DATA CONGESTION HANDLING IN SENSORS ENABLED INTERNET OF THINGS

A thesis Submitted to

DELHI TECHNOLOGICAL UNIVERSITY

For the Award of degree of

DOCTOR OF PHILOSOPHY

In

Computer Science and Engineering

By

Aastha Maheshwari
2K18/PHDCO/502

Under the Supervision of

Dr. Rajesh Kumar Yadav
Assistant Professor
Department of Computer Science and Engineering
Delhi Technological University

Dr. Prem Nath
Associate Professor
Department of Computer Science and Engineering
H.N.B. Garhwal University

Department of Computer Science and Engineering
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042, India
2023

DELHI TECHNOLOGICAL UNIVERSITY
(Govt. of National Capital Territory of Delhi)
BAWANA ROAD, DELHI – 110042

DECLARATION

I, Aastha Maheshwari (2K18/PHDCO/502), hereby affirm that the research work presented in my thesis titled "**DATA CONGESTION HANDLING IN SENSORS ENABLED INTERNET OF THINGS**" is an original contribution, conducted under the guidance of Dr. Rajesh Kumar Yadav from the Department of Computer Science and Engineering at Delhi Technological University, and Dr. Prem Nath from the Department of Computer Science and Engineering at H.N.B. Garhwal University. This thesis has not been previously submitted to any other academic institution for the purpose of obtaining a degree or diploma. Throughout the writing process, I have adhered to the prescribed Ph.D. rules and regulations set forth by the Institute.

I confirm that the thesis does not contain any classified information. Whenever I have utilized external sources, proper acknowledgement has been provided by citing them within the text and including them in the reference list. Direct quotations from external sources have been explicitly identified by quotation marks and have been appropriately referenced both in the text and the reference list.

Date:

Aastha Maheshwari
2K18/PHDCO/502

DELHI TECHNOLOGICAL UNIVERSITY
(Govt. of National Capital Territory of Delhi)
BAWANA ROAD, DELHI – 110042

CERTIFICATE

This is to certify that the research work presented in this thesis titled "**DATA CONGESTION HANDLING IN SENSORS ENABLED INTERNET OF THINGS**" by Aastha Maheshwari (2K18/PHDCO/502) is an original contribution conducted under the guidance of Dr. Rajesh Kumar Yadav from the Department of Computer Science and Engineering at Delhi Technological University and Dr. Prem Nath from the Department of Computer Science and Engineering at H.N.B. Garhwal University.

This thesis has not been previously submitted for any other degree or diploma. Aastha Maheshwari has followed the prescribed Ph.D. rules and regulations throughout the writing process.

The thesis does not contain any classified information. Proper acknowledgment has been given for external sources through citations within the text and inclusion in the reference list. Direct quotations have been appropriately identified and referenced.

Date:

Dr. Rajesh Kumar Yadav
Assistant Professor (Supervisor)
Department of Computer Science and Engineering
Delhi Technological University

Dr. Prem Nath
Associate Professor (Joint Supervisor)
Department of Computer Science and Engineering
H.N.B. Garhwal University

ACKNOWLEDGEMENT

I want to express my sincere gratitude to **Dr. Rajesh Kumar Yadav**, Assistant Professor, Department of Computer Science and Engineering, Delhi Technological University, Delhi, and **Dr. Prem Nath**, Associate Professor, Department of Computer Science & Engineering, H.N.B. Garhwal University, for providing valuable guidance and constant encouragement throughout the work. Their immense knowledge, motivation, expertise, and insightful comments have helped me immensely at every stage of the preparation of this research plan.

I would also like to extend my heartfelt thanks to the DRC and SRC committees for their valuable insights, suggestions, and critical evaluation of my research work. Their expertise and scholarly advice have played a crucial role in improving the quality of this thesis.

I would like to sincerely thank the esteemed members of the Department of Computer Science & Engineering at Delhi Technological University. Their unwavering support and provision of invaluable research resources have been instrumental in the successful completion of this work. Their guidance and assistance have been a constant source of motivation throughout my academic journey.

Moreover, I am profoundly grateful to my parents (Mr. Pankaj Maheshwari and Mrs. Poonam Maheshwari), my in-laws (Late. Mr. Rajiv Maheshwari, Mrs. Sunita Maheshwari and Mrs. Nehal Maheshwari), my husband (Mr. Aditya Maheshwari), my daughter (Ms. Aarya Maheshwari) and my siblings (Mrs. Nishtha Maheshwari and Mr. Vishwas Maheshwari) whose unconditional support and blessings have been a cornerstone of my achievements.

Their unwavering faith in my abilities, endless lessons, moral support, and care have played an integral role in shaping my academic pursuits. I hold the utmost respect and affection for my elders, whose wisdom and encouragement have guided every step of my journey.

I recognize that this accomplishment would not have been possible without the collective efforts and contributions of all those mentioned above. Their presence in my life has been truly invaluable, and I am forever indebted to them for their unwavering support and belief in my abilities.

Aastha Maheshwari
2K18/PHDCO/502

ABSTRACT

The Internet of Things (IoT) is a network of physical devices connected to the Internet, enabling them to interact with their internal states or the external environment. This transformative technology has diverse applications in various domains, such as healthcare monitoring, transportation, and smart cities. The growth of IoT networks led to a significant increase in data traffic. However, this surge in data, coupled with the limitations of constrained IoT devices, has created a bottleneck in the network, resulting in congestion.

Congestion poses several issues, particularly in terms of packet delivery. The overwhelming data traffic strains the network infrastructure, causing delays and hindering the timely delivery of packets. Additionally, the limited storage capacity of IoT devices exacerbates the problem, leading toa substantial number of packet losses.

This congestion problem hampers the efficiency and throughput of IoT networks. It disrupts the smooth flow of data and jeopardizes the integrity of the entire network. Addressing congestion in IoT networks is crucial to ensure seamless communication and optimal performance. Therefore, it becomes imperative to develop effective strategies and solutions to mitigate congestion, improve data traffic management, and enhance the overall performance of IoT networks. By doing so, we enabled the successful delivery of packets, reduce packet losses, and ensure the smooth operation of IoT applications and services.

This thesis aimed to investigate the existing literature on congestion control in IoT networks and identify the research gap. Specifically, we examined the focus of most authors, which primarily revolved around congestion control without adequately determining its occurrence in the IoT network. While packet loss and delay were commonly used indicators of congestion, congestion problems could also be influenced by factors such as link failure and channel noise, making them less reliable.

Therefore, we proposed more accurate schemes to detect and control congestion in IoT networks by considering a broader set of parameters in the prediction process. Additionally, we addressed the limitations of applying traditional IP-

based congestion control approaches to resource-constrained IoT environments. We emphasized the importance of incorporating congestion prediction approaches to effectively manage congestion before it impacted network performance. Furthermore, we recognized the significance of accounting for the limited resources of IoT devices, the heterogeneous nature of IoT networks, and the dynamic changes in network conditions when designing congestion control techniques. By filling this research gap, we aimed to contribute to the development of robust and efficient congestion control mechanisms for IoT networks that were focused on resource control schemes where we offloaded data packets or routed the packets in a congestion-aware manner.

To address the above research gaps, we defined three primary objectives:

The first objective was to design an approach for predicting congestion in IoT networks by considering multiple parameters. We utilized a Deep Neural Network-Restricted Boltzmann Machine (DNN-RBM) model to detect data congestion. The input to the Deep Neural Network (DNN) included performance factors such as congestion window, throughput, propagation delay, Round Trip Time (RTT), number of packets sent, and packet loss. The Restricted Boltzmann Machine (RBM) was employed to optimize the weights of the proposed DNN-RBM model.

The second objective focused on designing an approach for congestion control by implementing data offloading techniques. Data offloading techniques involved transferring data from one network or device to another to relieve congestion or improve performance. Rather than requiring the child node to select a new parent node, our approach identified a suitable neighbour node capable of sharing the load and assisting the congested node. The approach involved two steps: identifying the congested node and selecting the appropriate neighbour node to carry the data packets.

The third objective entailed designing an approach for congestion-aware data transmission in IoT networks. We employed an improved Analytic Hierarchy Process (AHP) method to select the most suitable node based on multiple parameters such as the distance, hop count, residual energy, link quality, and

buffer occupancy. This approach enabled efficient hop-to-hop data communication/ transmission while considering congestion levels and network efficiency.

By addressing these objectives, our research aimed to enhance congestion prediction accuracy, optimize congestion control through intelligent data offloading, and improve overall data transmission efficiency in IoT networks while accounting for congestion awareness.

CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Expanded Form
IoT	Internet of Things
IETF	Internet Engineering Task Force
LPWPAN	Low-Power Wireless Personal Area Networks
TCP	Transmission Control Protocol
CoAP	Constraint Application Protocol
RTT	Round Trip Time
UDP	User Datagram Protocol
OWL	Web Ontology Language
RFID	Radio Frequency Identification
EPC	Electronic Product Code
WSN	Wireless Sensor Network
DNN-RBM	Deep Neural Network-Restricted Boltzmann Machine
GPRS	General Packet Radio Service
EXI	Efficient XML Interchange
MAC	Medium Access Control
XML	Extensible Markup Language
RDF	Resource Description Framework
BOR	Buffer Occupancy Ratio

ETX	Expected Transmission Count
EC_i	Energy Consumed by Node i
IE_i	Initial Energy of Node i
RE_i	Remaining Energy of Node i
ACK	Acknowledgment
RPL	Routing Protocol for Low-Power and Lossy Networks
HTTP	Hypertext Transfer Protocol
REST	Representational State Transfer
RFC	Request for Comments
OF	Objective Function
DAG	Directed Acyclic Graph
DODAG	Destination-Oriented Directed Acyclic Graph
ICMPv6	Internet Control Message Protocol version 6
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DAO	Destination Advertisement Object
DNN	Deep Neural Network
RBM	Restricted Boltzmann Machine
AODV	Ad-hoc On-Demand Distance Vector
ANN	Artificial Neural Network
P_{ij}	Value of the ith alternative for the jth parameter

$(\overline{p_j})$	Mean value of the jth parameter
ETX	Expected Transmission Count
Σ	Standard Deviation
V_{jk}	Linear correlation matrix between jth and kth parameters
β_{kj}	Amount of information obtained by parameter k of j alternative
w_k	Weight value for the kth parameter
N_P	Currently occupied buffer
Q_P	Actual size of the buffer
P_{data}	Data packet probability of successful delivery
P_{acK}	Acknowledgement packet's successful delivery
AHP	Analytic hierarchy process
M	Judgement matrix
m_{ij}	Degree of variable estimating probability of the event
y_i	Parameter value
Λ	Eigenvalue of the judgement matrix m
C. I	Consistency index
C.R	Consistency ratio
$\overrightarrow{W}_{i.k}$	Weight value provided to i^{th} index under the k_{th} judgment matrix

CHAPTER-1

INTRODUCTION

With the help of the Internet of Things (IoT), physical equipment like sensors may communicate automatically with one another, negating the need for human involvement [1]. This development has greatly improved communication between people, things, sensors, and services. Establishing a network environment that guarantees reliable communication among diverse things, like sensors, automobiles, and everyday products like refrigerators, microwaves, dishwashers, and pharmaceuticals, independent of the network or time is the main goal of the Internet of Things (IoT) [2]. The fast spread of IoT devices is expected to cause a significant increase in communication traffic. By 2025, there will probably be 10 billion linked gadgets, according to predictions [13]. Congestion control is essential for managing this increased traffic and ensuring reliable internet connectivity. The primary objective of congestion control is to facilitate reliable communication within the IoT network while minimizing delays and packet loss.

The Internet of Things (IoT) is a cutting-edge communication technology that improves people's quality of life by incorporating intelligent sensors into real-world items, or "Things," that could communicate. In theory, the IoT allows for constant connectivity between people and objects, independent of the time, place, path/network, or service [21]. The Internet of Things (IoT) architecture connects things to carry out functions specified by real-world applications, or "apps." Human involvement is not necessary since decision-making and action execution are based on predetermined rules. IoT networks improve inter-machine communication as well. In the sections that follow, we have given a succinct overview of IoT and its congestion control approaches to help readers stay up with the industry's changing trends.

