Swati Jat, 2024, 12:5
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

A Survey on Cooperative Spectrum Sensing Techniques and Requirements

Swati Jat[1], Professor Rani Kushwaha[2], Professor Jayshree Boaddh[3]
M.Tech student, Mittal Instiitute of technology, Bhopal[1]
Ap, Cse Mittal Institute of technology, Bhopal[2]
HOD, Cse Vaishnavi Institute of technology and Science, Bhopal[3]

Abstract- The advent of cloud computing, powered by advancements in virtualization technologies, provides significant opportunities for cost-efficient hosting of virtual resources, eliminating the need for physical infrastructure ownership. Cloud data centers often employ a wide array of heterogeneous commodity servers to support numerous IoT devices with varying specifications and dynamic resource demands. This diversity can lead to imbalanced resource utilization across servers, potentially causing performance degradation and violations of service level agreements (SLAs). Edge computing has emerged as a viable solution to these challenges by redistributing workloads, thereby improving overall system performance. This paper discusses the necessity of edge computing, detailing its architectural framework. Additionally, it explores various load balancing techniques crucial for maintaining balanced resource utilization. The paper also reviews several models proposed by researchers aimed at enhancing edge network performance. Lastly, it outlines key evaluation metrics used to compare different load balancing models.

Keywords- Load Balancing, Task Scheduling, Edge Computing, Cloud Computing.

I. INTRODUCTION

The rapid proliferation of inexpensive and compact network devices connected to the Internet is continuing to grow exponentially. Beyond conventional computing devices, billions of smart 'things' now autonomously interact and communicate, forming a vast network known as the Internet of Things (IoT) [1, 2]. These smart devices, including low-power wireless sensor nodes, typically feature limited processing power and memory capacity. In the IoT landscape, end-users are often unaware of the underlying resources, services, and capabilities, yet they benefit from an environment where intelligent services are seamlessly integrated into daily life, providing conveniences anytime and anywhere.

Within this IoT ecosystem, low-power wireless sensor devices are integral to collecting vital data from the physical environment—such as temperature, pressure, and motion—enabling the delivery of intelligent services. These applications range widely, from environmental monitoring to real-time decision-making for efficient resource management.

Load balancing (LB) is a critical function within IoT networks, responsible for optimally distributing resources to user tasks to maximize resource utilization [3]. The resources in an IoT network include both the hardware capabilities of the nodes (e.g., computational power, storage, and energy) and network resources (e.g., bandwidth, load balancers, and traffic analyzers). Effective load balancing techniques help prevent overload conditions and improve the performance of large-scale IoT networks by optimizing Quality of Service (QoS) parameters such as response time, throughput, and resource utilization [4].

The vast number of events generated by IoT devices can lead to congestion on specific network paths, reducing performance and efficiency. Uneven traffic distribution can cause latency spikes and packet loss, decreasing the Packet Delivery Ratio (PDR). To mitigate such congestion, efficient load balancing strategies are essential [5]. These strategies use local network information, such as network topology, to distribute traffic more evenly across various paths and resources, thereby enhancing overall network performance.

With the increasing adoption of IoT, edge cloud computing has emerged as a complementary paradigm to provide real-time services and improve responsiveness for users. Edge cloud computing extends some cloud services to the network's edge, closer to the end-users, reducing latency and improving user experience [6]. However, due to the limited capabilities and resources of edge devices compared to centralized cloud servers, implementing effective load balancing techniques in edge cloud environments requires specialized approaches to ensure efficient resource use and optimal performance.

The remainder of this paper is structured into four additional sections. The second section provides a detailed overview of edge computing architecture. The third section discusses various load balancing techniques. The fourth section summarizes relevant research works and compares different models. The paper concludes with final thoughts on the subject.

