International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Shared Cache Optimized Adaptive Load Balancing Strategy for IoT Devices

Tirupathi Puppala[1], Niranjan Polala[2]

Submitted: 07/05/2023 **Revised**: 15/07/2023 **Accepted**: 07/08/2023

Abstract: As the use of wireless communication expands, so does the need for more complex, easy-to-use, and low-cost solutions. The need for network solutions ranging from wireless sensor networks to wireless ad-hoc networks to the Internet of Things prompted academics to engage in the development of acceptable network solutions. Inventions made by researchers have led to an increase in the desire for additional advancements in current researchers. In the beginning, research and development focused on network protocols. IoT devices are being employed in a variety of industries and are amassing an enormous amount of data through sophisticated applications, regardless. This necessitates study into IoT network load balancing. As IoT networks become more overburdened, researchers have made many efforts to find ways to reduce the communication costs that result. In these studies, the IoT nodes were recommended to be evenly distributed in the network's load. The data gathered by IoT nodes and the applications that handle that data will eventually be moved to the cloud, but this will take time. A cloud-based load balancer meeting the needs of IoT network protocols is the difficulty here. A new technique is proposed in this study to deal with IoT network frameworks' load management. The main problem of this study is to develop a load balancer that considers the limited energy and processing capabilities of IoT nodes, yet with the goal of increasing the response time of the IoT network. Consideration has been given to the low-effort integrations with current IoT frameworks in the design of the suggested algorithm for load balancer.

Keywords: *Adaptive Load, IoT, Shared Cache.*

1. Introduction

This A game-changing networking invention is the Internet of Things, sometimes known as IoT. Three basic components have helped make the IoT networks a big success: gathering data from the open operating environment using sensors, autonomous machine-to-machine communication, and cloud computing for applications and data acquired by sensor agents. The Internet of Things (IoT) has become a vital aspect of current wireless communications because of the rising popularity of smart concepts in modern goods and lifestyles. A computerised innovation's layered hidden engineering is commonly evaluated by IoT frameworks. CPS, sensors, and machines are all included in the gadget layer. Physical system transports, distributed computing, and correspondence conventions compose the system layer, which totals and conveys data to the administration layer, which includes uses that manage and combine data into data that may be seen on the dashboard of the driver. The substance layer, often known as the user interface, is the highest level of the stack.

In 1968, Dick Morley developed the programmable rationale controller, which was used by General Motors in their programmed transmission manufacturing business. The history of the Internet of Things begins there. Individual components in the assembly chain were taken into account while designing these PLCs. It was in 1975 when Honeywell and Yokogawa unveiled the world's first DCS systems, the TDC 2000 and CENTUM, respectively. They were a further step in providing flexible process control across a plant, with the added benefit of removing the single point of failure in a central control room by transmitting control over the whole framework.

However, in light of the growing need for more advanced IoT load balancing algorithms for cloud-integrated apps and frameworks, this study suggests a new approach.

2. Foundational Strategy for Load Balancing on IoT

Henceforth, after setting the context of the research in the previous section of this work, in this section, the base line method for load balancing strategies for IoT networks are discussed.

Assuming that, for an IoT network, I[] of n number nodes, each node can be identified by IX. Thus, this can be formulated as,

$$I[] =< I_1, I_2, I_3, \ldots . I_n >$$
(1)

During the transmission of the data in the network, the IoT nodes are expected to process minimal amount of the data and few of the situations can be observed, where these

[1] *Kakatiya University, Warangal – 506009, INDIA*
ORCID ID: 0000-0001-9399-4873
[2] *Kakatiya Institute of Science Technology, Warangal – 506015, INDIA*
ORCID ID: 0000-0002-6005-7805
** Corresponding Author Email: puppalatirupathi@hotmail.com*

minimal data is becoming overload for the node capacity. In order to analyze the load condition, from each node in the network, the utilization must be extracted. Assuming that, the CX is the compute capacity of the node and C`X is the utilization of the compute capacity for any given node NX, thus this can be formulated as,

$$\boldsymbol{C_X < C'_X :: N_X \rightarrow OverLoad} \quad (2)$$

It is also understandable, that each node in the network has a specific range, which can be extracted from the properties table and also can be calculated using the standard IoT node properties. Here, it is assumed that using the function λ, the range for each node is extracted from the properties table.

Assuming that, for two different nodes as N_1 and N_2, the range are R_1 and R_2 respectively. Thus, this can be formulated as,

$$R_1 = \lambda(N_1.PT) \quad (3)$$

And,

$$R_2 = \lambda(N_2.PT) \quad (4)$$

Where, PT is the properties table.

On a given situation, if both the ranges overlap, then it is justified to mention that, in case of overloading on either of the nodes, the load can be distributed over two range overlapping nodes as,

$$\{N_1 \leftrightarrow N_2\} :: \{R_1 \cap R_2 \neq \phi\}, C_1 < C'_1 \quad (5)$$

This is the foundational strategy for load balancing in case of IoT networks.

Henceforth, in the light of the foundational strategy, in the next section of this work, the parallel and recent research outcomes are analyzed.

3. Parallel Research Outcomes

In the recent times, the parallel research outcomes have come a long way from the baseline method. This section of the work is dedicated to critically analyze the persistent research bottlenecks of the research.