The Internet Engineering Task Force (IETF) has taken on the task of standardizing IoT [3]. One of their working groups is dedicated to developing a routing approach specifically tailored for low-power and lossy networks commonly found in constrained IoT environments [4]. These networks typically consist of border routers, gateways, and constrained nodes, where data collected by the nodes may be sent directly to the border router or through intermediate nodes. The applications of IoT are diverse and have far-reaching impacts on various aspects of human life, including smart urban communities, transportation, and homes. The growth of IoT opens up opportunities to enhance urban areas by improving infrastructure, optimizing public transport, reducing traffic congestion, and promoting citizens' well-being through network-enabled services.

The widespread adoption of Wi-Fi in home automation has played a significant role in connecting electronic devices like smartphones, TVs, and other gadgets, thereby contributing to the expansion of IoT [5-7]. Furthermore, IoT advancements enable real-time monitoring capabilities, allowing continuous observation of the mental state of hospitalized patients. In the domain of smart health, sensors are utilized to collect behavioural data, which is then analyzed and stored in the cloud or gateway. This information is wirelessly transmitted to caregivers for further evaluation [8-10]. Despite the exciting possibilities presented by IoT applications and scenarios, their implementation comes with challenges. Performance-related concerns such as self-organization, scalability, data volumes, power supply, data interpretation, wireless communications, and interoperability need to be addressed [11, 12].

In one scenario, a significant number of IoT devices experience slow processing speeds and limited storage, leading to traffic congestion when multiple devices attempt to communicate with each other. The immense volume of data generated by the exponential growth of embedded devices, industrial systems, smart buildings, smart cities, and smart power management in our daily lives is the main cause of this congestion [12]. The appeal of IoT lies in its ability to connect billions of devices that may communicate and interact without human intervention.

Often referred to as "Things," IoT devices have restricted memory, processing power, and energy resources. These finite capabilities influence the service quality of the IoT networks, and the limitations of these devices may contribute to congestion within and between IoT networks. The Internet Engineering Task Force (IETF) categorizes IoT devices into different classes based on their memory capacity [14].

Class 0 devices: Class zero devices feature 100 kilobits of Flash memory and 10 kilobits of RAM. A Wireless Sensor Network (WSN) mote, for instance, is an example of a zero-category device [15].

Class 1 devices: The minimum memory size for a class one device is 100 kilobytes of Flash memory and 10 kilobytes of RAM. Compared to class zero devices, class one devices may run memory-intensive software. For instance, class one devices employ secure communication and routing protocols.

Class 2 devices: These devices may store up to 250 kilobytes of Flash memory and 50 kilobytes of RAM. When compared to either MOTE (class zero) or a communication protocol, class two devices are in a superior position (class one). They can't be contrasted with expensive IoT gadgets, though. Nowadays, class two device categories include conventional Internet hosts.

The resources in the class cannot manage traffic levels ranging from moderate to high because of their capacity restrictions. Because of the limited capacities, the network becomes congested. Older network designs, protocols, and communication technologies make it difficult for IoT devices to connect with each other via the Internet. Modern technology is used by IoT networks, according to experts. Numerous issues have been brought forward by researchers, including heterogeneity, security, traffic, energy, mobility, reliability, and service quality [16]. Congestion occurs when transmitting and receiving rates are out of phase or when two or more nodes compete for transmission on a single shared connection.

IoT applications communicate with each other using Low-Power Wireless Personal Area Networks (LPWPAN) to connect to the internet through a gateway. Similar to traditional networks, IoT networks may experience congestion at end devices, gateways, or intermediary nodes. Congestion control strategies may be categorized into two types: (i) End-to-end [17] and (ii) Hop-by-hop [18]. Both wireless and cable-based IoT networks employ various congestion control techniques [19].

In wired networks, the source node in a congestion control scenario receives feedback from the destination nodes. For end-to-end congestion control, resources are allocated at both the source and destination nodes, while intermediate nodes do not participate in congestion-easing operations [17, 18]. While the Transmission Control Protocol (TCP) is widely used for reliable communication between source and destination in wired networks, it may not be suitable for embedded IoT devices due to its large header size. In IoT applications, the User Datagram Protocol (UDP) is commonly employed but lacks built-in congestion control. Recent studies have introduced protocols like the Constraint Application Protocol (CoAP) with basic built-in congestion control at the application layer to address this. These solutions utilize

Round Trip Time (RTT) in a manner similar to TCP to detect and effectively manage congestion.

A hop-by-hop routing strategy considers intermediary devices along the route. In this approach, the previous router receives input from intermediate nodes to determine the end-to-end path [19, 20]. However, due to the unpredictable nature of wireless communications in IoT networks, achieving a reliable end-to-end connection may not always be possible. In such cases, hop-by-hop routing proves to be faster and more efficient than end-to-end congestion handling solutions. The choice between end-to-end or hop-by-hop transmission should be based on the specific requirements of the underlying application, considering factors such as reliability and time sensitivity.

1.1. IoT Components

The Internet of Things (IoT) is composed of various elements, including sensors, nodes, smart items, and things, which connect people in our daily lives. These elements may be categorized as follows [22]:

1. Identification: Identification is the process of distinguishing and recognizing IoT objects or devices in the network in a unique way. It entails giving each device a special identification code, such as a MAC address, IP address, or another code. Through this identification, devices may be identified and addressed on the network, facilitating easy management and communication. Techniques such as Radio Frequency Identification (RFID), Ubiquitous Code (code), and Electronic Product Code (EPC) [24, 23] are used for object identification. For instance, if there's a humidity sensor named H1, the object ID H1 corresponds to its name. The address serves as a unique identifier within the communication network, and the study focuses on IPv4, IPv6, and 6LoWPAN addressing techniques.

2. Sensing: Sensing is the capacity of Internet of Things (IoT) devices to gather information from their surroundings using a variety of sensors. There are several sorts of sensors that may be used, including temperature, humidity, motion, and light sensors. Sensing is an essential component of the Internet of Things because it enables devices to understand their physical surroundings and collect pertinent data for processing and analysis.

3. Communication: IoT devices must be able to communicate data and information with one another as well as with a central system, such as a cloud server, in order to be considered to be in communication. Wi-Fi, Bluetooth, Zigbee, cellular networks, and even specific low-power protocols like LoRaWAN are common communication approaches used by IoT devices. To provide sensor data, receive orders, and coordinate operations across IoT devices, communication must be effective.

4. Computation: The ability of IoT devices to process data is referred to as computation. IoT devices frequently have integrated processors or microcontrollers that enable them to carry out calculations locally as they develop in sophistication. These calculations may require data filtering, analysis, judgment, and occasionally even the execution of sophisticated algorithms.

Real-time or almost real-time answers are made possible via local computation, which helps minimize the latency of data transfer and processing. IoT devices use hardware and software for computations. Hardware components such as Raspberry Pi [25] and Arduino [26] are employed for computation, and software code is an essential part of the design and development phase for smart devices.

5. Services: IoT applications utilize services for tasks like item identification, data aggregation, and decision-making. Four main types of IoT services [27] are as follows:

a. Identity-Related Services: Used for identification of physical items in IoT networks, e.g., shipping and logistics.

b. Information Aggregation Services: Collect clear data from sensors and transfer it to IoT apps, e.g., smart healthcare and smart grid.

c. Collaborative Aware Services: Use information aggregation services to make decisions based on received data, e.g., smart traffic management and smart industrial automation.

d. Ubiquitous Services: Allow Collaborative Aware Services to be accessible anytime, anywhere, and by anyone [22], with Smart City being a successful example.

6. Semantics: IoT semantics involves extracting information from data scattered across different machines. Techniques like Resource Description Framework (RDF) [28], Web Ontology Language (OWL) [29], and Efficient XML Interchange (EXI) [30] are used for IoT semantics. EXI, particularly for converting XML to binary representation, stands out for its efficiency and lower computational power requirements.

1.2. Communication Technologies:

Networks and devices with internet access are used by sensors to communicate. Any intelligent Internet of Things network and apps must have communication technologies. In papers [31, 32] authors analyze and explore communication technology. We have provided a brief overview of current IoT communication technologies in the sections that follow, building on significant prior research for more investigation, analysis, and review. The essential factors to consider while thinking about IoT network connectivity are listed below [33].

Frequency: Taking signal interference and channel blockage into account.

Range: Depending on the deployment location, such as a state's institution organization.

Data Rate: The amount of available bandwidth determines the rate at which data changes.

Battery life: The presence of a reliable battery backup.

Topologies: Nodes are arranged physically in topologies to facilitate communication.

MAC: The channel access strategies are defined by the Medium Access Control layer.

1.3. Challenges in IoT:

In the following sections, we have explored crucial characteristics of IoT networks related to installation, commissioning, operation, services, and maintenance. These aspects significantly impact the effectiveness of IoT networks and the services it provides.

1. Standardization: One of the challenges in IoT is the fragmentation of networks and conflicting proprietary standards, making the establishment of common standards difficult. The growth of IoT devices and networks has been hindered by the presence of rival proprietary solutions, leading to issues with radio access, semantic interoperability, security, and privacy [34, 22].

2. System Architecture: IoT networks and devices utilize various system architectures, making it challenging to develop a common framework. Proprietary standards further complicate the development of a unified architecture. Some open and flexible layered designs, like IoT-A [35] and IoTivity [36], have been proposed to address interoperability concerns, but a common reference model is yet to be achieved [22, 37].

3. Integration and Interoperability: IoT networks face challenges in achieving interoperability due to the vast diversity of devices and platforms. Ensuring seamless service regardless of hardware constraints requires network compatibility among applications, device makers, and networks [22].

4. Reliability: For IoT networks to be considered reliable, all its components and applications must operate without interruption for a predetermined time. Communication that promptly acknowledges responses plays a crucial role in achieving reliability. However, limited resources in many IoT networks leave them vulnerable to channel loss and buffer overflows in busy networks, making it challenging to maintain reliability [22].

5. Availability: In the context of IoT, availability refers to the ability to provide uninterrupted hardware and software services anytime and anywhere. High availability may be achieved by allocating greater resources to overworked devices and services. However, redundant devices may face availability issues in IoT networks with limited capacity [38].

6. Fault Tolerance: Fault tolerance is essential for maintaining consistent system behaviour in the face of high traffic, network flaws, or device failures. Fault-tolerant solutions control network traffic congestion and address network or device failures. However, traditional fault tolerance techniques may not work optimally when new features and services are added to the system [39, 40]. Intelligent or adaptive fault tolerance techniques are needed to cope with system changes [41].

7. Scalability: Scalability is achieved by adding more hardware, software, or features without compromising other network characteristics. In dense IoT networks, certain protocols may suffer due to the inability of any node to maintain information about all its neighbours in the routing table. Low-bandwidth communication techniques may also result in coverage and bottleneck issues in scalable networks [42].

8. Mobility: Mobility in IoT refers to the ability to change the position of smart devices over time or move them from one network to another. The diverse nature of IoT devices and networks presents challenges in providing seamless services to mobile consumers. Issues like blackouts, hand-offs, and unavailable networks may disrupt mobile network operations [43, 22].

9. Management and Self-Configuration: IoT networks require lightweight protocols and services to address their diverse nature effectively. The challenges faced by IoT networks call for innovative solutions to ensure efficient management and self-configuration [22].

10. Performance and Quality of Service: Measuring the performance of IoT networks is challenging due to the diversity of devices, networks, and technologies involved. An efficient IoT system should provide cost-effective and evolving services to meet customer needs. Metrics such as connection speed, processing speed, device form factor, and cost are vital for evaluating IoT networks performance [22].

11. Power and Energy Usage: IoT networks heavily rely on limited battery resources, and massive data transfers may deplete battery capacity, affecting network performance. Reducing energy usage in IoT is a critical area of study [44].

12. Privacy and Security: The varied and heterogeneous nature of IoT networks and devices poses challenges for ensuring security and privacy. Unlike traditional systems, IoT networks have limited resources and power availability, making conventional security solutions inadequate. Innovative research on network security, analysis, and intrusion prediction is crucial for the security of smart devices and networks [45, 46].

13. Congestion: With the rapid proliferation of IoT devices and applications, congestion in IoT networks is becoming a pressing concern. The limited availability of resources in IoT networks may lead to network congestion, potentially compromising the quality of services provided. Effective congestion control mechanisms are essential for addressing this challenge [47, 48].

1.4. Causes of Congestion in IoT Networks

Data congestion in the Internet of Things (IoT) can occur due to several reasons, some of which are:

1. Large Number of Connected Devices: IoT involves a vast number of interconnected devices, each generating and transmitting data. As the number of devices increases, the volume of data being generated and exchanged also rises, leading to potential congestion in the network.