II. EDGE COMPUTING ARCHITECTURE

Edge computing (EC) moves storage and computational capabilities closer to the location where data is produced. This proximity enhances response times by reducing the distance data must travel. Moreover, EC reduces dependence on potentially unreliable internet connections, serving as an alternative to traditional cloud computing (CC) for IoT applications that require high accessibility and quick responses. This is crucial because IoT applications often struggle with inconsistent internet reliability. When internet speeds drop or connections are interrupted, response times lengthen, and application performance deteriorates due to reduced availability. EC helps alleviate this issue by decreasing reliance on the internet. By placing computation and storage resources near the source of data generation, EC improves response times and boosts application performance.

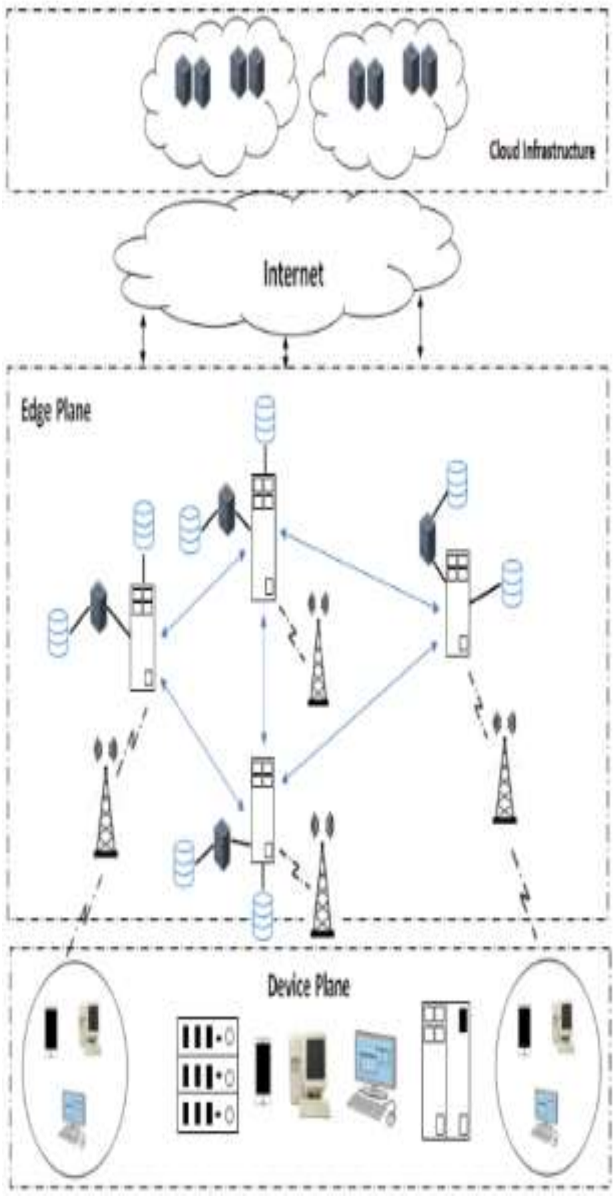

Figure 1. General architecture of edge computing.

Figure 1 depicts the architecture of edge computing (EC), a framework designed to enhance the effectiveness and performance of Internet of Things (IoT) systems. This architecture is composed of three primary layers: the Device layer, the EC layer, and the Cloud infrastructure layer.

Device Layer

This layer includes smart IoT devices like smartphones, tablets, sensors, and actuators. These devices are primarily tasked with collecting and exchanging data from their surroundings. At this level, the focus is on the devices' sensing capabilities, which prioritize data collection over computational power.

EC Layer

The EC layer is made up of distributed edge devices or nodes. These nodes act as intermediaries, bridging the gap between the devices in the Device layer and the Cloud infrastructure. They may function as smart devices themselves or as components that facilitate connectivity, such as gateways and routers. Essentially, the EC layer serves as a communication link, ensuring data is transferred efficiently between the devices and the Cloud infrastructure.

Cloud Infrastructure Layer

This layer is responsible for receiving and storing the data relayed by the EC layer. It contains the cloud resources required for storage and advanced processing tasks that may exceed the capabilities of the local EC layer. Additionally, the Cloud infrastructure layer is vital in managing the allocation of network resources within the EC framework, ensuring optimal resource utilization.