The total performance of a parallel and distributed computing system is intimately linked to load balancing, as shown in the work of Q. Liu et al. [1]. It is true that communication and processing difficulties related to the Internet of Things (IoT) have been extensively researched in data centre contexts, but this has not been the case for edge scenarios. For the latter, processing data in a load-balanced manner is more difficult. In an IoT edge system, both the data sources and the network infrastructure are dynamic, unlike in a data centre. This is the fundamental reason. The performance model and runtime optimization for the whole system will be difficult to quantify given the diverse performance needs of IoT networks and edge servers. This paper proposes a load-balancing networking strategy for effective data processing in IoT edge devices to address this issue. A dynamic clustering solution for IoT networks based on the upcoming deep reinforcement learning (DRL) is presented in this study. This approach fulfils both the communication and compute balancing needs of IoT networks and edge servers. Dueling Double Deep Q-Learning Network (D3QN) model is implemented in this study, and our tests using real world datasets obtained from an autonomous car show that our suggested technique may achieve considerable performance increase compared to benchmark methods.

A dynamic load-balancing technique, EdgeSafe, was presented by C. Roy et al. [2] for delivering Safety-as-a-Service (Roy et al., 2018). Sensor nodes in a Safe-aaS architecture may be either stationary or mobile, depending on whether the cars they're coupled to are stationary or moving. As a result, the mobile sensor node's distance from the edge nodes in its vicinity varies. It is imperative that the data be processed at the edge nodes since it is so time critical. This work does load balancing in two phases, using road transportation as the application case for Safe-aaS. The initial step in this process is to determine the desired capacity ratio of the edge nodes within the sensor nodes' communication range. second step of this study uses Markowitz Portfolio Selection Theory for edge node selection. Edge nodes are selected based on their profitability and risk, and their portfolio is designed accordingly. After then, the utility of each node on the edge is tallied up separately. In order to minimise the utility of the edge nodes, this paper develops an optimization function. According to extensive simulation findings, EdgeSafe is capable of increasing data rates by 41.37 percent, 35.64 percent and 21.05% respectively, compared to the current schemes HO (Park et al. 2018), MLB (Lobinger et al., 2010) and Honeybee HO (Park et al. 2018). (Fernando et al., 2019).

According to A. Amjad et al. [3], the Internet of Things (IoT) has led to the development of linked infrastructures with more flexibility and efficiency for Industry 4.0. The term "Industrial IoT (IIoT)" has been coined to describe the IoT-enabled remote monitoring of a broad range of industrial operations as part of Industry 4.0. Data interoperability is the most difficult issue in enabling smooth real-time communication in the IIoT because of the enormous number of heterogeneous devices involved. Various IoT application layer protocols must be integrated with different data collecting protocols and hardware/software platforms in order to achieve data interoperability. The state-of-the-art has many such integrations, but it is rare to find an in-depth exploration and analysis of these techniques in a single research paper. The interoperability of application layer protocols for IIoT is

examined in this article as a result of a Systematic Literature Review (SLR) that examines 34 significant academic publications released between 2014 and 2020. As a result, the studies chosen for inclusion fall into three distinct categories: To address interoperability challenges, 13 papers addressed the integration of IoT application layer protocols; 6 papers addressed the integration of IoT and data collection protocols; and 15 papers addressed the integration of IoT protocols with hardware and software platforms. 1. An additional nineteen (19) tools that were included into the chosen research projects are also documented in this paper as a result, leading methods for integrating data, such as Gateways, are widely accepted. As a result of this research, it has been determined that, although researchers frequently propose centralised intermediaries (such as a gateway or a protocol converter) and distributed intermediaries (such as Middleware), in order to achieve the required interoperability for IIoT, a device-level approach is now required. Multi-industry interoperability issues including scalability, load balancing and a failure point in a system will be addressed.

According to study by A. Asghar et al. [4], health monitoring systems have begun experiencing challenges, such as efficient processing and latency, as a result of the rising volume of data. The Internet of Things (IoT), cloud computing, fog computing, and wireless sensor networks (WSN) have all been used in the development of health monitoring systems (IoT). Health monitoring systems in general have been built with cloud computing architecture in mind. Big-scale deployment of latency-sensitive healthcare applications is hampered by the significant delay generated by the cloud-based architecture while processing large amounts of data. Latency issues may be alleviated, resources can be more easily accessed, and security can be improved considerably using fog computing. To reduce latency and network utilisation, this study proposes a fog-based health monitoring system design for large-scale deployment of the health monitoring system, the authors introduce a novel Load Balancing Scheme (LBS) for balancing load among fog nodes. Researchers used iFogSim to run simulations and contrasted their findings with those of other approaches, such as the cloud-only implementation, Fog Node Placement Algorithm (FNPA), and LoAd Balancing Algorithm (LAB). With the suggested health monitoring system implementation, latency and network use will be drastically reduced in comparison to the current cloud-only model, as well as FNPA and LAB Scheme.

