DOI: http://dx.doi.org/10.26483/ijarcs.v16i2.7197

Volume 16, No. 2, March-April 2025

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

A COMPREHENSIVE LITERATURE STUDY ON LOAD BALANCING METHODS IN CLOUD COMPUTING WITH INTERNET OF THINGS

M. Parveen Taj
Phd Research Scholar,
Department of Computer Science,
PPG College of Arts and Science,
Coimbatore-641035, Tamil Nadu, India

Dr.N.Muthumani
Principal,
PPG College of Arts and Science,
Coimbatore - 641035,
Tamil Nadu, India

Abstract: Incorporating IoT devices such as smart appliances, routers, sensors, and cellphones into a singular network is the novel concept behind the Internet of Things. Cloud computing manages the storage and the processing of data produced by these IoT devices which require extensive amounts of data. The cloud also enables real time analysis of data that allows fast decision-making. In settings that require the integration of Cloud Computing and Internet of Things like Cloud-IoT, making the most out of the infrastructure with effective load balancing techniques becomes critical. These techniques serve the purpose of drowning out the faults, increasing throughput, and lowering execution and response time while also ensuring that there is fair workload distribution. With the intent of balancing the distinct and complex features of IoT, this articles aims to provide a summary on the various techniques of load balancing within a cloud environment, covering both centralized techniques as well as more advanced ones. This study digs deeper into the performance metrics and challenges, as well as application scenarios of all solutions, including energy usage, execution period, scalability, and workflow adaptability. Lastly, we present a comparison of the leading load sharing algorithm's effectiveness on a few tested IoT cloud systems.

Keywords: Load balancing, Internet of Things (IoT), cloud computing, Energy Consumption, Execution time

I. INTRODUCTION

The quick development of IoT devices—such as smart homes, healthcare, transportation, and industrial automation—has affected many different sectors [1]. The enormous volumes of data these devices generate call for real-time storing and processing. Growing in relevance inside IoT systems as a tool to achieve these goals, cloud computing offers scalable, flexible, reasonably priced solutions. Cloud computing has the architecture needed to handle enormous amounts of data. Combining the Internet of Things (IoT) with cloud computing generates a strong synergy that opens various applications to access advanced analytics, remote management, and real-time data processing [2]. Once gathered and sent by IoT devices, a great volume of data is processed, examined, and kept on the cloud. Apart from providing the processing capability required for complex operations, cloud platforms are scalable and elastic—qualities crucial for enabling the exponential growth of IoT devices [3]. Among the most urgent issues with IoT-cloud systems, though, are reducing latency, preserving dependability, and keeping resource use low. Dealing with these problems depends much on using effective load balancing techniques.

Load balancing aims to enhance efficiency, prevent straining any one computing resource, and guarantee that all of the resources of a system are used to their best [4]. It guarantees that none of one server or node becomes a bottleneck so that Cloud-IoT applications run effectively and without problems. Effective load balancing guarantees scalability and endurance of cloud-based IoT systems as well as improves Quality of Service (QoS) [5]. This is a must-have since growing demand for consistent IoT solutions spans many different fields. Over years, many load-balancing techniques have been developed to address these problems. These algorithms employ task scheduling, resource allocation, and dynamic load distribution [6] among other tactics to meet their performance criteria. Furthermore, changes in optimization algorithms and artificial intelligence (AI) have opened extra opportunities for creating more sophisticated and flexible load balancing techniques. These advancements aim to keep up with the growing complexity and size of IoT-cloud systems thereby ensuring their usefulness in practical applications.

Moreover, edge computing as well as fog computing have become indispensable additional tools to overcome the limits of centralized cloud-IoT systems. An all-encompassing architecture, fog computing assigns resources to smart devices sequentially via the cloud. It not only stretches the cloud but also actively incorporates IoT-using networks [7]. Figures 1 shows the load balancing process.

The parts that follow address algorithms, metrics influencing load balancing, challenges in load balancing implementation in Cloud-IoT environments. Understanding the function of load balancing in distributed systems depends on some basic ideas that maximize performance, scalability, and effective use of resources.

.

1.1 Factors Influencing Load Balancing

Many significant steps are taken to evaluate load-balancing systems' general performance and effectiveness [8].

- Response Time (RT)- It is the period of time needed for a request to advance from submission to system first response. In time-sensitive applications, such IoT-dependent healthcare systems, a reduced response time usually denotes better efficiency.
- Throughput (TP)- Measuring the overall number of chores completed during a given period of time yields. Higher throughput assures that resources are being

used more wisely and shows that the system can properly handle tasks.

- Scalability- It is also a crucial factor since it shows that the system can control unanticipated workload variations without performance deterioration. With a scalable load balancing method, which can adapt to various needs, one can maintain constant performance in always changing surroundings.
- Resource Utilization (RU)- It is important since it gauges the efficiency of memory, storage, processing capability, and processing power consumption. Effective resource use is essential for both enhancing system performance and lowering energy consumption in environments sensitive to energy use, such IoT-cloud ecosystems.
- Makespan (MS)- Another important consideration is the general time required to finish any chore. Munising makespan is essential for applications with limited deadlines since it accelerates the execution of tasks.
- Associated Overhead (AO)- It considers additional costs including task migration, algorithm running time, and communication delays, all of which must be avoided if the goal is to improve the operational effectiveness of the system. Using techniques that cut computational overhead without compromising performance is more long-term sustainable and efficient.
- Energy Efficiency (EE) - Environmentally friendly computing is context, EE, is becoming ever more crucial. Effective load balancing systems aim to maximize energy consumption without compromising system performance, therefore enabling the sustainability of big-scale cloud environments.
- Service Level Agreement (SLA)- It gauges the system's conformity to accepted criteria of service quality including availability, response time, and prioritizing. Minimizing SLA deviations helps to maintain users' trust and satisfy contractual obligations.
- Processing Cost– An important component is whether load balancing techniques are practical. Included in this group are running costs like extra software overhead, device upkeep, and energy consumption. Effective algorithms seek to increase job effectiveness while avoiding resource waste, therefore helping to control expenses. Thanks to cost-effective load balancing, which also ensures system durability, cloud-based IoT solutions are becoming accessible and reasonably priced for businesses. .