2. Continuous Data Generation: Many IoT devices produce data continuously, especially those involved in real-time monitoring and sensing. This constant flow of data can overwhelm the network's capacity if not appropriately managed.

3. Limited Bandwidth: IoT devices often rely on wireless communication technologies like Wi-Fi, cellular networks, or LPWAN (Low Power Wide Area Network). These communication channels have limited bandwidth, and when numerous devices try to transmit data simultaneously, it can result in congestion.

4. Inefficient Data Handling: If the data generated by IoT devices is not efficiently processed, routed, or aggregated before transmission, it can consume more bandwidth than necessary and exacerbate congestion.

5. Lack of Prioritization: Some IoT applications might not prioritize data packets based on their importance or urgency. As a result, less critical data might clog the network, affecting the transmission of critical data.

6. Interference and Signal Obstruction: In wireless communication, interference from other devices or physical obstructions (walls, buildings, etc.) can disrupt data transmission, leading to retransmissions and congesting the network.

7. Security Measures: Security protocols and encryption used in IoT can add overhead to data packets, increasing the data size and contributing to congestion.

8. Geographic Concentration: In certain scenarios, IoT devices might be highly concentrated in specific geographical areas (e.g., smart cities). This concentration can strain the local communication infrastructure and cause congestion.

9. Malfunctioning Devices: Faulty or misconfigured IoT devices may generate excessive data or continually attempt to transmit data, even when not necessary, further burdening the network.

To mitigate data congestion in IoT, various strategies can be employed, such as optimizing data transmission protocols, improving network infrastructure, using efficient data compression techniques, prioritizing critical data, and enhancing communication technologies to handle a larger number of devices and data efficiently.

1.5. Congestion Handling

Congestion handling is a process that consists of major three steps: congestion prediction/detection, congestion notification, and congestion control. Congestion prediction is responsible to identify the congestion in the network with more accuracy. Congestion prediction is the initial step and further steps increase its output accuracy. The information of the congested node is notified to the sender either by implicit or explicit method. Control action is needed to take care of proper load balance and efficient working of the complete network. Figure. 1.1 shows the various parameters/methods used to perform these actions.

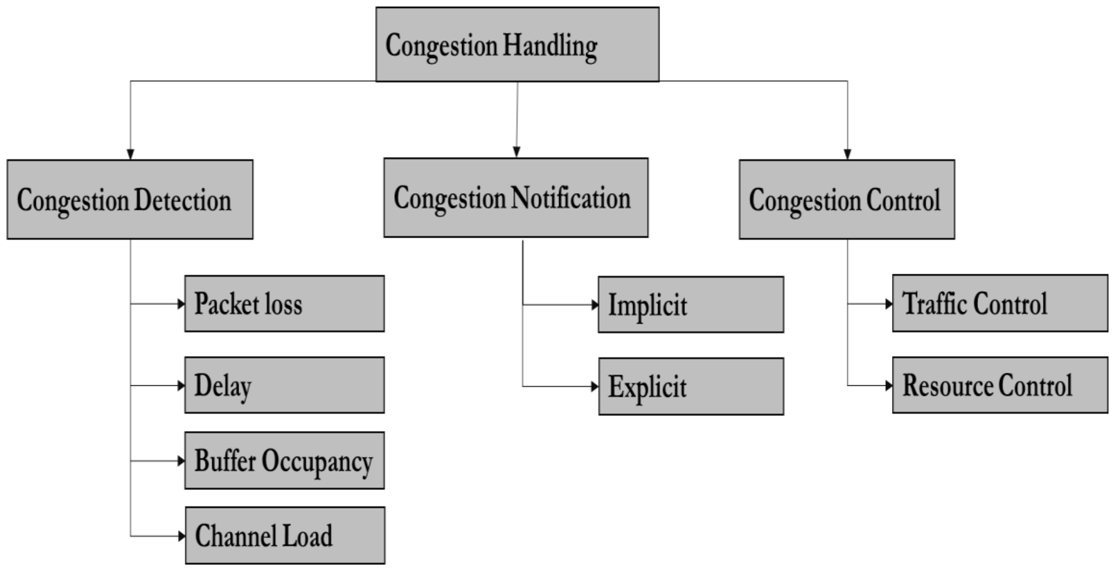

Figure. 1.1: Process of Congestion Handling

1.5.1. Process of Congestion Handling

a) Congestion Prediction

Congestion prediction is done by checking the values of the affecting parameters and if the value is exceeded to threshold, it will consider it congested. These are generally considered parameters for the prediction of congestion.

1. Buffer Occupancy: Every node in the network has a buffer, which controls how much memory it may hold. A node briefly retains packets it receives in its buffer before sending them. Parent nodes receive packets from their kid nodes and, depending on their transmission rate, store them in their buffers. The buffers may fill up and cause packet loss if there is a difference in transmission rates between the child and parent nodes. The Buffer Occupancy Ratio, which is determined by dividing the number of packets in the node's queue (i) by the queue's size, represents the current condition of a node by displaying the percentage of its occupied buffer.

$$BO_R = N_P/Q_x \tag{1.1}$$

where Np represents the packet present in the queue and Qs represent the size of the queue in terms of the maximum packet that may be stored.

2. Expected Transmission Count (ETX): It is the factor that RPL considers while choosing the route. It aids in locating trustworthy, high-quality communication networks. If the value of ETX is little, the connection quality will be higher. RPL uses ETX to differentiate between loss and congested lines.

$$\text{ETX(ij)} = \frac{1}{Pdata(ij) - Pack(ji)} \tag{1.2}$$

Where represents the data packet probability of successful delivery from to and represents the acknowledgement packet successful delivery from *(j)* to *(i)*.

3. Channel Load: It calculates packet load. This number aids in determining the network's lifespan and assists in avoiding nodes with lower energy. One of the nodes is a crucial parameter in a restricted network. The remaining energy is calculated by deducting the node (i) *Eci's* consumed energy from the *IEi's* beginning energy.

$$REi = IEi - Eci \tag{1.3}$$

where, initial energy is *IEi,* energy consumed is *Eci* and the remaining energy of node i is represented by *REi* in the IoT networks. Channel busyness or Channel load ratio refers to the proportion of time intervals during which the channel experiences activity due to either successful collision or transmission, in relation to the overall time.

4. Packet Loss: This approach is implemented when ACK (Acknowledgment) is enabled. In the event that ACK is not received by a sender, it assumes that congestion has occurred. Nonetheless, packet loss may be attributed to wireless errors rather than collisions within the wireless channel.

5. Delay: This indicates the time it takes for a packet to successfully arrive and reach either the endpoint receiver or the next hop receiver after being generated at the sender. Using delay as a sign of congestion, however, might be misinterpreted when a radio duty cycle (RDC) is used at the MAC (Media Access Control) layer and causes significant packet delays.

a) Congestion Notification

To avoid network-wide congestion, it is crucial to alert the source nodes about any congestion that has been discovered or is anticipated for a given set of nodes. It is possible to send this congestion notification directly or implicitly.

1. Implicit Notification: Using this approach, the congestion data is appended to the header of data packets or ACK (Acknowledgment) packets, allowing it to be transmitted alongside the regular payload. By using this approach, extra overhead packets are eliminated, reducing network congestion and preventing future congestion. In wireless networks, this method is used to control congestion.

2. Explicit Notification: In this strategy, crowded nodes broadcast more overhead packets to other nodes to let them know they are congested. By using this method, extra overhead packets are introduced into the network, which worsens the congestion issue.

b) Congestion Control

Congestion control is the step taken after the congestion information is received from the source nodes so that further steps are taken to mitigate and reduce network congestion. There are two ways to reduce and handle IoT network data congestion that may be managed through traffic control where rate adjustment of data transmission in performed, resource control involving the selection of non-congested paths for packet forwarding, and packet offloading from congested to uncongested nodes.

1. Traffic Control: Traffic control is a method that may be used to manage network congestion. According to this method, source nodes transmit fewer packets into the network, which decreases the sending rate to a predetermined level. The window-based method and the rate-based method are two frequently employed strategies for traffic rate adaptation. In the window-based approach, a source node gradually widens its congestion window to determine the bandwidth that is available. The congestion window is severely reduced if congestion is found. The additive increase multiplicative decline (AIMD) mechanism, which includes linearly expanding the congestion window and exponentially contracting it if congestion is reached, is an illustration of this strategy. The rate-based system, on the other hand, uses source nodes to estimate and validate the network conditions and available bandwidth. They modify their transmission rate in accordance with the predicted amount of available bandwidth. However, slowing down the rate at which significant data is carried is neither practicable nor desirable when working with event-based and time-sensitive applications, because packets include important information that must be provided instantly.

2. Resource Control: It is a different approach used to solve the shortcomings of traffic control. When there is congestion, resource management permits packets to be routed to target nodes through additional uncongested channels rather than lowering the sending rate. This strategy guarantees that the transmitting rate won't change. As a result, compared to the traffic management technique, the packet delivery ratio employing resource control is greater. Offloading is a type of resource control approach that, in contrast to other resource control methods, takes into account the packets that are waiting in the queue of congested nodes. In comparison to traffic control methods, it provides load balancing so that data packets may be transported efficiently and arrive at their destination with a greater packet delivery ratio.

1.5.2. Protocol for Congestion Handling in IoT Networks

The Internet of Things (IoT) protocol structure is still evolving, and there isn't a complete standardization across all aspects. However, the protocol stack is generally divided into five layers, which are as follows:

Physical Layer: This is the lowest layer of the IoT protocol stack and is responsible for connecting and coordinating multiple smart devices. These devices often have resource constraints, such as limited processing power, memory, and energy. They are typically designed to work with lossy and low-power networks. The physical layer deals with the actual transmission and reception of data over the communication medium.

Data Link Layer: The Data Link Layer, situated above the Physical Layer, facilitates communication between devices in the IoT network. It handles data frame encapsulation, error detection, and flow control, ensuring reliable and efficient data transmission. This layer plays a crucial role in coordinating data exchange and managing network connectivity, especially in resource-constrained environments with lossy and low-power networks.

Network Layer: The network layer is responsible for handling the routing of data within the IoT network. It manages the paths that data takes from the source device to the destination device.

In IoT networks, the Routing Protocol for Low-Power and Lossy Networks (RPL) is a standardized routing protocol introduced in 2012. RPL is designed to efficiently route data in networks with resource-constrained devices. IPv6 addressing method is commonly used to uniquely identify devices within the IoT network.

Transport Layer: The transport layer is responsible for ensuring reliable data delivery and managing end-to-end communication between devices. It offers two main transport protocols TCP and UDP.

TCP (Transmission Control Protocol): TCP provides reliable and ordered delivery of data, making it suitable for applications that require error-free data transmission. UDP (User Datagram Protocol): UDP offers a connectionless and unreliable form of data transmission, which is suitable for applications where speed and low overhead are more important than error recovery.

Application Layer: The top layer of the IoT protocol stack is the application layer, which contains various protocols for facilitating specific functions and services for IoT applications. Two popular application layer protocols used in constrained IoT are CoAP and MQTT.

CoAP (Constrained Application Protocol): CoAP is a lightweight application layer protocol designed for resource-constrained devices and networks. It is designed to provide similar functionality to HTTP (Hypertext Transfer Protocol) but with lower overhead, making it well-suited for IoT devices. And MQTT (Message Queuing Telemetry Transport): MQTT is a publish-subscribe messaging protocol that facilitates efficient communication between devices in IoT networks. It is designed to be lightweight and efficient, making it ideal for low-power and constrained environments.

Overall, the IoT protocol stack is structured to accommodate the specific needs and constraints of IoT devices and networks, enabling efficient and reliable communication between connected devices. While some protocols have been standardized, the IoT landscape is continually evolving, and new protocols may emerge or existing ones may evolve further to meet the growing demands of the IoT ecosystem.