Y. Shao et al. have made yet another significant contribution with their work on software-defined Internet-of-things networking (SDIoT). [5] In large-scale IoT networks, per-flow sampling dramatically simplifies network monitoring by keeping track of all the current flows in the network and sampling the IoT devices on each flow route to obtain real-time flow data. Controller sampling preference and balancing device loads are intertwined in this system. When it comes to IoT devices in the flow route, controllers may opt to sample some of them for more accurate flow statistics. However, uniform sampling of the devices is preferred in order to maintain an equilibrium between their energy use and longevity. Markov decision processes are used to model the flow sampling issue in large SDIoT networks, and strategies that strike a suitable compromise between these two objectives are proposed in this research. The optimum policy, state-independent policies, and index policies are all examined (including the Whittle index and a second-order index policies). The most favoured policy is the second-order index policy: First, it's on par with the Whittle index policy in terms of performance; second, it surpasses state-independent policies significantly. State-independent and Whittle index policies are equal in terms of complexity, although this policy is simpler than optimum policy in terms of realizability and does not need previous knowledge of network dynamics.

An earlier study by H. Choi et al. [6] took a similar turn in this direction, as well. When it comes to cell range expansion (CRE) in dense heterogeneous networks, this research provides an online bias offset (BO) management strategy based on COMORL (HetNets). Comorl aims to increase the number of user devices (UEs) that meet their quality of service (QoS), particularly in terms of latency and data rates, by controlling BOs for CREs. In order to meet these needs, this study created a QoS satisfaction indicator that quantifies delays by taking into account both the QoS standards and the ratio of signal to interference and noise in a given channel (SINR). In addition, this study developed a cooperative multi-agent online reinforcement learning system to solve a Markov decision process model. Load-coupled base stations benefit from the planned COMORL concept. Throughput, delay satisfaction ratio, and fairness are all shown by our simulation findings for COMORL. A dynamic scenario including medium and full traffic loads is used to test if the suggested COMORL scheme improves the delay satisfaction ratio by as much as 27% and 30% compared to the max-SINR method.

Because of the scalability of the cloud and the need to minimise network latency, a new processing paradigm for the Internet of Things (IoT) has been proposed by Z. Nezami and colleagues [7]. It is, on the other hand, very difficult to achieve this level of flexibility in the dispersed networked ecosystem of heterogeneous computing resources at the edge-to-cloud continuum. Traffic patterns and a growing demand for low-latency services need a balance between service location and response time. A key component of efficient fog computing administration and operations is load-balancing. A decentralised load-balancing problem for the placement of Internet of Things (IoT) services is formulated and studied in this paper in order to minimise the cost of deadline violation, service deployment, and unhosted

services, in terms of (global) workload balance and (local) quality of service (QoS). It is recommended that an edge-to-cloud node-based decentralised multi-agent learning system, called EPOS Fog, be used to balance input workloads throughout the network and reduce service execution costs at the same time. In order to improve edge usage while minimising service execution costs, the agents collaborate to develop a set of feasible resource allocations. It has been shown that EPOS Fog outperforms First Fit and Cloud-based techniques in terms of of workload balancing as well as of Quality of Service. Using EPOS Fog, network nodes are more evenly distributed, and service execution delays are lowered by up to 25%. By collecting collective intelligence, the results indicate how dispersed computing resources on the edge may be more cost-effectively employed.

S. Aljanabi et al. [8] suggested that the need to equip these devices with high processing capabilities seems to execute the applications more rapidly and smoothly as the internet of things (IoT) devices and apps continue to grow. Even while manufacturers strive to offer IoT devices with the greatest technology, there are still certain disadvantages associated with running some advanced apps like virtual reality and smart healthcare based. Complex application activities related with the cloud servers are offloaded to the cloud servers and returned to the appropriate applications using a hybrid cloud-fog-offloading (HFCO) model. IoT nodes in the HFCO must decide on where to offload a high-requirement processing job if they can't perform it themselves. The choice is based on the task's needs and the presence of neighbouring nodes of fog. While many fog nodes and IoT nodes need to offload their jobs, selecting the optimal fog node for each activity is an issue. According to the needs of the applications and the circumstances of nearby fog nodes, this study proposes a unique solution to the issue, in which the IoT node may choose whether to offload duties to a fog node or to the cloud. The cloud or other fog nodes may be used to offload work, distributing the burden and improving present circumstances, so that activities can be completed more quickly. A Markov Decision Process (MDP) is used to model the situation (MDP). The model is solved, and the best offload strategy is selected using a Q-learning technique. By lowering latency and completing more jobs, the suggested strategy outperforms other approaches in numerical simulations.

Task deployment has become a research hotspot in the combined "cloud-edge" datacenter, according to Y. Dong et al. [9]. The present "cloud-edge" datacenter is overcrowded with most of the hosts, which might lead to an uneven load in the datacenter. As a result, existing research focuses primarily on the issue of unilateral load balancing of cloud or edge computing centres. JCETD (Joint Cloud-Edge Task Deployment) is a resource management and task deployment technique based on pruning algorithm and deep reinforcement learning that is designed to achieve efficient deployment of "cloud-edge" jobs and overall load balancing on the basis of joint "cloud-edge" deployment. In the first place, the "cloud-edge" hosts are pruned based on the physical host's attribute value. So, the algorithm's computational complexity will be reduced, and the system's computational efficiency will be improved by a group of joint hosts that are not dominant. Second, the "cloud-edge" model simulates task deployment as a deep reinforcement learning process. The cloud computing centre and the edge computing centre are able to effectively and appropriately distribute work via constant investigation and use of the system environment. To sum it up, the "cloud-edge" approach has the potential to improve computing performance while also better distributing workloads. When compared to previous studies, our findings reveal that our method dramatically lowers the overall completion time and average response time, allowing us to better serve our customers and achieve load balancing for our combined "cloud-edge" system.