These criteria taken together provide a thorough framework for evaluating load-balancing methods' handling of problems including varying workloads, various resources, and economy of cost-effectiveness.

1.2 Challenges in Load Balancing for IoT-Cloud Environments

The features of IoT networks cause several challenges that load balancing systems must overcome. Creating effective solutions calls for a complete awareness of these issues [9].

- The performance of cloud systems is greatly affected by workload patterns, which are defined by changing needs, unpredictable traffic behavior, and various applications. To manage these differences and ensure smooth operations, effective techniques are required.
- The geographical dispersion of data centers in the cloud—typically found in far-off locations—cause transmission delays. Technologies like fog computing and edge computing are absolutely essential to address this since they help to lower latency by evaluating data closer to its source. Although fog and edge devices have few resources, it can be difficult to effectively control them.
- Load balancing techniques are much shaped by financial and cost factors. All methods have as their shared objectives reducing running expenses and optimizing the use of the resources at hand by removing pointless idle.
- Given the often-changing character of applications and the consequent need for continuous monitoring, cloud services must also be elastic and scalable. Maintaining proper load distribution can be challenging depending on the level of monitoring.
- Compliance with SLAs is an extra vital component since breached SLAs undermine the quality of cloud service providers. It is imperative to strike a mix between throughput, energy consumption, makespan, cost, and service quality.
- While in some situations Virtual Machine (VM) migrations might maximize resources, if done too often they could compromise service quality. The temporal complexity of frequent virtual machine migrations rises due to the enormous volume of work involved in data migration including copy pages of memory to the destination system.
- Load balancing's success mostly depends on the resources' availability. The existence of limited resources-induced bottlenecks can compromise the system's efficiency in managing tasks. Data centers also much worry about their energy use. Effective load balancing can lower power consumption by transferring VMs from hosts with high workloads to those with lesser loads.

1.3 Algorithms used in Load Balancing

In order to achieve efficient workload distribution in cloud and IoT-based systems, algorithms with a variety of approaches for workload allocation, resource optimization, and system performance improvement have been developed. Generic categorization into static, dynamic, and hybrid approaches is possible because to the fundamental ideas of these algorithms. The algorithms utilized for load balancing are displayed in Figure 2.

A. Static Load Balancing Algorithms

Since they distribute work according to pre-defined criteria [8], static algorithms are faster than dynamic ones and easier to implement. Among such methods are Weighted Round Robin (WRR) [11] and Round Robin (RR) [10].

B. Dynamic Load Balancing Algorithms

Dynamic algorithms modify the allocation of tasks in real-time based on the current condition of the system. They perform more effectively when faced with ambiguous obligations. These techniques include Least Connection (LC) [12], Equally Spread Current Execution (ESCE) [13] and Biased Random Sampling (BRS) [14].

C. Hybrid Load Balancing Algorithms

Hybrid algorithms leverage the optimal characteristics of both dynamic and static methodologies through their integration. One technique is PSO-DA, which denotes Particle Swarm Optimization combined with a Dragonfly Algorithm [15].

D. Artificial Intelligence and Machine Learning Algorithms

Combining artificial intelligence and machine learning is helping new approaches enhance load balancing. Two such are algorithms grounded on fuzzy logic [17] and Reinforcement Learning (RL) [16].

E. Metaheuristic Algorithms

A main reason metaheuristic algorithms are becoming more and more popular is their ability to find almost ideal answers to challenging issues. Such algorithms derive from biological processes previously reported in the physical sciences based on nature [17].

.