Table 1.1: IoT Protocol Stack

Layer	**Description**	**Protocols and Standards**
Application Layer	Enables communication between IoT devices and applications. It defines the data formats, protocols, and services used by IoT applications.	HTTP, CoAP, MQTT, XMPP
Transport Layer	Responsible for end-to-end communication and data delivery reliability.	TCP, UDP, SCTP
Network Layer	Handles the routing of data packets across the network. It ensures that data reaches its intended destination.	IPv6, 6LoWPAN, RPL
Link Layer	Deals with the transmission of data over the physical medium (wired or wireless).	IEEE 802.15.4 (for low-power wireless), Ethernet, Zigbee, Bluetooth
Physical Layer	Represents the physical hardware and transmission medium, defining how bits are transmitted and received.	Wi-Fi, Bluetooth, Zigbee, LoRa, Cellular (LTE, NB-IoT)

Congestion control in IoT networks with constrained devices is most needed. For this reason, the CoAP protocol includes a basic congestion control mechanism. Another method of controlling congestion which is focused on by various authors is designing suitable mechanism for scheduling or routing the data packets. RPL is widely recognized as the standard routing protocol for IoT networks. Both CoAP and RPL are explained.

1. Constrained Application Protocol (CoAP)

In IoT networks simple TCP or UDP-based congestion control mechanisms will not work effectively due to their various constraints and higher amount of traffic. IoT networks use COAP at the application layer due to the constraints in the network. It is a web-based transfer protocol having the functionality of HTTP however, it is not the replacement for HTTP. When the communication is between constraint devices COAP is used. However, communication between resourced devices is using HTTP (COAP and HTTP mix network). COAP works over UDP/TCP to provide congestion-handling mechanism which is explained below section. Traditional COAP-based congestion control is based on the traffic control mechanism. COAP is defined to support IoT devices by introducing lightweight messages. It is developed by the Internet Engineering Task Force. It is based on the concept of representational State Transfer architecture (REST). There are publish/subscribe and client/server forms of iteration. The interaction in the client/server model may one-to-one or multicast where several servers are interrogated by a client using requests. In the case of publish/subscribe the role of subscriber and publisher is played by the observer and server respectively [39]. Notification messages are sent to them by the server. COAP work in a connectionless communication path client-server interaction because COAP works over UDP. For addressing COAP allowed to use UDP broadcast and multicast. In case of UDP basic verification and error check may be done due to which it becomes suitable for IoT networks [49]. COAP uses four message types i.e., conformable, non-confirmable, and reset acknowledgement. For reliability COAP uses confirmable and non-confirmable messages [50]. If the message type is confirmable then acknowledgement is sent by the receiver otherwise sender will retransmit the packet. However, in the case of non-confirmable packet, there is no need for acknowledgement and it works in send and forget manner.

COAP need a proper congestion control method for itself as it works on UDP. Unlike

HTTP which works upon TCP have end to end mechanism to handle the congestion. In the case of confirmable messages, basic congestion control is performed by using a fixed RTO value [51]. A random value is set initially for RTO, the range of random value is between ACK_TIMEOUT and ACK_RANDOM_FACTOR multiplied with ACK_TIMEOUT [52]. Retransmission is done acknowledgement is not received within fixed RTO, the value of RTO will be doubled. And COAP also set the number of maximum retransmissions that may be done by the sender. The value of max transmission is set to 4 in the basic COAP congestion control approach. However, the approach is not considered a good mechanism for congestion control because it doesn't have the capability to adapt the network condition [53]. The basic approach doesn't consider the value of previous RTT which makes it difficult to select optimal RTO value. If the value of the RTO is lesser than the RTT it results in false retransmission and increases the unnecessary traffic. Or if the value of RTO is higher than RTT, it will cause long waiting time before retransmission for the lost packet. This increases the delay in the transmission and network communication.

2. Low Power and Lossy Routing Algorithm (RPL)

As instructed by the IETF, the ROLL group released the RPL (Routing Protocol for Low-Power and Lossy Networks) protocol in a 2012 RFC [54]. Since then, several new RFCs have been released to offer more comprehensive details regarding the core elements of RPL, including objective functions [55, 56], routing metrics [57], and the Trickle timer [58]. Specifically created for low-power and lossy networks, RPL operates at the network layer and supports a number of data link layer protocols, including IEEE 802.15.4.

By constructing a Destination-Oriented Directed Acyclic Graph (DODAG), RPL creates multi-hop paths through intermediary nodes from leaf nodes to a root node. As its next stop on the road to the root, each node chooses a collection of possible nodes.

The following three control messages are utilized:

DIO (DODAG Information Object): This multicast message contains DODAG information, including the DODAG ID, node rank, and RPL instance.

DIS (DODAG Information Solicitation): Nodes send this message to request their neighbours to transmit a DIO message.

DAO (Destination Advertisement Object): Used to propagate destination information upwards within the DODAG.

To construct the DODAG, the root node first sends the DIO message to all neighbouring nodes within its communication range. Each neighbour decides whether or not to participate in the DODAG based on the aim function of the RPL. The DIO message is then retransmitted by the neighbour with the root designated as its parent node. To create a network-wide tree-like structure, all nodes continue to work toward the same objective. When a node decides to join the DODAG, it adds the DIO sender's address to its candidate parent list.

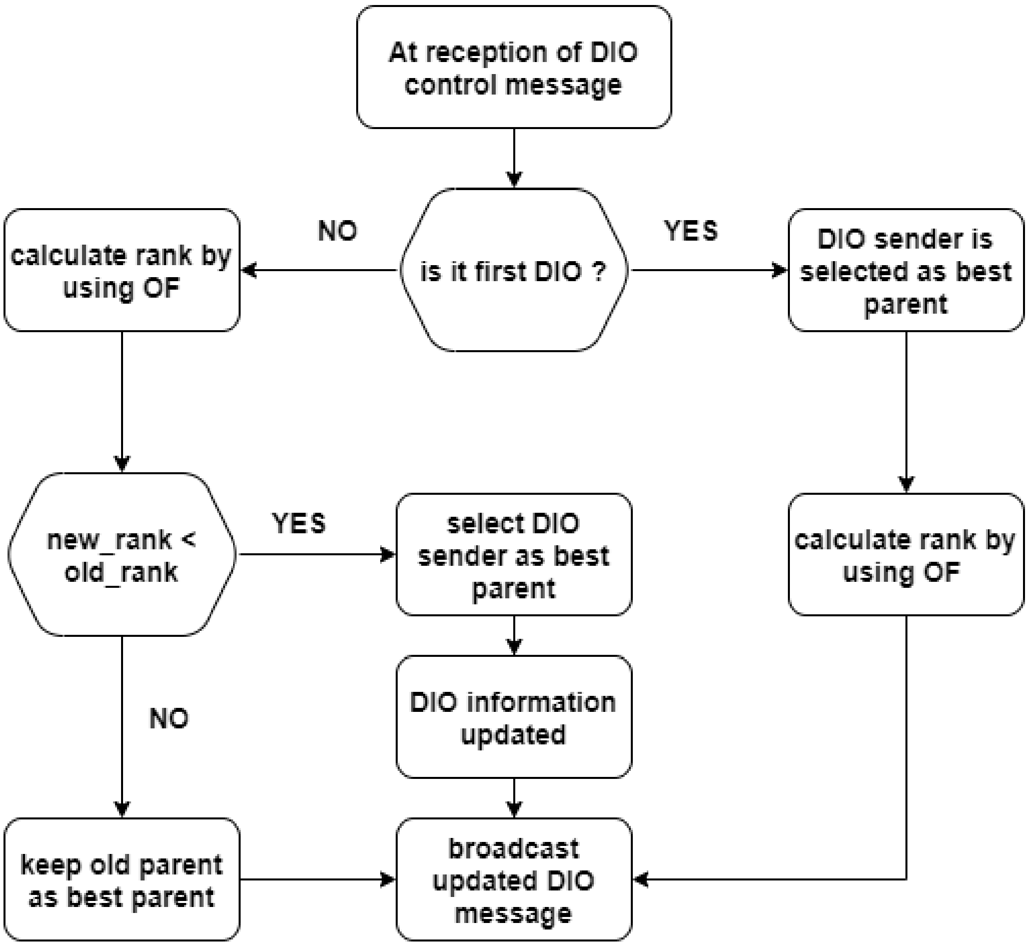
At reception of DIO control message
is it first DIO ?
NO
YES
calculate rank by using OF
DIO sender is selected as best parent
new_rank < old_rank
YES
select DIO sender as best parent
calculate rank by using OF
NO
DIO information updated
keep old parent as best parent
broadcast updated DIO message

Figure.1.2. Flowchart of RPL

The node determines its rank in a way that ensures it is higher than any other specified parent nodes and delivers the updated information within the DODAG. Nodes in the DODAG determine whether to retain or increase their current rank after obtaining this information. Lower-ranking parent nodes are deleted to prevent routing loops. If a node wishes to join the DODAG but hasn't heard from its neighbours in a particular length of time, it may send them a DIS message to request DIO transmission. The procedure of determining the optimum parent node based on n rank calculation and DODAG building is shown in the flowchart in Figure. 1.2.

1.6. Motivation:

Among the various challenges and issues faced by IoT networks, congestion stands out as a critical problem that significantly impacts the effectiveness and quality of IoT services. The motivation for selecting the problem of congestion handling in IoT networks for this thesis is rooted in the following key factors:

Impact on IoT Applications:

IoT networks are extensively used in diverse domains, including smart hospitals, smart cities, and driverless vehicles. These applications rely on the seamless functioning of IoT networks to deliver real-time data and services. However, with the rapid growth of IoT devices and the limited availability of network resources, congestion becomes a prevalent issue that affects the performance and reliability of these applications. Addressing congestion through effective control mechanisms is crucial to ensure the consistent delivery of IoT services.

Quality of Service Enhancement:

Congestion in IoT networks leads to increased delays, packet loss, and degraded network performance. These factors directly affect the quality of service experienced by end-users. To provide a seamless and satisfactory user experience, it is essential to develop congestion-handling techniques that may alleviate network congestion, reduce latency, and maintains reliable data transmission. Improving the quality of service is a primary motivation for investigating congestion control mechanisms in IoT networks.

Resource Utilization and Efficiency:

IoT networks often operate with limited resources, including bandwidth, processing power, and battery life. Congestion exacerbates resource constraints, resulting in inefficient utilization of these scarce resources. By implementing effective congestion control mechanisms, it is possible to optimize resource allocation, minimize wastage, and enhance the overall efficiency of IoT networks. This optimization may lead to improved network performance, reduced energy consumption, and extended device battery life.

Scalability and Network Expansion:

The growth of IoT networks is expected to continue exponentially, with billions of devices connecting to the network. As the network scales, congestion handling becomes increasingly challenging. Traditional congestion control mechanisms designed for conventional networks may not be suitable for IoT environments due to the unique characteristics and constraints of IoT devices. Therefore, developing scalable congestion-handling techniques tailored specifically to IoT networks is crucial to accommodate the increasing number of devices and sustain the growth of the IoT ecosystem.

Security and Reliability Considerations:

Congestion may also have implications for the security and reliability of IoT networks. It may be exploited by malicious actors to launch attacks or disrupt network operations. Additionally, congestion-induced delays and packet loss may impact critical IoT applications that rely on real-time data transmission, such as healthcare monitoring or autonomous systems. By addressing congestion handling, this research aims to enhance the security and reliability of IoT networks, ensuring the uninterrupted and secure operation of IoT applications.

The selection of the problem of congestion handling in IoT networks for this thesis is motivated by the significant impact congestion has on IoT applications, the need to enhance the quality of service, improve resource utilization and efficiency, address scalability challenges, and ensure security and reliability. By investigating effective congestion control mechanisms, this research aims to contribute to the seamless operation, performance optimization, and sustainable growth of IoT networks, ultimately benefiting a wide range of IoT applications and their users.

1.7. Problem Statement and Objectives:

This thesis focused on the problem of congestion handling in IoT networks that causes delayed packet delivery and high packet loss. To address congestion detection/prediction and congestion control, methods are proposed in this thesis. As the number of IoT devices rapidly increases, and network resources become limited, congestion becomes a critical problem leading to delays, packet loss, performance degradation, and compromised reliability. Existing congestion control mechanisms designed for traditional networks are not well-suited to handle the unique characteristics and constraints of IoT environments. Factors such as the mobility of IoT devices, lack of standardized protocols, and diverse system architectures further complicate congestion handling in IoT networks.