By outsourcing demanding calculations to the omnipresent MEC server, mobile edge computing (MEC) may reduce the resource constraints imposed on mobile device users (MDU) [10]. However, existing offloading rules enable MDUs to send their jobs to the same linked small base stations (sBSs), which necessarily increases latency and limits performance gain owing to overload. It's also not clear how to deal with the problem of protecting sensitive information from being shared. As a result, in this research, in addition to offering a combined load balancing and computation offloading (CO) approach for MEC systems this work introduces a new security layer. At this point in the process, we offer a load balancing method for redistributing MDU resources efficiently across the sBS. As an additional layer of security, a novel advanced encryption standard (AES) cryptographic approach infused with electrocardiogram (ECG) signal-based encryption and decryption key is provided. It's also structured as an integrated model for balance, CO, and security, with the purpose of saving time and energy. According to extensive experimentation, we found that our approach, with and without extra security layers, may reduce system usage by about 68.2 percent and 72.4 percent, respectively.

There is adequate evidence to support the use of ISDN, as shown by N. A. S. Al-Jamali and colleagues [11]. Using an intelligent controller and an intelligent ISDN (software defined network), a network may be managed and controlled in an amazing manner. A partial recurrent spike neural network (PRSNN) congestion controller is proposed in this article to estimate the packet flow at the sensing plane in the software defined network-Internet of Things (SDN-IoT) to predict the next step ahead of packet flow and thus reduce the congestion that may occur. A congestion controller is added to the suggested model (spike ISDN-

IoT). In the suggested approach, this controller serves as a proactive controller. A reactive controller based on an artificial neural network is also proposed in this study to regulate clustering in the spike ISDN-IoT sensor's sensing region. An intelligent queuing model is provided to control the buffer capacity of the spike ISDN-IoT network's flow table buffers, hence enhancing the overall network's quality of service. In order to teach the PRSNN to alter its weight and threshold, a new training algorithm is developed. As compared to a convolutional neural network, the suggested model improves the quality of service by 14.36 percent.

According to X. Zheng et al. [12], sensor-cloud infrastructure provides a flexible and reconfigurable storage platform for massive amounts of sensed data in various application areas monitored by resource-limited networks such as wireless sensor networks (WSNs), ad-hoc networks, and the Internet of things (IoT) (IoT). These networks are employed in a variety of ways to aid people in their everyday lives because of their overwhelming qualities. However, these networks face a variety of challenges, including dependability in communication and processing, storage of the huge data, effective usage of on-board battery, maximum lifespan accomplishment, the least feasible average packet loss ratio, and dependable routing mechanisms. Even though several communications and load balancing strategies have been presented in the literature, these systems are either application-specific or overly complicated. Using the resources that are currently available, this research aims to provide a dependable communication and load balancing strategy for resource-constrained networks. Each sensing device Ci is obligated to calculate the transmission capabilities of its nearby devices, which are residual energy Er, hop count Hc, round trip time (RTTi), and processing cost under the proposed architecture in order to fulfil these aims. To begin with, a source device chooses a nearby device Ci with a minimal Hc value over those with highest Hc values in order to ensure reliable wireless connection. By requiring all devices in a network to locate and use just two of the four most trustworthy and shortest routes, this design ensures that data is always sent via the most secure and efficient channel. As a result, gadgets Ci that exist on these pathways do not consume their on-board battery faster than those that do not. When one or two nearby devices Ci use up to 80% of their on-board battery, the allocated weight-age factors may be fine-tuned such that the maximum weightage is given to the remaining energy Er and the lowest weight-age is applied to Hc value. In terms of average packet delivery ratio, average throughput, end-to-end latency, and total network lifespan, simulation findings reveal that the proposed reliable communication and load balancing system outperforms field-proven techniques.

Many different types of Internets of Things (IoT) devices are now linked to each other over the Internet. The bottleneck noted by W. Zhang et al. [13] is how to fulfil the needs of running IoT applications. Using the public cloud to host IoT applications is a cost-effective way to increase the computational power of these devices. IoT devices are located far away from the public cloud, which means that transmission delays will occur. Since IoT devices are close to MEC servers, this problem can be solved using mobile edge computing (MEC). When it comes to providing cloud services, pricing and load balancing are critical. Choosing an edge cloud service provider (ESP) based on pricing and load balancing is critical since it has a direct impact on the quality of the cloud service. Multi-cloud systems employ cloud service brokers (CSBs) to reserve resources from several cloud service providers (CSPs) and then provide cloud services to customers. Despite the fact that a lot of previous work has focused on offloading IoT applications to the MEC, many of these studies have only explored one scenario for multi-MEC deployments. This study examines the price and selection of services for offloading IoT applications in a multi-MEC system with several ESPs. It is described in this paper. This work specifically considers load balancing. Cloud service providers (CSBs) are initially given service pricing and load balancing techniques by CSB to optimise their revenues. Then, IoT users decide which ESP they want to utilise for their devices. The best answers may be found by using a backward induction method. Simulation data are used to verify the suggested strategy.

