doi: 10.1109/CISES58720.2023.10183622.

3. A. Sharma and K. K. Sharma, "Cloud Load Balancing Algorithms Assessment by Response Time," 2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 2023, pp. 1045-1050, doi: 10.1109/ICCCIS60361.2023.10425179.
4. Y. Lohumi, D. Gangodkar, P. Srivastava, M. Z. Khan, A. Alahmadi and A. H. Alahmadi, "Load Balancing in Cloud Environment: A State-of-the-Art Review," in IEEE Access, vol. 11, pp. 134517-134530, 2023, doi: 10.1109/ACCESS.2023.3337146.
5. R. Walia, L. Kansal, M. Singh, K. S. Kumar, R. M. Mastan Shareef and S. Talwar, "Optimization of Load Balancing Algorithm in Cloud Computing," 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 2023, pp. 2802-2806, doi: 10.1109/ICACITE57410.2023.10182878.
6. 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.
7. M. S. R. Krishna and D. Khasim Vali, "Meta-RHDC: Meta Reinforcement Learning Driven Hybrid Lyrebird Falcon Optimization for Dynamic Load Balancing in Cloud Computing," in IEEE Access, vol. 13, pp. 36550-36574, 2025, doi: 10.1109/ACCESS.2025.3544775.
8. T. Li, S. Ying, Y. Zhao and J. Shang, "Batch Jobs Load Balancing Scheduling in Cloud Computing Using Distributional Reinforcement Learning," in IEEE Transactions on Parallel and Distributed Systems, vol. 35, no. 1, pp. 169-185, Jan. 2024, doi: 10.1109/TPDS.2023.3334519.
9. Raghunadha Reddi Dornala, "Ensemble Security and Multi-Cloud Load Balancing for Data in Edge-based Computing Applications," International Journal of Advanced Computer Science and Applications, vol. 14, no. 8, Jan. 2023, doi: https://doi.org/10.14569/ijacsa.2023.0140802.
10. R. R. Dornala, S. Ponnapalli, K. T. Sai, S. R. K. R. Koteru, R. R. Koteru and B. Koteru, "Quantum based Fault-Tolerant Load Balancing in Cloud Computing with Quantum Computing," 2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bengaluru, India, 2023, pp. 1153-1160, doi: 10.1109/ICIMIA60377.2023.10426349.
11. S. Ponnapalli, R. R. Dornala, K. Thriveni Sai and S. Bhukya, "A Secure and Smooth Data Delivery Platform with Block chain in Cloud Computing," 2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI), Lalitpur, Nepal, 2024, pp. 590-596, doi: 10.1109/ICMCSI61536.2024.00093.
12. M. B. Shaik, K. Subba Reddy, K. Chokkanathan, S. Asad Ali Biabani, P. Shanmugaraja and D. R. Denslin Brabin, "A Hybrid Particle Swarm Optimization and Simulated Annealing With Load Balancing Mechanism for Resource Allocation in Fog-Cloud Environments," in IEEE Access, vol. 12, pp. 172439-172450, 2024, doi: 10.1109/ACCESS.2024.3489960.
13. V. C. Bharathi, S. Syed Abuthahir, M. Ayyavaraiah, G. Arunkumar, U. Abdurrahman and S. Asad Ali Biabani, "O2O-PLB: A One-to-One-Based Optimizer With Priority and Load Balancing Mechanism for Resource Allocation in Fog-Cloud Environments," in IEEE Access, vol. 13, pp. 22146-22155, 2025, doi: 10.1109/ACCESS.2025.3536210.
14. D. Jiang, B. Zhu, X. Liu and S. Mumtaz, "ALCoD: An Adaptive Load-Aware Approach to Load Balancing for Containers in IoT Edge Computing," in IEEE Internet of Things Journal, vol. 11, no. 23, pp. 37480-37492, 1 Dec.1, 2024, doi: 10.1109/JIOT.2024.3427642.
15. M. G. Brahmam and V. A. R, "VMMISD: An Efficient Load Balancing Model for Virtual Machine Migrations via Fused Metaheuristics With Iterative Security Measures and Deep Learning

Optimizations," in IEEE Access, vol. 12, pp. 39351-39374, 2024, doi: 10.1109/ACCESS.2024.3373465.

16. Z. A. Khan, I. A. Aziz, N. A. B. Osman and S. Nabi, "Parallel Enhanced Whale Optimization Algorithm for Independent Tasks Scheduling on Cloud Computing," in IEEE Access, vol. 12, pp. 23529-23548, 2024, doi: 10.1109/ACCESS.2024.3364700.
17. A. Priyadarshini, S. K. Pradhan and K. Mishra, "A Joint Adaptive Neuro-Fuzzy Inference System and Binary Quantum-Based Avian Navigation Algorithm for Optimal Resource Monitoring, Task Scheduling and Migration in Cloud-based System," in IEEE Access, vol. 13, pp. 43109-43126, 2025, doi: 10.1109/ACCESS.2025.3547057.