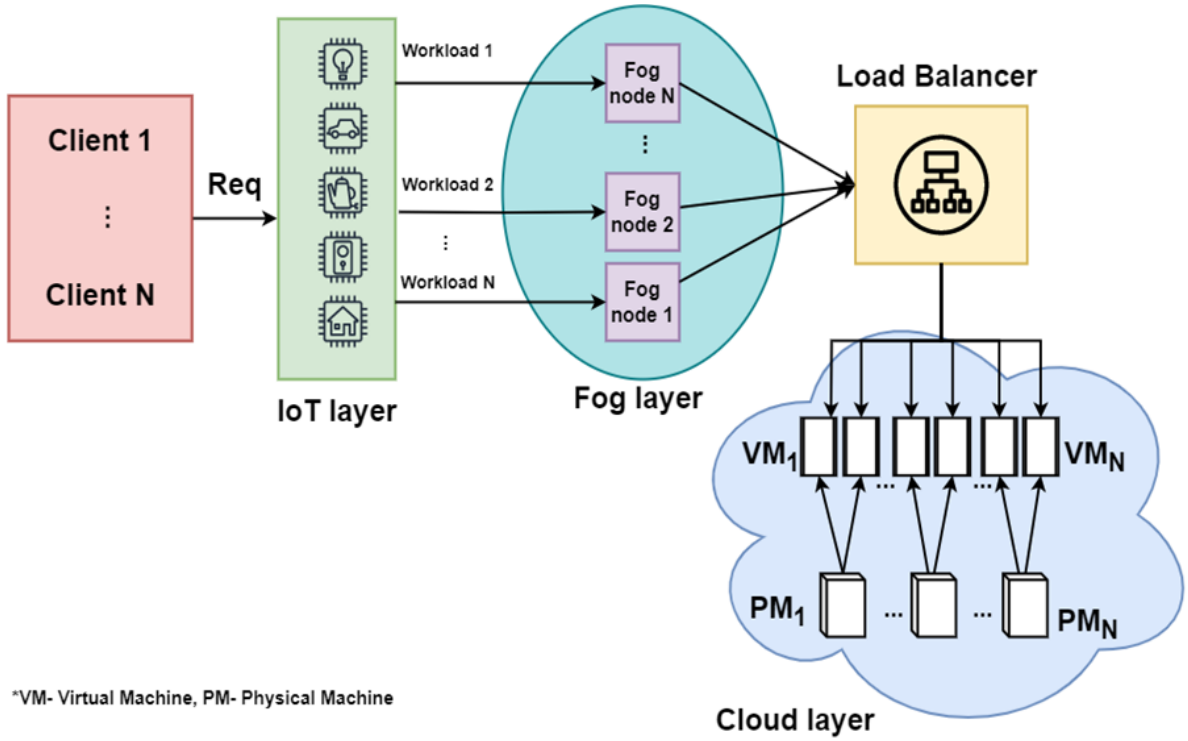

Figure 1. Structure of load balancing

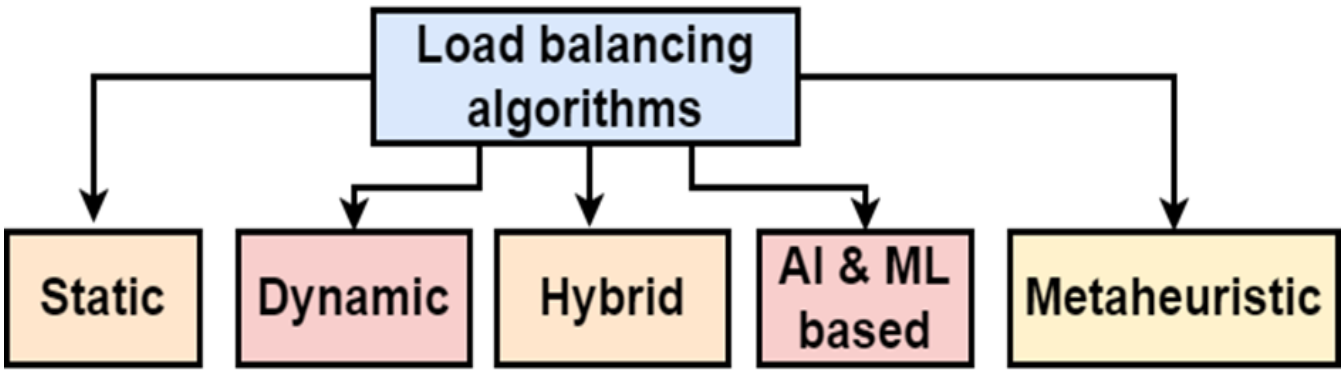

Figure 2. Algorithms used in load balancing

This include Ant Colony Optimization (ACO) [18], Genetic Algorithm (GA) [19], Simulated Annealing (SA) [20], Bat Algorithm (BA) [21], Firefly Algorithm (FA) [22], Cuckoo Search Algorithm (CSA) [23], Artificial Bee Colony (ABC) [24], Dragonfly Algorithm (DA) [25], Crow Search Algorithm (CSA) [26] and various hybrid algorithms such as HEHO-HAS [27].Because of its own respective benefits and drawbacks, every one of these algorithms is appropriate for a certain job. Through load-balancing solutions tailored for cloud and IoT systems, these algorithms address scalability, latency, energy efficiency, and cost-effectiveness. This survey aims to provide a summary of the subject and focuses on modern load balancing algorithms created for IoT installations in the cloud. By means of classification and evaluation of current approaches, this study reveals their advantages and drawbacks, therefore shedding light on the present situation of the subject by class.

The following is the configuration of the surviving elements: Emphasizing the techniques and applications of load balancing algorithms created for IoT cloud systems, Section II provides a literature review on them. In Section III, it contrasts and contrasts different survey forms highlighting both their advantages and drawbacks as well as their similarities. By analyzing significant performance indicators and their impact on general system efficiency, Section IV evaluates the effectiveness of the approaches. Section V at the conclusion of the survey offers a synopsis of the key ideas and recommendations for future research directions in this discipline.

II. SURVEY ON LOAD BALANCING ALGORITHMS FOR CLOUD BASED IOT ENVIRONMENTS

Xingjun et al. [28] showed using a Grey Wolf Optimization (GWO) technique to disperse the burden in cloud-based IoT environments, hence lowering reaction time and improving system performance. Inspired by the social structure and hunting behavior of grey wolves, the approach follows how virtual machines (prey) are given tasks (wolves) depending on their load condition. Using fuzzy logic, which assigns linguistic values—such as high, low, or moderate—to activities depending on CPU performance and virtual machine load guarantees efficient task allocation. By matching resource-constrained virtual machines with non-used workloads, the method maximizes load distribution. Considering both response time and load imbalance, the fitness function of the algorithm aims to lower response time while yet attaining a balanced load among VM.

Zhang et al. [29] introduced a multiuser, execute many jobs, multitier Mobile-Edge Cloud Computing (MEC) system design, so optimizing processing time and energy utilization. Computation responsibilities are dynamically dispersed in this three-tier architecture among Mobile Devices (MDUs), edge servers or tiny Base Stations (BSs), and a central cloud server. The system integrates an orthogonal frequency-based communication model and a computation-based model to enable local or offload of activities and lowers interference. Load balancing among BSs is accomplished by use of a central control manager to allocate MDUs according to resource usage and computational demands. The concept addresses security issues by including an AES encryption method augmented with cryptographic keys based on ECG data. This method preserves important data during offloading. Recasting the optimization problem as a binary linear one allows them to use a secure Load Balancing and Computational Offloading (LBCO) technique ensuring efficient offloading options with minimum complexity.