Resource-constrained smart devices that can perceive and interpret data make up the Internet of Things (IoT). In other words, it links a large number of smart sensing devices, or "things," as well as many heterogeneous networks. Different applications, such as smart health, smart home, and smart grid, use the Internet of Things (IoT). Pilot experiments at healthcare institutions have led to the notion of "smart healthcare," which is being studied in several countries. The safety of IoT devices and data is critical in IoT-enabled healthcare systems, and Edge computing provides a potential architecture for addressing these issues. As IoT devices in healthcare systems improve their communication and calculation speed, they have the potential to deliver low latency data services enabled by edge computing. IoT Edge-Based IoT-enabled healthcare systems use artificial intelligence (AI), a smart software-defined network (SDN) management, to balance network traffic, optimise resource usage, and save costs. Using SDN-based Edge computing, IoT devices may make better use of their limited resources. Patients' private sensitive data is at risk because of the limited power of these devices and the related data. To address these concerns in a healthcare IoT system, this study proposes a safe architecture for Edge computing based on SDN. The Edge servers in the proposed architecture use a lightweight authentication system to verify the identity of IoT devices. They then submit the patient's information to Edge servers for storage and analysis after authentication.

Load balancing and network optimization in the healthcare system are handled by an SDN controller that is linked to the Edge servers. Computer simulations are used to test the suggested framework. According to J. Li et al. [14], the findings reveal that the proposed framework offers superior solutions for IoT-enabled healthcare systems.

Work by Babou and colleagues [15] also showed substantial results as the edge computing system is becoming more popular and is projected to meet the ultra-low reaction time requirements of developing IoT applications. The growing traffic that requires highly sensitive delay has led to a new system design for Edge Computing, dubbed Home Edge Computing (HEC), which supports these real-time applications. the central cloud, HEC servers, and Multi-access Edge Computing (MEC) servers all work together to provide a three-tiered architecture for HEC. Latency issues on HEC servers may be alleviated using this paper's proposed approach. When there is an increase in traffic, there is an increase in processing time (delay) for requests on these servers. This paper proposes a novel approach, HEC-Clustering Balance, based on clustering and load balancing techniques. It enables us to distribute requests on HEC clusters in a hierarchical manner, which is an important aspect of the design for reducing latency. The findings reveal that HEC-Clustering Balance is more efficient than conventional clustering and load balancing strategies. On two experimental cases, our approach reduced processing time on HEC servers by 19 percent and 73 percent relative to the HEC design.

Henceforth, after realizing the parallel research improvements, the persistent problems in this domain of research are formulated mathematically in the next section of this work.

4. Problem Formulations

Further, after the detailed analysis of the parallel research outcomes carried out in the recent times, in this section of the work, the persisting problems in this domain of research are formulated mathematically. During the analysis of the parallel research outcomes in the previous section of this work, primarily two major research problems are identified. Firstly, the complete process for load balancing actually relies on the availability of the neighboring nodes. Considering a situation, where the neighboring nodes are not found, then the complete process is bound to fail.

Continuing from Eq. 5, Assuming that, for the node NX, the set of neighboring nodes are NN[]. The number of elements in the NN[] set can be calculated using the γfunction as,

$$\gamma\{NN[]\} = R \tag{6}$$

In case of very low number of members in the NN[] set, as $R \to 0$, the possibilities of finding the appropriate nodes for balancing will fail. Secondly, during the load balancing condition, if the neighboring node, which are in range, does not contain the same data values, then, the existing strategies are bound to incur additional time complexity for the overall process.

From Eq. 5 again, assuming that for the node NX, the set of neighboring nodes are identified with T1 time complexity. This can be formulated for each unit of time m as,

$$T_1 = \prod_{\{N_X \leftrightarrow N_Y\} :: \{R_X \cap R_Y \neq \phi\}, C_1 < C'_1} N[] * m \tag{7}$$

This can be reformed in the light of Eq. 6 as,

$$T_1 = R * m \Rightarrow O(m^2) | R \approx m \tag{8}$$

Further, assuming that, out of R number of elements in the neighbouring nodes and all these nodes does not contain the similar data as the source node. Thus, the data replication must be performed before load distribution. Assuming that, T2 is the time complexity for data replication with a unit time for k. Thus, this can be formulated as,

$$T_2 = R * K \tag{9}$$

This can be reformed as,

$$T_2 = O(K^2) \tag{10}$$

Henceforth, the total time complexity can be formulated as,

$$T_1{}^{T_2} = O(m^2)^{O(K^2)} \tag{11}$$

This can be re-written in terms of n as,

$$T_1{}^{T_2} = O(n^{2^2}) \tag{12}$$

Considering a large network, it is natural to realize that, the total time complexity will be significantly higher and this added time complexity will surplus the time reduced using the load balancing algorithms.

Henceforth, these two identified problems are to be solved. In the next section, the foundation of the proposed solution is furnished using mathematical models.

5. Proposed Solutions and Mathematical Models

After realizing the problems in the current research, in this section of the work, the proposed solutions are furnished using mathematical models.

Assuming that, the total data in the process for the considered network is D[] and each data item at various nodes are considered as DX. Thus, for a p number of data items, this can be formulated as,

$$D[] = < D_1, D_2, D_3, \ldots . D_p > \tag{13}$$

Assuming that FND() is the function to extract the data available in each node, thus for a two given nodes NX and NY, the relation can be formulated as,

$$D_X = FND(N_X) \quad (14)$$

And,

$$D_Y = FND(N_Y) \quad (15)$$

Here, the data points available in both the nodes must be identical to some extend as,

$$D_X \approx D_Y \quad (16)$$

Also, assuming that, these two nodes share arrange, thus in the light of Eq. 5, the following relation can be formulated,

$$\{N_X \leftrightarrow N_Y\} :: \{R_X \cap R_Y \neq \phi\} \quad (17)$$

Hence, combining Eq. 5, Eq. 16 and Eq. 17, the following formulation can be established as,

$$\{N_X \leftrightarrow N_Y\} :: \{R_X \cap R_Y \neq \phi, D_X \approx D_Y\}, C_X < C'_X \quad (18)$$

This strategy not only solves the problem of finding the neighboring nodes rather also ensure the availability of the data in the selected node. Hence, the reduction of the time complexity is significant here.