Within the EC layer, edge devices and nodes equipped with computing power handle various computational tasks to reduce dependence on the internet. These tasks include data processing, device monitoring, temporary storage, and decision-making. By performing these tasks closer to where data is generated, EC aims to increase efficiency, shorten response times, and lessen reliance on external internet connections, thereby enhancing the overall performance of IoT systems.

III. TECHNIQUES OF LOAD BALANCING

Numerous techniques have been proposed based on static and dynamic scheduling. Given that most current research and applications are dynamic in nature, this paper focuses on dynamic load balancing algorithms [8].

1. Throttled-Based Algorithm

The Throttled Load Balancer (TLB) maintains a record of the status of each node, stored in a table. When a request arrives, TLB checks this table to find a suitable node based on its availability and capacity. If a suitable match is found, the request is processed; otherwise, a negative response is returned, and the request is queued for later. However, this algorithm is relatively slow, making it less effective for large-scale load balancing. The algorithm prioritizes requests from nodes with lighter loads, compressing these requests to correct imbalances.

2. Divide and Conquer

This technique involves breaking down the load into smaller segments, which are then processed independently. Once processing is complete, the loads are merged in pairs across two nodes, which then carry out load balancing. This iterative process continues until all nodes are balanced. If a large request cannot be processed effectively, it is split into smaller sub-requests for more manageable processing.

3. Pipelining

The Pipelining algorithm works by establishing virtual pipelines where the output of one stage serves as the input for the next. This method enables continuous processing without interruption, allowing multiple nodes' loads to be calculated simultaneously, assuming a pipelined processing approach.

4. Cyclic Algorithm

The Cyclic-based load balancing algorithm modifies the RAND algorithm to function in a cyclic manner. Loads are distributed among nodes in a cyclic fashion, ensuring that a process does not receive further requests for a certain period until the cycle completes. Data is exchanged locally, and if a source node fails, the information is redirected to a designated node.

5. Probabilistic Algorithm
The Probabilistic load balancing algorithm keeps track of each node's status and assigns probabilities accordingly. Nodes with higher probabilities accept loads from nodes with lower probabilities, thus redistributing the loads more evenly. The probability for each node is calculated by determining the ratio of its load to the total load.

6. Prioritized Random Algorithm
In the Prioritized Random load balancing method, nodes are chosen at random, but each node is assigned a priority level. Nodes with higher priorities transfer their loads to those with lower priorities, helping to alleviate stress on overloaded nodes.

IV. RELATED WORK

Hu, P. et al. (2019) conducted a study [9] in which they proposed that while global popularity metrics can offer valuable insights into average content request patterns across a broad user base, they may not fully capture the unique preferences of individual users. The authors argue that achieving a higher click-through rate would be more effectively accomplished by focusing on personalized user preferences rather than relying solely on aggregate popularity data. To improve the accuracy and relevance of load predictions, the study suggests incorporating individualized load predictions for each user, thereby addressing the gap left by global popularity metrics.

In their research, Kong, W. et al. (2020) explored the potential applications of artificial intelligence (AI) within the Internet of Things (IoT), specifically examining edge-based video processing [10]. Unlike previous studies, their approach emphasizes tackling the diverse challenges present across various layers of serverless edge systems, which include hardware, platforms, and software. Their aim is to develop load balancing policies that not only focus on reducing latency but also take into account a range of performance metrics, such as cost, throughput, energy consumption, and the precision of AI inference.

Saba, T. et al. (2021) introduced a secured data management system that incorporates a distributed load balancing protocol using particle swarm optimization in their study [12]. Their objective is to enhance the response times for cloud users while ensuring the security and integrity of network communication. By combining distributed computing methods with optimizations that bring high-cost computations closer to the requesting nodes, their approach aims to reduce latency and transmission overhead. Additionally, their system evaluates trust levels systematically to protect communication channels from potential threats posed by malicious devices.

S. Shao et al. (2022) presented a load balancing algorithm specifically designed for edge computing environments, referred to as LBA-EC, in their research [13]. This algorithm leverages a weighted bipartite graph to optimize the utilization of network edge resources, thereby minimizing user delays and improving service quality. The task scheduling process is divided into two phases: the first phase involves matching tasks to different edge servers, and the second phase allocates tasks to various containers within edge servers, taking into account factors such as energy consumption and task completion time.