Agarwal et al. [30] constructed a Load Balancing-Assisted Access Control Mechanism (LB-ACM) inside a multilayer Edge-Fog-Cloud network design in order to provide load balancing and safe access control in IoT environments. This architecture addresses excessive latency, network congestion, and unlawful access among other problems. The design consists of three layers: the IoT sensors, the edge, and the cloud. At the edge layer, abnormalities from IoT sensor generated data are pre-processed. Dynamic load balancing and periodic network architecture refreshment by the fog layer helps before filtered data is sent to the cloud. Comprising VMs, firewalls, and Access Control Lists (ACLs), the cloud layer ensures safe access and effective resource use. Regarding the first load processing and distribution, the LB-ACM's fog and edge layers take front stage. Regarding the cloud layer, it keeps unwanted people out and manages traffic effectively using a three-tiered access control system comprising virtual machine-level authentication, ACLs, and firewalls.

Nezami et al. [31] developed a distributed multi-agent system known Electronic Point of Sale Fog (EPOS Fog) to effectively position IoT services over the edge-to---cloud continuum. This approach balances loads and slow reaction times in IoT environments. EPOS Fog seeks to Minimize the Variance in usage across network nodes (MIN-VAR) and the Cost of service execution (MIN-COST) with its two-objective optimization method. The technology lets local agents design service placement strategies based on workload criteria and proximity of resources. Using the I-EPOS method for group decision-making, they might also cooperate to maximize worldwide placement.

Abdulhammed et al. [32] suggested a two-stage technique for healthcare systems based on the IoT using the Sparrow Search Algorithm (SSA), therefore solving the load balancing issue in cloud computing. This approach moves healthcare from standard in-hospital techniques toward real-time remote monitoring by using IoT wearable devices linked via wireless networks. Between sensors and task queues, an IoT gateway serves as a link; these devices collect and forward patient data—including blood pressure and temperature—to it. The system manages the enormous volumes of data using cloud computing, therefore relieving some of the demand on the infrastructure of the Internet. The system depends critically on task queues, VMs, cloud brokers, and VM managers. Maintaining focused attention on tasks, maintaining sufficient resources, and liaising with the cloud broker—who assigns tasks to VM—using the SSA—are responsibilities of the VM manager. By weighing their fitness value—derived from metrics including execution speed, storage use, and CPU use—the SSA decides how best to divide work among VMs.

Li et al. [33] developed a load-balanced data-layered transmission strategy inside a cooperative Cloud-Edge-End (CEE) IoT architecture in order to address the several and diverse data processing requirements of IoT applications. The design divides the edge layer into two sections: the intelligent edge device level and the edge infrastructure layer, therefore increasing data routing efficiency. By assigning jobs to the suitable levels depending on data type, flow size, and available resources, the classification of data transmission into real-time, near-real-time, and non-real-time guarantees that heterogeneous devices may communicate effortlessly. A central control module dynamically controls resource allocation and hierarchical data routing to maximize system performance, hence perhaps using Software-Defined Networking (SDN). Furthermore balanced in this module are packet delivery rates, energy usage, and latency.

Aqeel et al. [34] used the Chaos-based Horse Ride Optimization Algorithm (CHROA) method to develop a load-balancing and energy-saving strategy for IoE systems housed in the cloud. IoT networks as well as cloud computing are included into the system design. Under this system, data is transferred from cluster members to the cloud via cluster heads while clusters are generated using CHROA. Using the notion of chaos and hierarchical behavior in horse herds, the CHROA algorithm balances the exploration and exploitation phase so enhancing global optimization. Apache Flume and Apache Spark respectively handle real-time data input and processing while storage is under control by the Hadoop Distributed File System (HDFS).

Using a Deep Load Balancer (DLB) to control the enormous volumes of data generated by IoT devices in the cloud, the authors Devi et al. [35] proposed a method for IoT

data storage in the cloud. By combining a real-time scheduling approach and deep learning techniques such as Restricted Boltzmann Machines (RBM) and Deep Belief Networks (DBN), the model minimizes latency and maximizes load balancing. The three key components of the DLB approach are Dynamic Scheduler (DS), which chooses which servers are most suitable to perform the task; Deep Classifier (DC), which decides whether cloud servers are fit for the task; Loud Resource Monitor (CRM), which preserves cloud resources. The DLB approach works well since it maximizes and normalizes resource characteristics, therefore improving resource allocation and reducing latency. The model also underlines the need of evenly spreading workload among VMs in the cloud in order to maximize general load distribution, Energy Consumption (EC), and task execution time (ET).

Shamsa et al. [36] developed the Multiple Workflow Scheduling with a Load-Balancing Approach and Dynamic Resources Allocation (MWL-DRA) paradigm to control several workflow scheduling in hybrid cloud, fog, and IoT settings. It consists of layers one through four: Workflow Generator Layer (WGL), Fog Computing Layer (FCL), Resource Analyzer Layer (RAL) and Cloud Computing Layer (CCL). By means of fog nodes, the intermediary level with moderate processing capacity bridges the WGL with the IoT devices producing particular resource requirements. Part of the CCL are powerful computers situated in data centers capable of handling large processing and vast data storage. At the core of this design, the RAL manages the cellular, regional, and system levels of load balancing; it is in charge of analyzing and assessing events from the other layers, therefore generating workload projections, and allocating resources. Using Kruchten's 4+1 model—which incorporates a scenario view for requirement identification in addition to process, logical, physical, and developmental viewpoints—the framework is further detailed Essential components of RAL designed to dynamically allocate resources and maximize workflow scheduling are the Data Logger, Models Trainer, Status Predictor, LBP Maker, and Load Balancer. Including ATAM-based evaluation into the design helps one assess quality characteristics including performance, scalability, and dependability.

Vijarania et al. [37] proposed a combined fog-computing model for efficient energy management and load balancing in an IoT-fog-cloud environment to address problems including increasing congestion in fog devices, decreasing performance of fog gateways, and delayed transmission links generated by the fast expansion of IoT devices. By giving fog nodes top precedence over cloud nodes, the three- stage architecture of the model allowed for the optimization of resource usage, decrease of latency, and energy economy.