Assuming that, the time complexity of the proposed strategy as T, thus T can be formulated as,

$$T = O(n) \quad (19)$$

As the selection of the nodes also ensured the reduced replication of the data. Hence the proposed method has to do only one iteration for neighbor selection during load balancing strategy. Further, based on the proposed mathematical model, in the next section of this work, the proposed algorithms are furnished.

6. Proposed Algorithms

After the detailed analysis of the proposed strategy using the mathematical model, in this section the proposed algorithms are furnished. Firstly, the Data Affinity Detection with Process Detection algorithm is furnished.

Algorithm - I: Data Affinity Detection with Process Detection (DAD-PD) Algorithm

Input: List of Nodes as N[], Source Node as SN

Output: List of available Nodes as N1[]

Process:

Step - 1. For each element in N[] as N[i]

a. If N[i] is SN

b. Then, Next: i++

c. Else,

d. Read process table for N[i] as PT[i]

i. If PT[i] == PT[i+1] and PT[i]. Data == PT[i+1]. Data

ii. Then, N1[j]=N[i]

Step - 2. Return N1[]

The benefits of this proposed algorithm are already been furnished in the previous sections of this work. Secondly, Neighbour Selection using Range Detection algorithm is furnished.

Algorithm - II: Neighbour Selection using Range Detection (NS-RD) Algorithm

Input: List of Nodes as N[], Source Node as SN

Output: List of available Nodes as N2[]

Process:

Step - 1. For each element in N[] as N[i]

a. If N[i] is SN

b. Then, Next: i++

c. Else,

d. Extract the range for N[i] using Eq. 3

i. If N[i].range == N[i+1].range

ii. Then, N2[j]=N[i]

Step - 2. Return N2[]

The benefits of this proposed algorithm are already been furnished in the previous sections of this work. Thirdly, Load Balancing using Range and Data Affinity algorithm is furnished.

Algorithm - III: Load Balancing using Range and Data Affinity (LB-RDA) Algorithm

Input: List of Nodes as N[]

Output: Load Balancing Nodes as NLB[]

Process:

Step - 1. For each node in N[] as N[i]

a. Calcuate the capacity for N[i] as N[i].C using Eq. 2

b. Calcuate the utilization for N[i] as N[i].C' using Eq. 2

c. If N[i].C < N[i].C'

d. Then,

e. Mark as N[i] as over loaded

f. Check for N[i+1] in N1[] and N2[]

g. NLB[]=N[i+1]

Step - 2. Return NLB[]

The benefits of this proposed algorithm are already been furnished in the previous sections of this work.

Further, in the next section of this work, the obtained results are furnished.

7. Results and Discussions

After the realization of the proposed strategy and the proposed algorithms in the previous sections of this work, in the next section of this work the obtained results are furnished.

During the simulation operations, nearly 500 IoT device connected network for more than 100 iterations of testing, the obtained results are seen to be extremely satisfactory. For the representation purposes, only 10 iteration results are displayed.

Firstly, the dataset [20] total number of nodes and the overloaded nodes are compared [Table 1].

Table 1. Units for magnetic properties

Trail Seq. (#)	Number of Nodes	Number of Overloaded Nodes	Percentage of Over Loaded Nodes (%)
1	422	273	64.69
2	496	95	19.15
3	435	429	98.62
4	484	241	49.79
5	490	138	28.16
6	401	206	51.37
7	458	399	87.12
8	405	262	64.69
9	494	156	31.58
10	469	262	55.86

The results are visualized graphically here [Fig. 1].

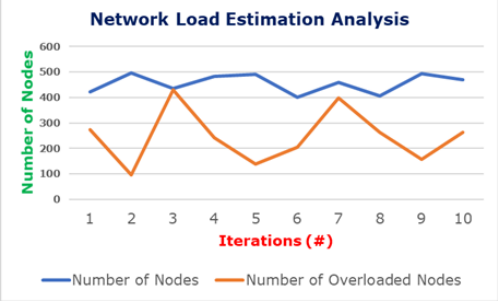

Fig. 1. Network Loaded Node Estimation Analysis

It is natual to realize that, the network under consideration is highly loaded and the application of the load balancing strategy is justified. Secondly, the energy consumption analysis is carried out [Table 2].

Table 2. Analysis of Energy Consumption

Trail Seq. (#)	*Energy Consumption (kW)*
1	8.594
2	9.952
3	7.322
4	7.546
5	6.187
6	6.469
7	6.011
8	9.863
9	9.621
10	9.018

The results are visualized graphically here [Fig – 2].