The scalability of congestion-handling mechanisms becomes even more crucial as IoT networks experience exponential growth in the number of devices and data traffic. Congestion problems need to be addressed to enhance the delivery rate and reduce packet loss and delay in IoT networks. It is also important for critical IoT networks where resources are limited or need timely delivery

Thus, there is an urgent need to develop efficient and scalable congestion-handling mechanisms specifically tailored to the unique requirements of IoT environments. These mechanisms must consider factors like mobile devices, limited resources, real-time data transmission, and scalability to address congestion-related issues and ensure the smooth operation of IoT networks.

The primary objective of this thesis is to investigate and propose novel congestion-handling techniques that have effectively tackled the challenges posed by congestion in IoT networks. These techniques should aim to minimize delays, packet loss, and network performance degradation while optimizing resource utilization and scalability. Through the development of such effective congestion control mechanisms, this research seeks to provide the congestion handling methods at different stages and provide performance enhancement of IoT networks by the reduction of packet loss and delay.

Based on the above problem statement, we have considered the following objectives:

1. To design an approach for congestion prediction in IoT networks by considering multiple parameters such as congestion window, throughput, propagation delay, round-trip time (RTT), number of packets sent, and packet loss.
2. To design an approach for congestion control by performing data offloading.
3. To design an approach for congestion-aware data transmission in IoT networks.

4. To perform the comparative analysis of our proposed approach with the existing model.

1.8. Simulation Platforms

In this thesis, we performed simulations of an IoT network to handle the problem of data congestion. To achieve this, we utilized two primary platforms, namely Python and Cooja.

Python, being a widely-used and powerful programming language, allowed us to develop simulation scripts for IoT devices, network protocols, and data transmission scenarios. It was the best tool to implement machine learning algorithms as it had various supportive libraries. With Python, we efficiently implemented congestion prediction and control mechanisms and easily adjusted simulation parameters for numerous experiments.

On the other hand, Cooja, a specialized network simulator for IoT and Wireless Sensor Networks (WSNs), provided an accurate representation of the IoT environment. It allowed us to model the physical characteristics of IoT devices, and observe real-time data flow, packet collisions, and network load, all of which were crucial for analyzing and validating our congestion prevention strategies. The use of Python and Cooja for different approaches to congestion handling offered a comprehensive and reliable framework for our research on managing data congestion in IoT networks.

1.9. Organization of Thesis

This thesis focuses on conducting a performance analysis of composite fading channels. The work is organized into distinct chapters, and the contributions of each chapter are summarized as follows:

Chapter 1: Introduction

In this chapter, we have introduced the IoT, congestion handling and its process in IoT followed by the motivation and problem statement and also briefly about various chapters of the thesis.

Chapter 2: Literature Survey

In this chapter, thesis presents a comprehensive and thorough literature survey on the diverse approaches and techniques utilized for congestion handling in IoT networks across various layers and levels.

Chapter 3: Congestion Prediction in IoT Networks

This chapter explains the proposed approach for congestion prediction in detail along with its results as compared with other approaches.

Chapter 4: Data Congestion control using offloading in IoT network

This chapter explains the approach to control congestion by offloading data packets to reduce packet loss and delay in the IoT networks by modifying the RPL protocol.

Chapter 5: Congestion-Aware Data Transmission in IoT Networks

This chapter explains the approach to congestion-aware data transmission in IoT networks. For this objective, the hop-to-hop data transmission approach is designed where the next hop selection considers several parameters to provide congestion-aware and effective data transmission.

Chapter 6: Conclusion and Future Scope

This chapter presents conclusions about the proposed methods and algorithms and a detailed discussion of potential areas for future work.

CHAPTER- 2

LITERATURE SURVEY

In this chapter survey of various methods and techniques from the state-of-art are explained that work in the field of IoT-based data congestion handling. The survey is structured into three main sections. The first and second section covers the approaches of congestion handling at the application layer and network layer, with a focus on different versions of the CoAP and RPL protocol. The third section delves into congestion handling by performing data packet offloading techniques.

2.1. Application Layer Congestion Control Protocol

Congestion control in the application layer involves managing the traffic rate in IoT networks. These approaches, also known as traffic control approaches, aim to handle congestion in IoT networks. The standard application protocol used in IoT, CoAP (Constrained Application Protocol), primarily operates over UDP. Various researchers have shown interest in developing congestion control mechanisms for CoAP.

In one study [59], the author provides a comprehensive examination of congestion control systems for CoAP. Many works in this survey focus on modifying the estimation process of Retransmission Time Out (RTO) to improve the performance of CoAP/CoCoA (Congestion Control for CoAP). However, the detection of congestion signals from noisy Round Trip Time (RTT) samples is not discussed extensively. CoAP/CoCoA is compared to other approaches in several studies [59-61]. Some studies consider CoCoA to be superior to CoAP, while others mention that CoCoA increases retransmissions when the request count is high [62].

In another paper [63], the RTO value is calculated based on the Eifel retransmission timer, which was initially proposed for TCP. The author replaces the coefficients α, β, and K from RFC 6298 with a single coefficient γ, making it suitable for a large sender load. However, the extraction of congestion signals from RTT samples is not addressed.

In [65], four modifications of CoCoA are discussed: CoCoA-F, CoCoA-S, and

CoCoA-4-Strongs. These modifications aim to improve the performance of CoCoA in lossy wireless links and address the side effects of weak estimators. CoCoA-F enhances competitiveness by reducing the values of the backoff threshold, maximum RTO, and initial values. It performs conservatively, similar to CoCoA. The author also proposes four-state higher granularity estimators to differentiate between wireless link losses and congestion losses. This work enhances the performance of CoCoA in lossy wireless networks. CoCoA-4-State-Strong increases throughput by 30-60% and adapts to packet losses, but it leads to an increased retransmission rate of around 20%.

In [66], an adaptive mechanism is proposed to handle congestion in CoAP by considering packet loss rate and traffic priority. The mechanism comprises three main components: congestion prioritization, optimized RTO, and return timers. Traffic priorities are assigned based on the type of equipment, and the value of RTO backoff is determined according to traffic priority and packet loss rate. However, the approach is not implemented, raising questions about its feasibility.

Another adaptive congestion control mechanism is presented in [67], suitable for highly congested networks where a single packet requires multiple retransmissions. This mechanism precisely selects the value of RTT for retransmitted packets, leading to increased successful transactions and throughput. However, it may not be efficient when multiple retransmissions are not needed.

In [68], a precise CoCoA-based approach is proposed, addressing limitations of the previous version (CoCoA+), such as the close proximity of RTO and RTT values and weak estimator weights and updates. The performance of CoCoA in burst traffic patterns is analyzed, but the results are not promising. The paper introduces fixed values that may not be suitable for practical scenarios in large IoT networks.

The problem of queuing delay, known as buffer bloat, is addressed in [69]. Various congestion control mechanisms, including CoAP and CoCoA, fail to handle buffer bloat, resulting in unnecessary retransmissions and wasted network bandwidth. The author identifies the issue as the inaccuracy of RTO's backoff logic and proposes a new logic to overcome buffer bloat.

In [70], the issue of buffer bloat in heavily congested networks is addressed through the suggestion of Fast-Slow RTO. This mechanism determines if packet losses are due to wireless link loss or congestion by employing slow and quick RTO computation and self-adaptive retransmission timers. The results show that Fast-Slow RTO achieves shorter Flow Completion Time compared to CoCoA and CoAP and effectively controls RTO in heavily congested traffic.

The evaluation of CoAP congestion control using Wishful in real-time is conducted in [71]. Wishful is a large-scale platform for runtime experiments and network design. The CoAP congestion control mechanism is compared to simple RTT-based algorithms, and the results indicate that the default congestion control mechanism is more robust and extensible than simple RTT-based algorithms.

In [18], Rathod et al. presents a congestion control scheme based on delay gradient for IoT networks. They propose a novel congestion control algorithm called CoCoA++ that utilizes delay gradients and probabilistic backoff, integrated with CoAP. The proposed approach is implemented in the Cooja network simulator, and it achieves better packet-sending rates and reduced delays.

These various approaches and algorithms aim to improve congestion control in CoAP-based IoT networks by considering factors such as traffic priority, packet loss rate, buffer bloat, delay gradients, and retransmission timers. The goal is to achieve better network performance, higher throughput, and reduced delays in packet transmission. All the above approaches which are discussed in this section are designed for CoAP congestion control mechanism. RTO and RTT begin important factors whose values are responsible to handle and avoid congestion.

2.2. Network Layer Congestion Control Protocol

Congestion is a significant issue in multi-hop routing, as it leads to node-level congestion due to the accumulation of data with an increase in the number of hops. This problem is exacerbated in scenarios where a large number of devices transmit data at high rates, resulting in both node-level and channel-level congestion [72, 73]. Congestion negatively impacts the reliability of the network, causing delays, packet loss, and high energy consumption [74]. To mitigate congestion, traffic flow control, traffic rerouting, and load balancing techniques are employed.

In the context of 6LoWPAN, congestion analysis reveals that packet loss is primarily caused by buffer overloading rather than channel loss [75]. To address this issue, [76] proposes a congestion control mechanism based on the IPv6-based routing protocol

RPL for low-power and lossy networks. The proposed mechanism introduces two metrics, buffer occupancy and a congestion-aware objective function, resulting in improved throughput and packet delivery ratio.

On the other hand, [77] suggests measuring congestion control using buffer occupancy, which incurs additional overhead to transmit buffer occupancy information. During parent selection in the routing process, packet losses are frequently observed in high-traffic scenarios due to congestion and load balancing challenges, as mentioned in [78, 79]. To tackle this problem, [84] proposes a method called Queue Usage-based RPL (QU-RPL), where the selection of the parent node takes into account the utilization of neighbouring nodes and their hop distance from the RPL router. QU-RPL effectively reduces high queue losses and improves the packet delivery ratio.

Efficiently regulating transmission energy and route topology in wireless networks is crucial for achieving good bandwidth and reliability [81]. Power-controlled RPL is proposed as a solution to handle fluctuations in transmission power and prevent the design of routing topologies that cause bandwidth loss. This approach effectively manages both routing topology and transmission power.

In Dense IoT Networks, an energy-efficient load-balancing scheme is presented by Farahani and Rahbar [118]. Their scheme aims to enhance the routing performance of IoT networks.

Another approach, proposed in [80], tackles congestion by utilizing the concept of duty cycles. The Duty Cycle-based Congestion Control for 6LoWPAN (DCCC6) adjusts network traffic based on buffer occupancy and Radio Duty Cycling (RDC), while the routing part is managed by RPL. Experimental testing with 25 randomly deployed nodes demonstrates improved performance in terms of energy consumption and delay.

In [81], authors propose the deaf, griping, and fuse congestion control schemes. These schemes employ buffer length and queue length as congestion indicators. The fuse scheme combines both buffer length and queue length, outperforming griping and deaf in congestion control.

These various approaches and schemes contribute to addressing congestion issues in IoT networks, utilizing concepts such as buffer occupancy, congestion-aware metrics, power control, load balancing, and duty cycles. Their objective is to enhance network performance, reduce energy consumption, and manage congestion effectively.

The priority of a node or application was taken into consideration by the authors of [82] when they suggested a theoretical method for congestion control. When creating the game, they took into consideration buffer, energy, and priority in order to predict the adaptive transmission rate for sensor nodes. Improved performance was seen in the simulation results in terms of throughput, energy use, and latency. The authors then provided a method based on resource control in [83]. By tracking buffer occupancy, it finds the least crowded route. This method works well in times of network congestion and is ideally suited for RPL/COAP-based networks. Its usage of "eavesdropping" in the algorithm causes it to spend a lot of energy on non-congested networks when packets are passively listened to by nodes. Load balancing is another tactic mentioned in [84] for reducing congestion. Based on queue occupancy, nodes use DIO messages to notify their offspring nodes when there is congestion. When there is network congestion, this strategy improves performance by regulating data flow inside the network.

The game theory-based technique described in [85, 86] is specially made to alleviate congestion by choosing a new parent node for a node that experiences congestion. The child node receives this information from the parent node, allowing the child node to proactively modify its parent node. This guarantees dependable network communication. This method considerably improves the total network performance as compared to native RPL, showing a phenomenal throughput boost of 100%. On the other hand, [87, 88] provides a different method of load balancing by choosing several parent nodes. By selecting many channels for data transmission, this technique spreads out the burden. The outcome is an increase in throughput and energy efficiency while assisting in the avoidance of congestion. However, putting this strategy into practice necessitates changing RPL standards and the DODAG (Destination Oriented Directed Acyclic Graph) construction process, which might cause compatibility problems with native RPL. [89] uses the implementation of multipath routing to address the network congestion issue. This method makes use of several channels for data transmission, allowing for effective data flow and lowering the possibility of congestion-related problems. In IoT networks, multipath routing has shown to be a successful method for reducing congestion.