Yakubu and Murali [38] presented the Modified Harris Hawks Optimization (MHHO) method and the layer fit strategy to control resources and distributing tasks in IoT-Fog-Cloud environments. The Layer Fit Method guarantees effective distribution of the fog as well as cloud layer with workloads by ranking jobs according to delays like processing and transmission time. Processing work with a higher priority and forwarding lower priority activities to the cloud helps one avoid the resources of the fog from becoming overwhelmed. The MHHO algorithm enhances the fundamental HHO by using a load-balancing mechanism for equitable distribution of resources and including an enhanced energy update strategy to avoid local optima. Resource administrators monitor the active capacity of every level in the model architecture—edge devices, BSs, fog devices, and cloud resources.

Ala'anzy et al. [39] created the Optimized Load Balancing (OLB) method to enhance network performance in healthcare systems depending on the IoT. IoT devices track patients' vitals in real time; fog nodes process data in real time near BSs; the cloud layer keeps data for use later on. OLB employs an array-based strategy to efficiently manage and update traffic and computational loads, therefore significantly lowering superfluous computations and increasing scalability.

Using a fuzzy logic approach, Mahapatra et al. [40] developed a three-tiered IoT, fog, and cloud architecture to maximize the load balancing and job scheduling in computer systems. User request generation from scattered smart devices comes from the IoT layer; intermediary networking nodes and clusters of Fog nodes supplied by a Fog Controller (FC) come from the Fog layer; and resource-intensive job hosting in data centers comes from the Cloud layer. The system sorts activities into high, medium, and low priority using fuzzy logic after weighing elements such task length, start time, and delay limits. While high-priority tasks are passed over effective nodes in the fog or the cloud to lower delay, low-priority tasks are managed locally. Apart from a Binary Linear-Weight JAYA (BLWJAYA) algorithm for effective task scheduling, a compatibility-based work offloading technique employing cosine similarity for fair load allocation between fog nodes was also developed.

Shamsa et al. [41] were able to maximize resource management in IoT-cloud-fog computing systems and prevent either over- or underuse of the resources by tackling the critical problem of allocating resource workloads equitably. A Load Balancing Plan (LBP) looking at resources to maximize usage among Data Centers (DCs), Fog Points (FPs), and IoT nodes achieves this. Thirdly, the model consists of the System Management Component (SMC), the Region Management Component (RMC), and the Cell Management Component (CMC), in that sequence. The SMC divides the system into grid cells of uniform size, generates a Cellular Coordinates Map (CCM) for effective administration, and assigns resources to their matching cells. The RMC creates Long Short-Term Memory (LSTM) algorithms to estimate workload patterns and generates the LBP so that tasks from Overloaded Cells (OLCs) may be transferred to Underloaded Cells (ULCs). It then examines system events. To keep everything in balance, the CMC last but not least handles local scheduling, process distribution, and cell-level load balancing.

With an eye toward enhancing task offloading between devices, fog, and cloud servers, Tishin et al. [42] proposed ML for use in load balancing in IoT systems. It shows how supervised and reinforcement learning may be employed as well as how Natural Language Processing (NLP) and Large Language Models (LLMs) may be used to forecast runtime complexity like Big(O) using examples of network traffic analysis and job scheduling. By matching tasks to suitable devices, assessing device capabilities, and hence ensuring effective resource utilization and task allocation, the LLM model guarantees.

Moparthi et al. [43] introduced an energy-efficient load-balancing system to address problems including resource restrictions, changing workloads, and energy usage in IoT-based cloud environments. IoT sensors gather varied data, process it using edge-computing technologies, and then forward some data to the cloud for analysis. One of its main characteristics is the Energy Sensitive Balancing Load (ESBL) algorithm, which considers metrics like CPU use, power consumption, memory usage, and SLA violations to optimize workload distribution. IoT sensors. The ESBL algorithm assures effective job allocation across VM,

therefore lowering operational expenses and energy consumption and increasing resource economy and scalability.

III. COMPARATIVE ANALYSIS

The evaluation of the aforementioned literature comparison is presented in this section.