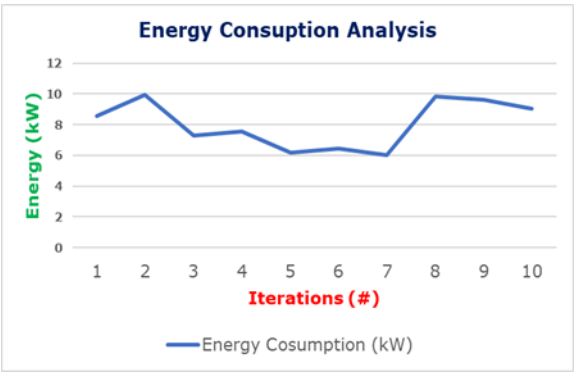

Fig. 2. Energy Consuption Analysis

It is notable that, during the actual load balancing process, the network maintained a nearly constant energy consuption. Thirdly, the load balancing time for the network are furnished here [Table 3].

Table 3. Load Balancing Time Analysis

Trail Seq. (#)	*Load Identification Time (ms)*	*Load Distribution Time (ms)*
1	0.193	0.152
2	0.145	0.152
3	0.165	0.165
4	0.145	0.172
5	0.114	0.181
6	0.167	0.138
7	0.126	0.131
8	0.140	0.159
9	0.168	0.187
10	0.170	0.197

The results are visualized graphically here [Fig. 3].

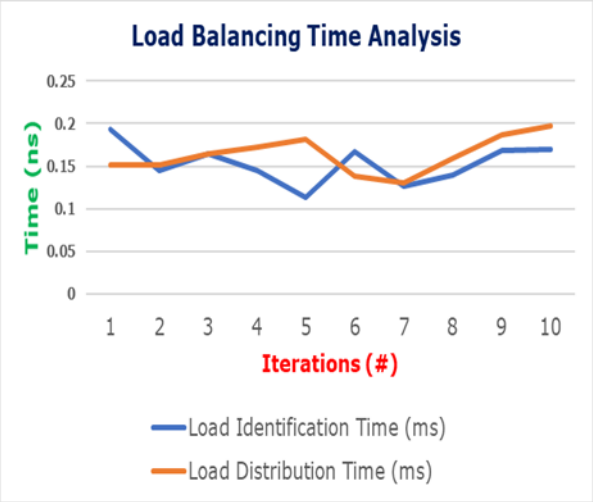

Fig. 3. Load Balancing Time Analysis

It is noteworthy to observe that; the load identification and the load distribution time is almost static.

Further, the improvements over the network life time are analyzed [Table 4].

Table 4. Network Life Time Analysis

Trail Seq. (#)	*Initial Network Life Time (hours)*	*Load Balanced Network Life Time (hours)*
1	1129.90	6779.40
2	832.50	5827.50
3	828.40	4142.00
4	1196.40	9571.20
5	1124.20	10117.80
6	843.10	5901.70
7	955.20	4776.00
8	1126.60	5633.00
9	1050.80	7355.60
10	1175.70	9405.60

Further, the results are graphically visualized [Fig. 4].

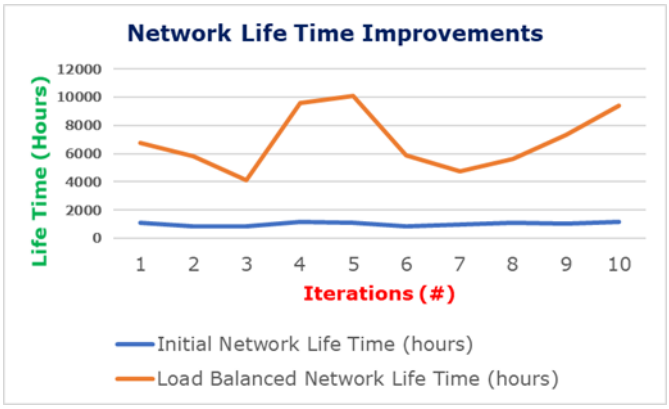

Fig. 4. Energy Consuption Analysis

Finally, the significant improvements of the network life time is the key indication, that the proposed algorithms have distributed the loads highly accurately. Further, in the next section of this work, the research conclusion is presented.

8. Conclusion

This means that as the use of wireless communication grows, there is a greater need for more complicated, easy to use and cheap solutions. A wide range of network solutions was needed, from wireless sensor networks to wireless ad-hoc networks to the Internet of Things. This led academics to get involved in the development of network solutions that were acceptable. Inventions made by researchers have led to a rise in the desire for more progress by current researchers. Network protocols were the focus of research and development at the start of the project. IoT devices are being used in a wide range of industries and are collecting a lot of

data through sophisticated apps, no matter what. This means that you need to learn more about IoT network load balancing. As IoT networks become more crowded, researchers have tried to find ways to cut down on the costs of communication. IoT nodes should be spread out evenly across the network's load in these studies, they said. A cloud service will one day store and process data from IoT nodes and the applications that use that data. This will take some time. IoT network protocols make it hard to find a cloud-based load balancer that can meet the needs of those protocols, though. In this study, a new method is used to deal with the load management of IoT network frameworks. The main goal of this study is to come up with a load balancer that takes into account the limited energy and processing power of IoT nodes, but also wants to improve the response time of the IoT network. In the design of the algorithm for a load balancer, it was thought about how easy it would be to integrate with current IoT frameworks.

References

[1] Q. Liu, T. Xia, L. Cheng, M. van Eijk, T. Ozcelebi and Y. Mao, "Deep Reinforcement Learning for Load-Balancing Aware Network Control in IoT Edge Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 6, pp. 1491-1502, 1 June 2022.