A multipath technique is used to transfer the data, with an objective function being used to determine the best path. A DOI message is used to alert and start the multipath operation in the event of congestion. The grey theory is used as a foundation for congestion control in [90], which takes into account several variables for optimization, including buffer occupancy, latency, and expected transmission count. Utility functions are used to increase throughput in this approach to integrated traffic and resource control.

In [91], the authors put forth a technique that effectively balances the load while consuming less energy by using a probability measure in traffic forwarding to destination parents. With the help of a cutting-edge tool called Expected Lifetime (ELT), which locates possible bottlenecks in the system, the administration of broadcast is carefully monitored. The lifespan measure, which ensures parents with larger ELT values rank lower in the selection process and takes into account the lifetime of possible bottlenecks, is vital throughout the parent selection process. This leads to a balanced topology where many parents share the responsibility of raising their kids. Through parent selection, the ELT values and regressing traffic weight of the parents are established. The outcomes show an overall improvement in load balancing and network longevity. However, there is a chance that fragmentation will still happen, necessitating algorithm improvements. The extension suggested in [92] more fluidly handles network convergence and associated instabilities.

In [93], the issue of network congestion is addressed with M-RPL (Multipath RPL), an extension of RPL that employs a two-pronged approach. The first prong involves congestion detection using PDR measures to identify congested paths. If congestion is detected, the second prong, congestion avoidance, reduces forwarding to the congested node and utilizes alternative paths for routing traffic. Although this approach adds extra processing overhead due to the two prongs, M-RPL significantly improves overall throughput by effectively reducing congestion compared to standard RPL.

In [92], an enhancement to RPL's objective function called LB-OF is proposed to address load balancing in the network involving bottleneck nodes. This enhancement disperses the children of bottleneck nodes to other parents with the same children count, utilizing a new metric named CNC (Child Node Count). Nodes with lower ranks prioritize accepting new children, while higher-ranked nodes are more reluctant.

The results demonstrate improved network lifetime and effective load balancing, although increased power consumption and frequent parent changes may introduce network instability.

The author [94] introduces a smart grid-based approach to enhance RPL, leading to better load balancing in the network and termed Objective Function for Quality of Service (OFQS). This proposal incorporates latency, link quality, and residual energy into a new metric called OFQS. The approach is derived from MRHOF while introducing thresholds to stabilize routes by reducing frequent parent changes for nodes. The load balancing achieved enables traffic to utilize less reliable but shorter routes, resulting in improved network lifetime. Although the experiment was conducted with a limited node count, its extension to a higher node count is explored in [95].

The authors in [96] proposed a new scheme called Multi-gateway Load Balancing Scheme for Equilibrium (MLEq), drawing inspiration from the behaviour of flowing water. This scheme efficiently balances the load in a distributed and dynamic manner. Networks with multiple DODAGs adopt MLEq by implementing a rank parameter called Virtual Level (VL) metric, similar to the one used in DODAGs. The approach mitigates congestion caused by message traffic by identifying high VL values, indicating high traffic, and shifting the overloaded DODAG junctions to areas with lower message traffic. The VL metric is transmitted as a multicast using special messages called VL Information Objects (VIO) to every neighbour. Each node calculates its VL based on the VIO message received with the shortest hop distance. The DODAGs' VL metric determines the final node topology for achieving load balancing. However, this scheme consumes more energy than standard RPL due to the routing of special VIO messages.

In [97], the authors propose an extension to RPL called the Heuristic Load Distribution Algorithm, which is a multipath enhancement based on braided multipath concept [98]. It employs two mechanisms: multipath routing, where nodes have multiple parents simultaneously, and Tangential Load Balancing, which balances energy consumption in the network. This combination results in significantly improved throughput, enhanced lifetime, and smoother load balancing. However, it may not exhibit the same level of improvement in real-life scenarios where the network topology is heterogeneous.

In [99], a different method for multipath routing called Load-Balanced Data Collection through Opportunistic Routing (ORPL-LB) is proposed. Unlike traditional mechanisms that pre-determine the packet's route, ORPL-LB dynamically selects the next hop based on the availability of nodes in the route, effectively avoiding congestion in paths. Opportunistic routing takes on the responsibility of load balancing in ORPL-LB, employing a sleep/wake-up idea to select the next hop for the packet. Nodes in the sleep state, experiencing high traffic or low energy, are avoided as potential candidates. The results show promising reductions in node duty cycles without negatively impacting packet delivery success or delivery delays.

In [100], the authors introduce "Energy-Aware and Load Balanced Parent Selection" to dynamically select parents, reducing energy consumption and achieving effective load balancing among nodes. They modify the IEEE 802.15.4 standard's topology, the cluster-tree MAC, to distribute traffic more uniformly. This modification remains compatible with RPL and allows the selection of multiple parents. Each node transmits the packet to a parent node selected based on two factors: the parent node's residual energy and the recently experienced load in the parent path. This novel cluster-tree MAC approach extends the network's lifetime, reduces end-to-end delays, and improves the successful packet delivery ratio, ultimately enhancing the network's overall performance.

Authors in [101] present an approach called Minimum Degree RPL, which incorporates RPL with better load balancing using spanning trees of minimum degree. By employing such spanning trees, the resulting network becomes broader rather than taller, leading to reduced network congestion. The approach comprises four stages: determination of the tree's highest degree node, high-degree nodes seeking alternate edges to nodes with lower degrees, optional handling of multiple alternative edges for a node, and swapping nodes to ultimately reduce the highest degree node's degree. This approach yields promising results, reducing energy consumption in the network by 15.6% and increasing network lifetime.

In [102], the authors propose the Load Balanced Routing for RPL (LB-RPL) method, which addresses the issue of load imbalance in a reliable and decentralized manner. The model quantifies the limited resources of nodes analytically and identifies the count of nodes sending packets as a crucial factor affecting the delivery rate or successful delivery ratio of packets. The model follows a dual-goal approach:

determining the level of load imbalance in the network by monitoring buffer utilization and deferring DIO message transmission to reduce congestion and workload on heavily loaded buffer and forwarding data in a load-balanced manner by identifying less congested parents. This model results in reduced packet loss, lower latencies, and a more even spread of workload throughout the network.

In [103], authors propose an AI and machine learning-based approach called Load Balanced Optimization based on Q Learning (LBO-QL) that uses Q Learning to check the count of children for a parent node, aiming to balance the network. Each node only keeps information about its immediate parents to reduce overhead. LBO-QL experiences quicker convergence and fewer control messages compared to standard RPL. However, its scalability is limited due to the dependence on the network hub for calculations.

In [104], a new protocol called Fuzzy Logic-based Energy-Aware (FLEA) RPL is introduced to achieve a better distribution of energy and load among nodes. Unlike other fuzzy logic-based enhancements to RPL, FLEA-RPL specifically addresses load balancing. It employs linguistic variables for three selected routing metrics (ETX, Load, and Residual Energy) and calculates a "Quality" measure based on fuzzification rules. This quality measure is used in selecting the parent node. The results demonstrate an improved network lifetime, increased success in packet delivery, and more evenly distributed residual energy among nodes, enhancing overall load balancing in the network. However, the method lacks mobility support and may reduce network stability.

In [107], congestion control is achieved by offloading data packets to neighbouring nodes, alleviating congestion on overloaded nodes. The selection of the neighbour node is determined using a mathematical model to predict the congestion level of in-range nodes.

Moving on to [108], congestion prediction in the 6LowPAN network is focused on utilizing a hybrid approach of resource control and traffic control. The authors propose selecting the parent node based on congestion conditions, and analyzing factors like buffer occupancy, ETX, and delay. They calculate the node's congestion value using grey relational analysis and control the transmission rate using Lagrange multipliers and KTT conditions. The approach shows superior performance in terms of throughput, network lifetime, packet loss, and delay compared to QU-RPL and DCCC6. However, it does not consider the impact of neighbours or mobility in the network.

In [109], authors propose a fuzzy logic-based parent node selection approach for congestion avoidance, emphasizing optimal route selection for data transmission. They consider route status, transmission count, and buffer occupancy in the selection process and use a fuzzy weighted sum model for decision-making with multiple parameters. The model dynamically switches routes by checking congestion status and selecting non-congested routes, outperforming QU-RPL and OHCA by increasing network throughput and reducing packet loss. Nevertheless, the approach does not take mobility into account, which is common in real-time IoT networks.

It is noted that load balancing is a key method for avoiding congestion in the network. Multipath routing is also utilized to distribute data through multiple paths or provide alternative paths to handle congestion situations. Additionally, the process of selecting a new parent node when the old parent node becomes congested plays a crucial role in ensuring efficient network management based on various parameters reflecting the new parent node's congestion state.

The performance of all the surveyed approaches for congestion control is examined based on the network throughput, packet loss rate, latency, and the network lifetime.

Table 2.1: Congestion Control Approaches

Ref. No.	**Parameter/Concept used**	**Benefits**	**Limitations**
78	The concept of the Duty cycle is used. Control traffic transmission	Reduce delay Improve energy efficiency	Does not support mobility Reduce throughput Does not use uncongested Node
79	Buffer occupancy.	Improve energy efficiency and packet delivery ratio	Does not support mobility Reduce throughput Does not use uncongested Node

80	Adaptive transmission rate	Reduce delay, improve packet delivery ratio and throughput Support priority of the packet	Does not support mobility Does not use uncongested node
81	Bird flocking technique	Improve energy ratio	Does not support mobility Consumes more energy
83	Queue occupancy	Improve energy efficiency and packet delivery ratio Better load balancing	Does not support mobility Does not use uncongested Node
84,85	Queue occupancy Game theory to find non-congested path	Improve throughput and packet delivery ratio	Does not support mobility Increase energy consumption
86,87, 88	Find multiple parent node	Improve throughput and energy	Does not support mobility Does not follow RPL Standards
89	Multipath routing	Improve throughput and energy efficiency Perform better load balancing	Does not support mobility
90,91	Adaptive multipath routing	Enhance Throughput, delay and energy efficiency. Perform load	Does not support mobility and Increases computation overhead

		balancing	
92	ETX Residual Energy Traffic control	Reduce Energy consumption	Fragmentation causes high risk
93	Packet delivery ratio	Increased Throughput	Considering the lower number of Nodes with High overhead
94	Multipath Routing Rate of transmission controlled	Increase network lifetime	comparison to other methods is less
95, 96	ETX, Delay and Residual Energy	Increased network lifetime and packet delivery ratio	Reduce stability Frequent change in parent node
97	Multipath routing	Increase network capacity	Increase energy consumption
98	Hop count and route cost	Increase throughput	No real testbed
99	Opportunistic routing	Reduce Duty cycle without causing Delay	Comparison to other methods is less
100	Residual energy Node load	packet Increased delivery ratio and network lifetime	Increase overhead
101	A minimum spanning tree is used to adjust the load of the overloaded node	Reduce power consumption	High message overhead Evaluated with a low number of nodes
102	DIO message is used to transmit congested node	A better spread of workload in the network	Implementation doesn't focus on the reliability
103	The number of the child node is preserved using	Increase stability	Lesser number of nodes is considered and no real testbed

	Q-learning		experiments
104	Node load, residual energy and ETX	Better Network lifetime and packet delivery ratio	No real testbed experiments and Lowered Stability
109	Buffer occupancy, transmission count	Increase network throughput, reduce packet loss and delay	Consideration of mobility in the network is missing, the energy of node is not considered

2.3. Congestion Control by Performing Offloading

Offloading is a process that redistributes the load of traffic nodes to other nodes, effectively passing on the load. This approach proves useful in handling congestion, reducing delays, and improving the success rate of data transmission. However, the implementation of offloading may result in additional processing time and queues in gateways and infrastructure, which may contribute to in-service delays. Nevertheless, offloading requests are proposed in congested IoT networks to prevent communication failures and packet loss.

In [110], the author proposes a game theory-based computation offloading method to improve user benefits and reduce operational costs. This offloading process optimizes resource allocation rates at fog and cloud levels, leading to energy and delay optimization.