Y. Wang et al. (2023) proposed a novel edge-computing load-balancing method tailored for Low Earth Orbit (LEO) satellite networks, focusing on maximizing the flow of virtual links within their study [14]. Their methodology involves identifying minimal rectangles of computing nodes, establishing virtual edge computing links between these nodes and users, and applying the Ford-Fulkerson algorithm to determine the maximum flow within the network topology with virtual links. This approach is designed to facilitate efficient resource allocation for both computing and transmission tasks.

P. V. Lahande et al. (2024) conducted research on load balancing mechanisms within the WorkflowSim environment using the Sipht task dataset [15]. They tested various load balancing algorithms, including First Come First Serve (FCFS), Maximum – Minimum (Max – Min), Minimum Completion Time (MCT), Minimum – Minimum (Min – Min), and Round-Robin (RR). The study was organized into four phases, each characterized by different task lengths, and included sixteen scenarios with varying numbers of virtual machines (VMs) utilized in each scenario.

V. EVALUATION PARAMETERS

In order to evaluate load balancing algorithm following parameters were compared and evaluated. Many of researchers estimate parameters as per machine, network environment [8, 20, 22].

Makespan is defined as the time required for processing all thejobs or the maximum time required for completing a given set of jobs. Minimization of makespan ensures better utilization of the machines and leads to a high throughput [7].

$$J_{max}=\text{Max}\ \{ J_1, J_2, J_3,\ldots\ldots\ldots J_n\}$$

Total Flowtime is defined as the sum of completion time of every job or total time taken by all the jobs. Total flowtime of the schedule is computed using equation [8]:

$$F = \sum_{i}^{n} J_i$$

Completion Time variance is defined as the variance about the mean flowtime and is computed using equation [9]:

$$V = \frac{1}{n}\sum_{1}^{n}(J_i - \bar{F})^2$$

Where $\bar{F}$ is the mean flowtime.

Relative Percent Deviation (RPD)

$$RPD = \left[\frac{G - C^*}{C^*}\right] \times 100$$

where, G represents the global best solution obtained by the proposed algorithm for a given problem and C^*represents the upper bound value.

VI. CONCLUSION

Load balancing is a crucial element for ensuring optimal efficiency in distributed computing environments. In the realm of edge computing—a pivotal paradigm for modern systems—load balancing plays an essential role by enhancing system performance and easing the load on cloud services. This is achieved by relocating computational resources closer to the end-users, thus reducing communication overhead and improving the efficiency of processes at the edge. This article provides a thorough review of current load balancing techniques specific to edge computing. It covers various approaches, including those based on optimization algorithms, methods focused on traffic load management, and strategies designed for managing heterogeneous environments. The review highlights that most load balancing efforts are concentrated in distributed IoT networks, where dynamic and adaptable techniques are necessary to address fluctuating demands and conditions effectively. The study further reveals that incorporating genetic algorithms into task scheduling can significantly enhance performance, demonstrating improved efficiency and effectiveness. Looking ahead, future research could focus on developing models that do not require prior training for load balancing. Such advancements could simplify the implementation of load balancing strategies and make them more versatile and efficient in dynamic and varied environments.

REFERENCES

1. Du, M.; Wang, Y.; Ye, K.; Xu, C. Algorithmics of cost-driven computation offloading in the edge-cloud environment. IEEE Trans. Comput. 2020.