Table I. Comparative analysis of different load-balancing models

Ref No.	Techniques	Merits	Demerits	Environment used	Performance Metrics
[28]	GWO	The model improves system performance, balances load, and reduces reaction time, all of which contribute to an improved user experience and meetability of service level agreements.	Because of the model's iterative optimization process, it may induce higher energy consumption.	CloudSim	Degree of load imbalance = 0.95/ 1000 tasks, Runtime = 96.5/ 400 tasks
[29]	MEC, Secure LBCO, AES	By reducing data transmission and optimizing resource allocation, integrating load balancing with security lowers communication costs.	While offloading, the model works with static MDUs, but performance could suffer if MDUs are constantly switching across sBSs.	MATLAB	System cost (energy and time to execute tasks) = $11x10^6$ / 10 tasks
[30]	LB-ACM	The approach enhances system efficiency by minimizing needless strain on resources and optimizing CPU consumption in the cloud network.	Packet loss can happen in surroundings that are constantly changing.	Amazon Web Service (AWS)	CPU utilization = 26%, Average Network packets in = 6845 count, Average Network Packets Out = 6472 count
[31]	EPOS Fog	As the number of agents increases within the network's fixed capacity, the model's performance remains good, demonstrating its ability to scale in dispersed contexts.	The most error-prone cases involve beta service distribution, which shows that workload allocation might impact the model's performance.	Google cluster trace	For BA topology: Load balance improvement = 0.13, Normalized error =0.9, Utilization variance = 0.17. For WS topology: Load balance improvement = 0.15, Normalized error =0.88, Utilization variance = 0.73. For ER topology: Load balance improvement = 0.172, Normalized error =0.73, Utilization variance = 0.198.
[32]	SSA	The possibility of overloading is decreased because the model makes optimal use of the VMs.	In extremely changing contexts, when job durations and virtual machine capabilities fluctuate erratically, the system's adaptability may be compromised.	CloudSim	For 500 task, Degree of imbalance = 177.7675, processing time=899.8979 ms, Makespan time =23.05 ms,
[33]	CEE	Reducing computing complexity, improving QoS, and supporting varied IoT scenarios are all achieved using this method.	Due to its absence of CEE IoT layer-specific data forwarding models and its use of a random method to pick target nodes, the system produces outcomes that are less than objective.	MATLAB R2020b	For 15 edge infrastructure, Cumulative delivery rate = 0.878/50 MB, System throughput = 579 Mbps/ 20000 packets, (For medium load) Average delay = 1 ms/ 320 kb
[34]	CHROA, HDFS	The suggested architecture provides real-time, low-latency services, allowing for efficient machine-to-machine interactions; this is particularly true in healthcare applications.	Particularly in large-scale deployments, the suggested methodologies' practical consequences and real-world applications are fully unaddressed.	MATLAB	Average throughput = 70.122 kbps, End-to-End (ETE) delay = 0.0643 s, Normalized overhead = 0.4069, Network lifetime = 510.256 s, Total energy consumption = 5.1289 J
[35]	DLB, CRM, RBM, DBN, DC, DS	By assessing available cloud resources before to assigning tasks, the approach ensures secure task allocation.	There has been no investigation into how external variables, such as network latency, could impact performance.	CloudSim	For 50 VMs, Response time = 0.36 ms, makespan = 0.709 ms, Overhead = 0.75, migration time = 0.09 ms
[36]	MWL-DRA	Job scheduling is very successful with few unscheduled jobs, which is a sign of good task management. .	In cases when there is a sudden increase in workload or when IoT or fog nodes fail, the approach might not be able to cope well.	MATLAB	Average time steps = 1210.5, standard deviation = 15.45, Max time steps = 190, Minimum time steps = 81, Skewness = 0.11, kurtosis = 0.522
[37]	Fog computing	By dividing up work across the cloud and fog layers, the technique decreases computational and transmission delays, leading to more efficient operations and quicker reaction times.	Under conditions of very high data flow, the algorithm's performance may suffer.	iFogSim	Network lifetime = 479 s/ 350 load, Energy consumption = 0.85 J/ 500 nodes, Average delay = 36 ms/ 700 nodes, response time = 98 ms/ 400 nodes
[38]	MHHO, Layer fit algorithm	In response to changes in the system's workload or the quantity of virtual machines, the MHHO paradigm performs admirably.	The model continues to encounter difficulties when dealing with higher quantities of jobs or virtual machines, which can lead to significant	iFogSim	For 100 tasks and 20 VMs, Execution cost = 17 x 106, Average makespan time = 5s, Energy consumption = 71234

			increases in energy consumption.		
[39]	OLB	OLB's real-time adaptability makes it particularly well-suited for time-sensitive applications	Improper allocation might still occur under certain conditions, which can reduce overall system efficiency.	iFogSim	For 4 cases, Latency = 7.11 ms, Execution time= 7525 ms, Cost of execution = 91609.27, Network usage = 13392 bytes, Energy consumption = 934162.95 kWh
[40]	Fuzzy logic, BLWJAYA	The use of fuzzy logic to classify tasks ensures that tasks are offloaded to the most suitable compute node	If the fuzzy logic rules are not finely tuned, it could lead to misclassification, resulting in suboptimal task offloading	iFogSim	For consistent and high task_machine heterogeneity: Service rate =1.12 E + 05, Resource utilization = 96.29, Latency = 164.73, Energy consumption= 262.82, Load balancing rate = 261.83
[41]	LBP	The model minimizes workflow rejections, fostering a more efficient and scalable resource management framework.	The effectiveness of the model relies on the accurate prediction of workflow loads and available resources, which could be challenging in environments with highly variable workloads.	Velociraptor	Average variance = 21468.28, Workflow completion rate =91.7%, Average resource utilization = 67.9%
[42]	LLM	Using Docker containers to replicate IoT devices with different processing capacity allows one to test several device configurations without using real hardware.	Notable issues such as uneven job distribution, inaccurate Big(O) forecasts, and energy limits are recognized.	Google Cloud instance	O (n^2): execution time = 5ms
[43]	ESBL	The platform easily supports scalability as well as adaptation for various kinds of IoT tasks.	Using cloud resources, particularly at scale, may still incur expenses, even while the framework strives to optimize resource utilization.	-	Energy consumption = 36.81 kWh, Number of node shutdowns = 795, Execution time = 0.00198 s

Table 1 lists all the models together together with their advantages, disadvantages, and performance records. Two fundamental performance criteria—energy consumption and execution time—are applied to evaluate several approaches. To assess the energy consumption and time needed for execution efficiency of several models, it investigated six different approaches ESBL [43], Fuzzy Logic [40], OLB [39], MHHO [38], Fog [37], and CHROA [34]. The impossibility to create direct comparisons results from the fact that every model presents various numbers for evaluating energy. In order to address this challenge, they thus substitute textual analysis for visual representation. Table 1 of the comparison analysis demonstrates that ESBL can easily manage several kinds of IoT tasks.

OLb consumes 934,162.95 kWh for 64 IoT and 4 fog nodes, far more than ESBL's 36.81 kWh per 100 VMs. Though all of the other models use less energy overall, the ESBL model remains the most efficient when it comes to scalable activities using standard VM setups. When it comes to execution speed, the OLB model takes 7, 525 ms for 64 IoT and 4 fog nodes; ESBL's 1.56 ms per 100 VMs is lightning quick and clearly shows advantage here. GWO executes in 96.5 ms, far less than the ESBL's time. Figure 3 shows the times needed for the execution of many models.

As OLB differs greatly from the other two models, it is not displayed in the graphic to help to clarify the distinction. Eliminating OLB reveals the relative execution speeds of ESBL and GWO, therefore stressing the significant difference in execution time between these two models and clarifying their performance. Looking at these criteria taken together, ESBL is obviously the best and most balanced choice. Its exceptional performance in these domains qualifies it for applications requiring quick processing and low power consumption. Table 2 offers a succinct summary of the literature review grounded in the chosen metrics. Most of the publications concentrate on time and energy since their main measurements are these.