In [111], the author focuses on energy conservation in IoT systems embedded in wireless networks and applies cooperative offloading. This approach distributes the download process to edge, cloud, and mobile nodes using various access communications technologies to achieve the best energy efficiency.

For dense IoT networks, [39] proposes an offloading method exclusively for edge computing. This greedy approach involves two tiers named local and mobile edge, where incoming tasks are distributed to minimize overhead processing time and energy consumption.

In [112], the authors introduce a joint offloading optimization method named Stabilized Green Cross-haul Orchestration for service-oriented IoT networks. They utilize Lyapunov theory for drift and penalty policies to optimize data processing rates in an energy-efficient manner, aiming to enhance energy efficiency, network stability, and reduce latency.

To improve scalability, [113] proposes a lightweight request framework integrating various architectures, suitable for IoT and cloud edge-based integrated structures. The framework allows selective independent offloading at the IoT and cloud layers.

In [114], the author considers both offloading and non-offloading devices in the network and analyzes the modelling and deployment of a heterogeneous mobile cloud. The proper placement of cloudlets and distribution of offloading processes are analyzed to minimize IoT communication outages. In [115] attempts to enhance IoT device communication using decision offloading with the K-means algorithm. The network is segregated into position edge servers using the K-means algorithm to balance the offloading process, effectively reducing latency and operational costs.

In [116], task offloading is performed for users using an IoT-based cloud that delivers services on an ad-hoc basis. Distributed location aura provides migration, processing, and initialization with the help of localized IoT devices, managing cost and capacity for load balancing. A task distribution method based on the constructive syncing method is accessible in [117] for IoT-based networks. This approach distributes tasks based on the popularity of processes among fog-connected edge nodes, reducing delay using offloading and queuing techniques simultaneously.

In [124] presents a load balancing method using Loadbot in IoT to reduce traffic congestion caused by high demand. Loadbot calculates the network load and performs structural configurations to manage network loads and user data effectively using the Deep Belief Network scheme, resulting in decreased network load with increasing network size.

In [119], a deep-learning-based load-balancing strategy for IoT is proposed to manage communication overload among users. Loadbot computes the network load and processes structural configurations, and the Deep Belief Network scheme is applied for effective load balancing.

[120] introduces a cognitive method for congestion control in IoT, aiming to improve

reliability, delay, and throughput. The approach incorporates cognition into IoT with a cognitive system based on learning automata, using the CCCLA scheme for congestion control. In [121] presents a data offloading method based on game theory for IoT systems to tackle traffic overload. Utilizing the Rubinstein bargaining game model and Vickrey-Clarke-Groves scheme, a novel pricing approach is proposed to reduce traffic congestion and enhance quality of service.

In [122], the authors propose data processing and traffic control optimization algorithms in IoT to enhance the processing time of traffic signals. They introduce a new smart traffic control system utilizing a remote cloud server and local traffic smart server. The system tracks vehicle transitions and employs the Optimized Regression algorithm to gather multi-path data. At four-direction roadway intersections, single-point nifty decisions are calculated based on waiting vehicle density. Simulation results demonstrate that the suggested strategy reduces wait time effectively.

In [123], the authors present an adaptive offloading scheme based on Genetic Algorithm (GA) for IoT devices to improve communication and traffic handling capacity. They aim to mitigate unnecessary delays and enhance the success rate of IoT requests. The GA-based Adaptive Offloading (GA-OA) approach is simulated using the Opportunistic Network Environment simulator (ONE). The proposed approach achieves reduced delay, processing time, and complexity.

All the referenced papers in this chapter focus on addressing data congestion issues caused by high traffic in IoT environments. While traffic control approaches emphasize the retransmission timeout value to manage traffic and reduce unnecessary packets, other factors like buffer occupancy, link quality, link capacity, network noise, and energy must also be considered in the approach design. Our approaches considered this gap and perform congestion prediction while using various parameters.

For time-constrained applications where delays have significantly affected data packet usefulness, traffic control approaches at the application layer may not be as effective. Instead, resource control approaches at the network layer, which considers congestion and prioritize data packets or offload and balance data to reduce node overhead, are considered more suitable, especially for critical applications like healthcare. To address this, we have focused on data packet offloading where the load is shared among neighbour nodes and congestion-aware data transmission to control the congestion in the network by performing hop-to-hop data transmission by picking the suitable neighbour node to avoid congestion and maintain the efficiency of the IoT networks.

CHAPTER-3

CONGESTION PREDICTION IN IOT NETWORKS

The initial and critical step in congestion handling is congestion prediction, as the effectiveness of control approaches relies heavily on accurate predictions. However, upon analyzing the existing literature, it becomes apparent that the congestion prediction step has received limited attention from the majority of authors**.** Typically, they tend to solely rely on buffer occupancy as the primary indicator of congestion.

In contrast, our approach aims to address this gap by exploring and incorporating various parameters for congestion prediction, thereby enhancing the accuracy of our predictions. Recognizing that buffer occupancy alone may not provide a comprehensive picture of congestion, we have researched deeper into other factors that contribute to congestion in order to improve our predictive capabilities.

By considering a wider range of parameters, we have obtained a more nuanced understanding of congestion dynamics and improved the accuracy of our predictions. These additional parameters include network traffic patterns, packet loss rates, delay variations, link utilization, and other relevant metrics. By analyzing and incorporating these factors into our congestion prediction models, we have aimed to provide more robust and reliable predictions, enabling more effective congestion-handling strategies.

Our approach not only emphasizes the importance of congestion prediction in congestion handling but also expands on the existing literature by exploring and incorporating multiple parameters to enhance the accuracy of predictions. By doing so, we have aimed to contribute to the advancement of congestion-handling techniques and ultimately improve the overall performance and efficiency of network systems.

3.1. Introduction

Our research focuses on congestion handling in IoT networks, specifically by proposing an efficient technique for congestion prediction. To achieve this, we have introduced a novel approach that combines Deep Neural Network (DNN) with

Restricted Boltzmann Machine (RBM) to identify and predict data congestion in Wireless Sensor Network (WSN)-based IoT networks. The machine-learning algorithm utilizes node parameters as input and determines the presence of congestion in a given node. This is particularly important as wireless communication mediums often encounter challenges such as data corruption and network noise.

The algorithm consists of two main sections. First, we have created a comprehensive dataset that includes effective parameters for accurate node congestion prediction. We have considered factors such as congestion window, throughput, propagation delay, round-trip time, number of packets sent, and packet loss.

Next, we have employed the DNN-RBM model to detect data congestion in the network. The DNN takes the dataset as input and predicts network congestion, while the RBM optimizes the weight parameters to enhance the performance of the DNN-RBM system.

We have evaluated the proposed approach's performance based on key metrics such as throughput, congestion window, propagation delay, and accuracy. The results obtained from our model may be effectively utilized to offload data packets from congested nodes to other IoT devices, thereby improving overall network performance and efficiency.

In summary, our research contributes to congestion handling in IoT networks by proposing an innovative technique that combines DNN and RBM algorithms for congestion prediction. By considering multiple parameters and optimizing the model's weight parameters, we have achieved improved accuracy in predicting and managing congestion in IoT networks, enabling more efficient data offloading and network optimization.

To generate the dataset, we have simulated the IoT networks, randomly placing 250 sensor nodes in a 1000x1000 area. Each node has an initial energy of 10.3J and an initial transmission power of 0.66W. AODV is used as the routing protocol, and the packet size is set to 512 bytes. For each sample, nodes have congestion window, throughput, propagation delay, RTT, number of packets sent, and packet loss as parameters.

3.2. Proposed Approach

In our research work, we have aimed to address the issue of congestion prediction within IoT networks by introducing an effective technique. Currently, we have proposed the utilization of DNN-RBM (Deep Neural Network-Restricted Boltzmann Machine) for identifying data congestion within the IoT organization that is based on Wireless Sensor Networks (WSNs). The machine learning algorithm takes the number of tasks or data originating from IoT devices as input. Using this input data, the proposed machine learning algorithm determines whether each device task should be offloaded to the server. In terms of accuracy and increase in the efficiency of the proposed DNN-RBM, the algorithm is presented. As the network is using a wireless communication medium, hence there is a high probability of data corruption and the presence of noise in the network. The algorithm consists of major 3 sections, the first step is to identify the important and effective parameters to predict the congestion in the node. The second step trains the model using RBM and in the last step, data congestion is detected with DNN-RBM.

In this approach, the new model is proposed for analysis and congestion prediction using the systems in the WSN-dependent IOT environment. It will describe the working method of the model: First, the dataset is created with a set of parameters such as congestion window, throughput, propagation delay, RTT, number of packets sent, and packet loss for each sample. The average value of all the parameters in the same data is calculated. With the help of the Marketing Cloud Intelligence Data Pipelines (MDP), based on conditions we have tried to label the data, based on the defined threshold “t” that says if the parameter’s value is less than the average, we have labelled it as congested otherwise not. This allows the labelling or grouping of all data into a dataset array in the next step. This collected dataset is used to generate models that may be used for prediction in various machine-learning algorithms.

3.3. Deep Neural Network (DNN) for Congestion Prediction

An Artificial Neural Network (ANN) is a type of Deep Neural Network (DNN) that incorporates multiple hidden layers positioned between the input and output layers. Deep learning techniques are particularly effective when there is a large volume of training samples available. In line with this, our proposed approach for congestion prediction relies on a DNN-based methodology. During the training process of the DNN, the weights of the neurons are iteratively adjusted until the error between the output and input falls within an acceptable range. This iterative process may be time-consuming.

In our approach, we have utilized the power of DNN to predict network congestion. To make this prediction, we have considered several performance factors as input, including congestion window, throughput, propagation delay, round-trip time, number of packets sent, and packet loss. By leveraging the DNN's capabilities, we have aimed to accurately forecast network congestion based on these input parameters, and RBM is used to optimize weights to the proposed DNN-RBM. DNN includes two phases: pre-preparing and fine-tuning stages in its parameter learning.

3.3.1. DNN Pre-training

A Deep Feed-Forward Neural Network (DNN) serves as a fundamental example of Deep Learning (DL) models. The primary goal of a DNN is to process information hierarchically through various transformation layers, aiming to grasp complex and abstract representations of the input data. The typical structure of a DNN comprises three sections: the input layer, hidden layers, and output layer, with each layer containing interconnected processing units.

Let the input features be represented by $[f_m]$, and the dataset output is denoted by $1<m<N$. The DNN can be envisioned as having multiple iterations (denoted by 'O') to produce the final network output, and at each step, the hidden layer's output is represented as 'O_H'. As the network contains more hidden layers, similar to the DNN architecture, the hidden information representations are enriched through further transformations in subsequent hidden layers. Each hidden layer introduces new sets of weights, which amplify the outputs of the preceding hidden layer. This process results in the generation of progressively refined and complex representations of the input data as it propagates through the network.

The neuron's gradient is added as the information's weight values and enhances the level at the first hidden layer as shown in (3.1):

$$O_{H\_1}(x = 1,2\ldots..,K) = (\sum_{m=1}^{M} w_{xm} f_m) + B_x \tag{3.1}$$

Where bias is represented as $B_x$the constant value, w_{xm} is the interrelated weight between the input and primary hidden layer highlighted with *K* and *M* signifying input hubs in the fundamental hidden layer and the amount of hidden layer. The first hidden layer output's actuation capacity is signified as,

$$F(O_{H\_1}(x)) = \frac{1}{\left(1+e^{-O_{H\_1}(x)}\right)} \tag{3.2}$$

Twisted activation capacity$F(.)$. Subsequently, the activity of n^th the hidden layer indicated as,

$$O_{H\_n}(q) = \left(\sum_{z=1}^{K} w_{qx} F(O_{H\_(n-1)}(x))\right) + B_q \tag{3.3}$$

here w_{qx} is the interrelated weight between the $(n)^{th}$hidden layer and $(n-1)^{th}$hidden layer with K hidden nodes, $B_q$specify the bias of *qth* the hidden node. The invitation work which is the yield of the n^{th} hidden layer is clarified as

$$F(O_{H\_y}(q)) = \frac{1}{\left(1+e^{-O_{H\_n}(q)}\right)} \tag{3.4}$$

In the release layer, the output of n^{th} the hidden layer is then copied back to each other with associated loads (for example weight at intervals the n^{th}output layer and hidden layer) and after a brief orientation by bias *Bp* as

$$O(p) = F\left(\sum_{p=1}^{K} w_{pq} F(O_{H\_n}(q) + B_q)\right) \tag{3.5}$$

Where w_{pq}represents the weight interrelated at the n^{th} output layer and hidden layer accepting p^{th} and q^{th} individually. At the output layer, the initial output serves as the model's prediction.