2. Li, C.; Bai, J.; Chen, Y.; Luo, Y. Resource and replica management strategy for optimizing financial cost and user experience in edge cloud computing system. Inf. Sci. 2020, 516, 33–55.
3. Li, C.; Bai, J.; Luo, Y. Efficient resource scaling based on load fluctuation in edge-cloud computing environment. J. Supercomput. 2020.
4. Hoang, K.D.; Wayllace, C.; Yeoh, W.; Beal, J.; Dasgupta, S.; Mo, Y.; Paulos, A.; Schewe, J. New distributed constraint reasoning algorithms for load balancing in edge computing. In International Conference on Principles and Practice of Multi-Agent Systems; Springer International Publishing: Cham, Switzerland, 2019; pp. 69–86.
5. Liu, Y.; Zeng, Z.; Liu, X.; Zhu, X.; Bhuiyan, M.Z.A. A novel load balancing and low response delay framework for edge-cloud network based on SDN. IEEE Internet Things J. 2019.
6. Puthal, D.; Ranjan, R.; Nanda, A.; Nanda, P.; Jayaraman, P.P.; Zomaya, A.Y. Secure authentication and load balancing of distributed edge datacenters. J. Parallel Distrib. Comput. 2019, 124, 60–69.
7. Ren, J.; Tian, H.; Lin, Y.; Fan, S.; Nie, G.; Wu, H.; Zhang, F. Incentivized Social-Aware Proactive Device Caching with User Preference Prediction. IEEE Access 2019, 7, 136148–136160.
8. Pranveer Singh, Raghuraj Singh Suryavanshi. "Survey on Load Balancing Techniques Shantanu Shukla Research Scholar". International Journal of Applied Engineering Research ISSN 0973-4562 Volume 14, Number 2, 2019.
9. Hu, P.; Dhelim, S.; Ning, H.; Qiu, T. Survey on fog computing: Architecture, key technologies, applications and open issues. J. Netw. Comput. Appl. 2017, 98, 27–42.
10. Kong, W.; Li, X.; Hou, L.; Li, Y. An efficient and credible multi-source trust fusion mechanism based on time decay for edge computing. Electronics 2020.
11. Salehe M., Hu Z., Mortazavi S.H., Capes T., Mohomed I. VideoPipe: Building video stream processing pipelines at the edge Middleware Industry 2019 - Proceedings of the 2019 20th International Middleware Conference Industrial Track, Part of Middleware 2019 (2019), pp. 43-49.
12. Saba, T., Rehman, A., Haseeb, K. et al. Cloud-edge load balancing distributed protocol for IoE services using swarm intelligence. Cluster Comput 26, 2921–2931 (2023).
13. S. Shao et al., "LBA-EC: Load Balancing Algorithm Based on Weighted Bipartite Graph for Edge Computing," in Chinese Journal of Electronics, vol. 32, no. 2, pp. 313-324, March 2023.
14. Y. Wang et al., "Load-Balancing Method for LEO Satellite Edge-Computing Networks Based on the Maximum Flow of Virtual Links," in IEEE Access, vol. 10, pp. 100584-100593, 2022.
15. P. V. Lahande, P. R. Kaveri, J. R. Saini, K. Kotecha and S. Alfarhood, "Reinforcement Learning Approach for Optimizing Cloud Resource Utilization With Load Balancing," in IEEE Access, vol. 11, pp. 127567-127577, 2023.
16. A. Bandyopadhyay et al., "EdgeMatch: A Smart Approach for Scheduling IoT-Edge Tasks With Multiple Criteria Using Game Theory," in IEEE Access, vol. 12, pp.
17. D. Baburao, T. Pavankumar, C.S.R. Prabhu Load balancing in the fog nodes using particle swarm optimization-based enhanced dynamic resource allocation method Appl. Nanosci. (2021), pp. 1-10.
18. Ajay Jangra, Neeraj Mangla. "An efficient load balancing framework for deploying resource schedulingin cloud based communication in healthcare", Measurement: Sensors, Volume 25, 2023.
19. Ali I.M., Sallam K.M., Moustafa N., Chakraborty R., Ryan M., Choo K.-K.R. An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems IEEE Trans. Cloud Comput., 10 (4) (2022), pp. 2294-2308.
20. F. Hoseiny, S. Azizi, M. Shojafar, F. Ahmadiazar, R. Tafazolli, PGA: A Priority-aware Genetic Algorithm for Task Scheduling in Heterogeneous Fog-Cloud Computing, in: Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2021, pp. 1–6.

21. 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," in IEEE Access, vol. 12, pp. 14334-14349, 2024.