Table II. Metrics used in the related papers

Metrics / Methods	Energy Consumption	Execution Time	Processing Cost	Latency	Makespan time	Resource Utilization	Throughput	Response time	Over head
GWO	×	✓	×	×	×	×	×	✓	×
Secure LBCO	×	×	✓	×	×	×	×	×	×
LB-ACM	×	×	×	×	×	✓	×	×	×
EPOS Fog	×	×	×	✓	×	×	×	×	×
SSA	×	✓	×	×	×	×	×	×	×
CEE	×	×	×	✓	×	×	✓	×	×
CHROA	✓	×	×	✓	×	×	✓	×	✓
DLB	×	×	×	×	✓	×	×	✓	✓
MWL-DRA	×	×	×	×	×	×	×	×	×
Fog	✓	×	×	✓	×	×	×	✓	×
MHHO	✓	×	✓	×	✓	×	×	×	×
OLB	✓	✓	✓	✓	×	×	×	×	×
Fuzzy Logic	✓	×	×	✓	×	✓	×	×	×
LBP	×	×	×	×	×	✓	×	×	×
LLM	×	✓	×	×	×	×	×	×	×
ESBL	✓	✓	×	×	×	×	×	×	×

IV. CONCLUSION

Effective load balancing is mostly responsible for improving the energy economy, scalability, and performance of cloud-based IoT systems. They have examined several load balancing techniques in this survey, each having advantages depending on the particulars of an IoT context such as resource availability, network condition, and real-time processing need. Although more recent systems using modern technologies handled the dynamic and heterogeneous character of IoT tasks considerably better, previous methods functioned effectively in centralized settings. Invest in creating innovative algorithms investigating additional speed, energy economy, and scalability far into the future.

V. REFERENCES

[1] G. Lampropoulos, K. Siakas,and T. Anastasiadis, "Internet of things in the context of industry 4.0: An overview," International Journal of Entrepreneurial Knowledge, vol.7, no. 1, 2019.

[2] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," Future generation computer systems, vol. 56, pp.684-700, 2016.

[3] P.P. Ray, "A survey of IoT cloud platforms," Future Computing and Informatics Journal, vol.1, pp.35-46, 2016.

[4] A. Thakur, and M.S. Goraya, " A taxonomic survey on load balancing in cloud," Journal of Network and Computer Applications, Vol. 98, pp. 43-57, 2017.

[5] M. Rostami, and S. Goli-Bidgoli, "An overview of QoS-aware load balancing techniques in SDN-based IoT networks," Journal of Cloud Computing, vol.13, no.1, p.89, 2024.

[6] D. Alsadie, "A Comprehensive Review of AI Techniques for Resource Management in Fog Computing: Trends, Challenges and Future Directions," IEEE Access, 2024.

[7] N.A. Angel, D. Ravindran, P.D.R. Vincent, K. Srinivasan, and Y.C. Hu, "Recent advances in evolving computing paradigms: Cloud, edge, and fog technologies," Sensors, vol. 22, no.1, p.196, 2021.

[8] 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," IEEE Access, vol.11, pp.134517-134530, 2023.

[9] N. Devi, S. Dalal, K. Solanki, S. Dalal, U.K. Lilhore, S. Simaiya, and N. Nuristani, "A systematic literature review for load balancing and task scheduling techniques in cloud computing," Artificial Intelligence Review, vol.57, no.10, p.276, 2024.

[10] S. Mohapatra, S. Mohanty, and K.S. Rekha, "Analysis of different variants in round robin algorithms for load balancing in cloud computing," International Journal of Computer Applications, vol. 69, no.22, pp.17-21, 2013.

[11] M. Kushwaha, B.L. Raina, and S.N. Singh, "Advanced weighted round robin procedure for load balancing in cloud computing environment," In 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence) pp. 215-219. IEEE, 2021.

[12] L. Zhu, J. Cui, and G. Xiong, "Improved dynamic load balancing algorithm based on Least-Connection Scheduling," In 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC) , pp. 1858-1862, IEEE, 2018.

[13] M.A. Alamin, M. K. Elbashir, and A.A. Osman, "A load balancing algorithm to enhance the response time in cloud computing," Red Sea University Journal of Basic and Applied Science, vol. 2, no.2, pp.473-490, 2017.

[14] O.A. Rahmeh, P. Johnson, and A. Taleb-Bendiab, "A dynamic biased random sampling scheme for scalable and reliable grid networks," INFOCOMP journal of computer science, vol. 7, no.4, pp.1-10, 2008.

[15] A. Thakur, and M.S. Goraya, "RAFL: A hybrid metaheuristic based resource allocation framework for load balancing in cloud computing environment," Simulation Modelling Practice and Theory, vol.116, p.102485, 2022.

[16] F.M. Talaat, M.S. Saraya, A.I. Saleh, H.A. Ali, and S.H. Ali, "A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment," Journal of Ambient Intelligence and Humanized Computing, vol. 11, no.11, pp. 4951-4966, 2020.

[17] S. Sethi, A. Sahu, and S.K. Jena, "Efficient load balancing in cloud computing using fuzzy logic," IOSR Journal of Engineering, vol.2, no.7, pp.65-71, 2012.

[18] K. Nishant, P. Sharma, V. Krishna, C. Gupta, K.P. Singh, and R. Rastogi, "Load balancing of nodes in cloud using ant colony optimization," In 2012 UKSim 14th international conference on computer modelling and simulation , pp. 3-8, 2012.

[19] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A genetic algorithm (ga) based load balancing strategy for cloud computing," Procedia Technology, vol.10, pp.340-347, 2013.

[20] B. Mondal, and A. Choudhury, "Simulated annealing (SA) based load balancing strategy for cloud computing," International Journal of Computer Science and Information Technologies, vol. 6, no.4, pp.3307-3312, 2015.