[2] C. Roy, S. Misra, J. Maiti and S. Nag, "EdgeSafe: Dynamic Load Balancing Among Edge Nodes for Provisioning Safety-as-a-Service in Vehicular IoT Applications," in IEEE Transactions on Vehicular Technology, vol. 70, no. 9, pp. 9320-9329, Sept. 2021.

[3] A. Amjad, F. Azam, M. W. Anwar and W. H. Butt, "A Systematic Review on the Data Interoperability of Application Layer Protocols in Industrial IoT," in IEEE Access, vol. 9, pp. 96528-96545, 2021.

[4] A. Asghar, A. Abbas, H. A. Khattak and S. U. Khan, "Fog Based Architecture and Load Balancing Methodology for Health Monitoring Systems," in IEEE Access, vol. 9, pp. 96189-96200, 2021.

[5] Y. Shao, S. C. Liew, H. Chen and Y. Du, "Flow Sampling: Network Monitoring in Large-Scale Software-Defined IoT Networks," in IEEE Transactions on Communications, vol. 69, no. 9, pp. 6120-6133, Sept. 2021.

[6] H. Choi, T. Kim, H. -S. Park and J. K. Choi, "A Cooperative Online Learning-Based Load Balancing Scheme for Maximizing QoS Satisfaction in Dense HetNets," in IEEE Access, vol. 9, pp. 92345-92357, 2021.

[7] Z. Nezami, K. Zamanifar, K. Djemame and E. Pournaras, "Decentralized Edge-to-Cloud Load Balancing: Service Placement for the Internet of Things," in IEEE Access, vol. 9, pp. 64983-65000, 2021.

[8] S. Aljanabi and A. Chalechale, "Improving IoT Services Using a Hybrid Fog-Cloud Offloading," in IEEE Access, vol. 9, pp. 13775-13788, 2021.

[9] Y. Dong, G. Xu, M. Zhang and X. Meng, "A High-Efficient Joint 'Cloud-Edge' Aware Strategy for Task Deployment and Load Balancing," in IEEE Access, vol. 9, pp. 12791-12802, 2021.

[10] W. -Z. Zhang et al., "Secure and Optimized Load Balancing for Multitier IoT and Edge-Cloud Computing Systems," in IEEE Internet of Things Journal, vol. 8, no. 10, pp. 8119-8132, 15 May15, 2021.

[11] N. A. S. Al-Jamali and H. S. Al-Raweshidy, "Intelligent Traffic Management and Load Balance Based on Spike ISDN-IoT," in IEEE Systems Journal, vol. 15, no. 2, pp. 1640-1651, June 2021.

[12] X. Zheng et al., "A Reliable Communication and Load Balancing Scheme for Resource-Limited Networks," in IEEE Access, vol. 8, pp. 179921-179930, 2020.

[13] W. Zhang, X. Li, L. Zhao, X. Yang, T. Liu and W. Yang, "Service Pricing and Selection for IoT Applications Offloading in the Multi-Mobile Edge Computing Systems," in IEEE Access, vol. 8, pp. 153862-153871, 2020.

[14] J. Li et al., "A Secured Framework for SDN-Based Edge Computing in IoT-Enabled Healthcare System," in IEEE Access, vol. 8, pp. 135479-135490, 2020.

[15] C. S. M. Babou et al., "Hierarchical Load Balancing and Clustering Technique for Home Edge Computing," in IEEE Access, vol. 8, pp. 127593-127607, 2020.

[16] S. Durkovic and Z. Čiča, "Multicast Load-Balanced Birkhoff-Von Neumann Switch With Greedy Scheduling," in IEEE Access, vol. 8, pp. 120654-120667, 2020.

[17] M. M. Shahriar Maswood, M. R. Rahman, A. G. Alharbi and D. Medhi, "A Novel Strategy to Achieve Bandwidth Cost Reduction and Load Balancing in a Cooperative Three-Layer Fog-Cloud Computing Environment," in IEEE Access, vol. 8, pp. 113737-113750, 2020.

[18] Y. Chen, Y. Sun, T. Feng and S. Li, "A Collaborative Service Deployment and Application Assignment Method for Regional Edge Computing Enabled IoT," in IEEE Access, vol. 8, pp. 112659-112673, 2020.

[19] J. Wang, H. Zhang, Z. Ruan, T. Wang and X. Wang, "A Machine Learning Based Connectivity Restoration

Strategy for Industrial IoTs," in IEEE Access, vol. 8, pp. 71136-71145, 2020.

[20] https://www.kaggle.com/caesarlupum/iot-sensordata

[21] Kumar, L. R. ., Ashokkumar, C. ., Pandey, P. S. ., Kannaiah, S. K. ., J, B. ., & Hussan, M. I. T. . (2023). Security Enhancement in Surveillance Cloud Using Machine Learning Techniques. International Journal on Recent and Innovation Trends in Computing and Communication, 11(3s), 46–55. https://doi.org/10.17762/ijritcc.v11i3s.6154

[22] Robert Roberts, Daniel Taylor, Juan Herrera, Juan Castro, Mette Christensen. Leveraging Machine Learning for Educational Data Mining. Kuwait Journal of Machine Learning, 2(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/176

[23] Dhabliya, D., Parvez, A. Protocol and its benefits for secure shell (2019) International Journal of Control and Automation, 12 (6 Special Issue), pp. 19-23.