[21] S. Sharma, A.K. Luhach, and S.A.Sinha, "An optimal load balancing technique for cloud computing environment using bat algorithm," Indian J Sci Technol, vol. 9, no.28, pp.1-4, 2016.

[22] A.P. Florence, and V. Shanthi, "A load balancing model using firefly algorithm in cloud computing," Journal of Computer Science, vol. 10, no.7, p.1156, 2014.

[23] M. Yakhchi, S.M. Ghafari, S.Yakhchi, M. Fazeli, and A. Patooghy, "Proposing a load balancing method based on Cuckoo Optimization Algorithm for energy management in cloud computing infrastructures," In 2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO) pp. 1-5, 2015.

[24] A. Ullah, N.M. Nawi, J. Uddin, S. Baseer, and A.H. Rashed, "Artificial bee colony algorithm used for load balancing in cloud computing," IAES International Journal of Artificial Intelligence, vol.8, no. 2, p.156, 2019.

[25] P. Neelima, and A.R.M. Reddy, "An efficient load balancing system using adaptive dragonfly algorithm in cloud computing," Cluster Computing, vol. 23, no.4, pp.2891-2899, 2020.

[26] H. Singh, S. Tyagi, and P. Kumar, "Cloud resource mapping through crow search inspired metaheuristic load balancing technique," Computers & Electrical Engineering, vol.93, p.107221, 2021.

[27] S.M. Ali, N. Kumaran, and G.N. Balaji, "A hybrid elephant herding optimization and harmony search

algorithm for potential load balancing in cloud environments," International Journal of Modeling, Simulation, and Scientific Computing, vol. 13, no. 05, p.2250042, 2022.

[28] L. Xingjun, S. Zhiwei, C. Hongping, and B.O. Mohammed, "A new fuzzy-based method for load balancing in the cloud-based Internet of things using a grey wolf optimization algorithm," International Journal of Communication Systems, vol. 33, no. 8, p.e4370, 2020.

[29] W.Z. Zhang, I.A. Elgendy, M. Hammad, A.M. Iliyasu, X. Du, M. Guizani, and A.A. Abd El-Latif, "Secure and optimized load balancing for multitier IoT and edge-cloud computing systems," IEEE Internet of Things Journal, vol.8, no.10, pp.8119-8132, 2020.

[30] N. Agrawal, "Dynamic load balancing assisted optimized access control mechanism for edge-fog-cloud network in Internet of Things environment," Concurrency and Computation: Practice and Experience, vol.33, no.21, p.e6440, 2021.

[31] Z. Nezami, K. Zamanifar, K. Djemame, and E., Pournaras, "Decentralized edge-to-cloud load balancing: Service placement for the Internet of Things," IEEE Access, vol. 9, pp.64983-65000, 2021.

[32] O.Y. Abdulhammed, "Load balancing of IoT tasks in the cloud computing by using sparrow search algorithm," The Journal of Supercomputing, vol.78, no. 3, pp.3266-3287, 2022.

[33] J. Li, X. Li, J. Yuan, and G. Li, "Load Balanced Data Transmission Strategy Based on Cloud–Edge–End Collaboration in the Internet of Things," Sustainability, vol.14, no.15, p.9602, 2022.

[34] I. Aqeel, I.M. Khormi, S.B. Khan, M. Shuaib, A. Almusharraf, S. Alam, and N.A. Alkhaldi, "Load Balancing Using Artificial Intelligence for Cloud-Enabled Internet of Everything in Healthcare Domain," Sensors, vol. 23, no.11, p.5349, 2023.

[35] K.D.S. Devi, D. Sumathi, V. Vignesh, C. Anilkumar, K. Kataraki, and S. Balakrishnan, "CLOUD load balancing for storing the internet of things using deep load balancer with enhanced security," Measurement: Sensors, vol.28, p.100818, 2023.

[36] Z. Shamsa, A. Rezaee, S. Adabi, and A.M. Rahmani, "A decentralized prediction-based workflow load balancing architecture for cloud/fog/IoT environments," Computing, vol.106, no.1, pp.201-239, 2023.

[37] M. Vijarania, S. Gupta, A. Agrawal, M.O. Adigun, S.A. Ajagbe, and J.B. Awotunde, "Energy efficient load-balancing mechanism in integrated IoT–fog–cloud environment," Electronics, vol.*12*, no.11, p.2543, 2023.

[38] I.Z. Yakubu, and M. Murali, "An efficient meta-heuristic resource allocation with load balancing in IoT-Fog-cloud computing environment," Journal of Ambient Intelligence and Humanized Computing, vol.14, no.3, p.2981-2992, 2023.

[39] M.A. Ala'anzy, R. Zhanuzak, R. Akhmedov, N. Mohamed, and J. Al-Jaroodi, " Dynamic Load Balancing for Enhanced Network Performance in IoT-Enabled Smart Healthcare with Fog Computing," IEEE Access, 2024.

[40] A. Mahapatra, S.K. Majhi, K. Mishra, R. Pradhan, D.C. Rao, and S.K. Panda, "An energy-aware task offloading and load balancing for latency-sensitive IoT applications in the Fog-Cloud continuum," IEEE Access, vol.12, pp. 14334 – 14349, 2024.

[41] M. Tishin, C.X. Mavromoustakis, and J.M. Batalla, "Machine Learning methods in tasks load balancing between IoT devices and the Cloud, IEEE Access, vol.12, pp. 133726 – 133733, 2024.

[42] Z. Shamsa, A. Rezaee, S. Adabi, A.M. Rahimabadi, and A.M. Rahmani, "A distributed load balancing method for IoT/Fog/Cloud environments with volatile resource support. Cluster Computing, vol.27, no. 4, pp. 4281 - 4320, 2024.

[43] N.R. Moparthi, G. Balakrishna, P. Chithaluru, M. Kolla, and M. Kumar, "An improved energy-efficient cloud-optimized load-balancing for IoT frameworks. Heliyon, vol. 9, no.11, 2023.