However, this prediction may differ from the desired target output. The model iteratively refines its predictions by minimizing the error between the model's output and the target output. The calculation of this error is defined by equation (3.6).

$$Error = \frac{1}{M}\sum_{m=1}^{M}(Actual(O_m) - T\,arg\,e\,t(O_T))^2 \tag{3.6}$$

where $T\,arg\,e\,t(O_T)$ denotes the target output and $Actual(O_m)$ is the real output. The fault should be minimized to achieve improved DNN. Subsequently, the weight esteems must be adjusted until the fault in every iteration decrease.

3.3.2. Fine-tuning phase

At this point, the weight parameters of the DNN are adjusted or improved using the restricted Boltzmann machine algorithm.

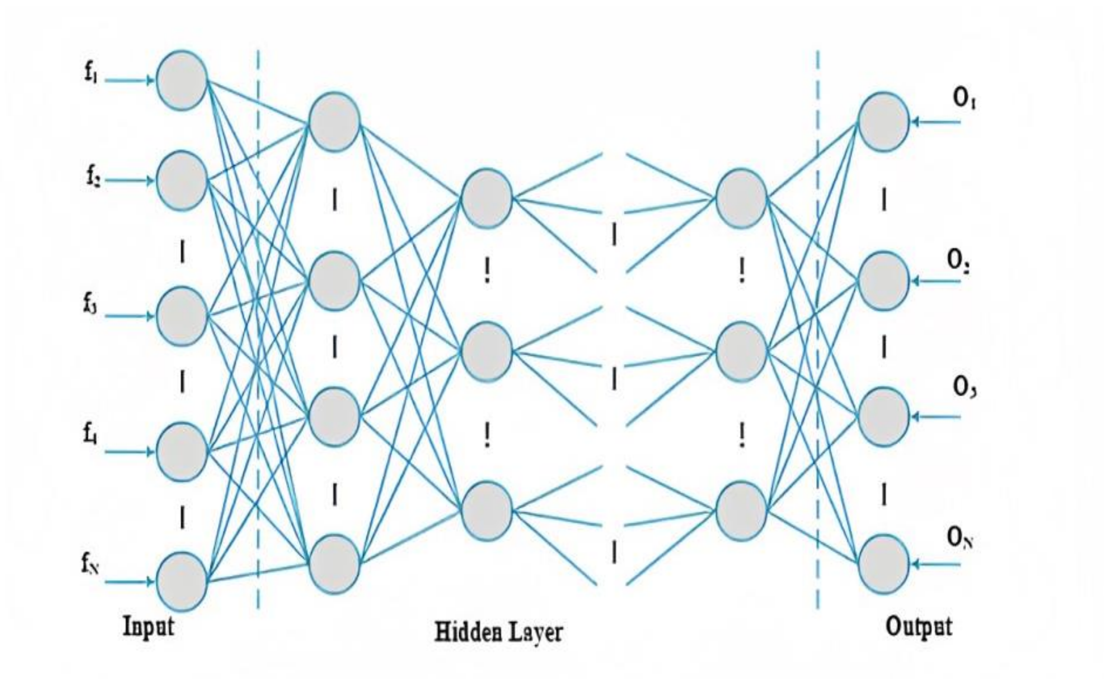

Figure. 3.1: Structure of DNN

Initialization: The weight parameters of the DNN are to be optimized. So, these weight parameters or solutions are initialized as follows in equation (3.8) and input features are shown in equation (3.7):

$$S = \{f_1, f_2, \ldots\ldots, f_N\} \tag{3.7}$$

$$s_N = \{w_{xm}, w_{qx}, w_{pq}\}_N \tag{3.8}$$

Where, w_{xm}denotes the interconnection weight between the input feature and first hidden layer,w_{qx}denotes the interconnection weight between the $(n)^{th}$hidden layer and the $(n-1)^{th}$hidden layer and w_{pq}denotes the interconnection weight at the $(n)^{th}$output layer and hidden layer having q^{th} and p^{th} nodes separately.

Fitness calculation: Once the solutions are initialized, the fitness function is employed to evaluate each solution. In this context, the fitness function is defined as follows:

$$Fit_N = Min(Error) \tag{3.9}$$

Using this fitness function as shown in (3.9) referring (3.6) , each solution is assessed and assigned a fitness value based on the minimum error achieved. The solution with the minimum fitness, i.e., the one that yields the lowest error, is selected as the optimal solution.

3.3.3. Restricted Boltzmann Machines and Deep Belief Network

The RBM (Restricted Boltzmann Machine) is a powerful deep learning algorithm utilized for various tasks such as classification, feature extraction, dimension reduction, feature selection, and regression. It operates through two types of biases. Firstly, the hidden layer bias assists the RBM in generating the activation function during the forward pass. Secondly, the visible layer bias supports the RBM in the reconstruction process during the backward pass.

RBM may be described as an undirected graphical model that employs an energy function as shown in (3.10) and (3.11). This energy function is transformed into a probability distribution by exponentiating the negative energy and normalizing it as shown in (3.12) to (3.21). In RBM, a distribution is defined over the visible layer (represented as S), which incorporates latent variables (represented by the hidden layer). The distribution is obtained by defining an energy function.

$$E_n(x, y) = -y^T Gx - d^T x - b^T y \tag{3.10}$$

$$E_n = -\sum_j \sum_k G_{j,k} y_j x_k - \sum_k d_k x_k - \sum_j b_j y_j \tag{3.11}$$

Where En is the energy function. x and y are the binary units of visible layer and hidden layer *d* and *b* are the biases invisible and hidden layer

Probability distribution

$$p(x, y) = exp\big(-E_n(x, y)\big) / Q \tag{3.12}$$

$$p(y|x) = \prod_j p\big(y_j|x\big) \tag{3.13}$$

$$p(y_j = 1|x) = \frac{1}{1+exp\left(-\left(b_j+G_jx\right)\right)} \tag{3.14}$$

$$= \lambda\left(b_j = G_jx\right) \tag{3.15}$$

$$p(x|y) = \prod_k p(x_k|y) \tag{3.16}$$

$$p(x_k = 1|y) = \frac{1}{1+exp\left(-(d_k+y^T G_{\cdot k})\right)} \tag{3.17}$$

$$= \lambda(d_k + y^T G_{\cdot k}) \tag{3.18}$$

$$p(x) = \sum_{y\varepsilon\{0,1\}^Y} p(x,y) = \sum_{y\varepsilon\{0,1\}^Y} exp(E_n(x,y))\,/Q \tag{3.19}$$

$$= exp\left(d^T x + \sum_{j=1}^{Y} log(\,1 + exp(\,b_j + G_j.x))\right) \tag{3.20}$$

$$= exp\left(-F(x)\right)/Q \tag{3.21}$$

The DNN employs a training technique where two layers are trained simultaneously, treating them as Restricted Boltzmann Machines (RBMs). In this approach, the hidden layer of one RBM serves as the input layer for the neighbouring RBM in the network.

The first RBM is trained, and its output is then used as input for the subsequent RBM. This process continues until the production layer is reached. Through this training phase, the DNN is able to capture the underlying patterns and trends present in the data.

Termination criteria

The termination criteria for this calculation occur when the maximum number of iterations is reached and the solution with the best fitness value is obtained. At this point, the weight parameters that yield the best results are identified. Once the RBM algorithm achieves this improved performance, the selected weight parameters are applied to the DNN. The overall process of weight improvement based on RBMs is depicted in Figure. 3.2 where The expression 'Wnew = w + 1' indicates the iterations, representing the number of times the weights are updated.

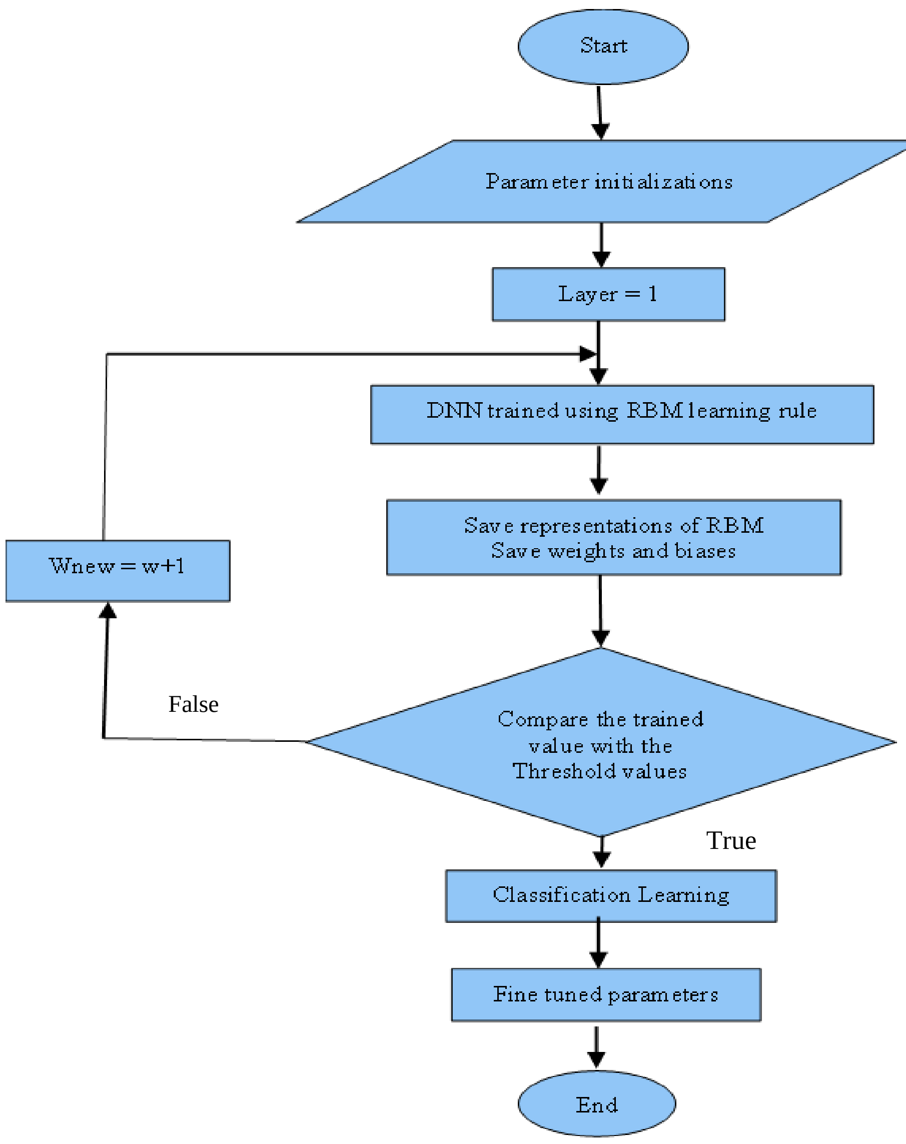
Start
Parameter initializations
Layer = 1
DNN trained using RBM learning rule
Save representations of RBM
Save weights and biases
Wnew = w+1
False
Compare the trained value with the Threshold values
True
Classification Learning
Fine tuned parameters
End

Figure. 3.2. Congestion Prediction Progress

3.4. Simulation Results and Comparative Analysis:

The proposed approach DNN-RBM for congestion control is executed in the foundation of Python. Table 3.1 presents the comparative analysis of three different machine-learning techniques for accurately predicting data congestion. This analysis helps determine the best method for the task.

From the experimental results depicted in Table 3.1, it is observed that Linear Regression supervised all other machine-learning techniques. This may be attributed to the suitability of regression methods in forecasting time-dependent data. Precision, recall, and accuracy serve as metrics for evaluating prediction performance. Accuracy, which represents the sum of true positives and false positives, is a crucial metric for measuring the prediction results. Recall is defined as the sum of true positives and false negatives, specifically considering true positives. Precision is determined by summing all true values and false values, focusing on true values.

Performance Metrics

Our proposed approach, DNN-RBM, is evaluated using the following metrics, which are then compared to the performance of DNN and Logistic Regression:

Packet Loss: The use of acknowledgements (ACKs) on the sender side allows for measuring accountability. The protocol ensures confidence by guaranteeing reliable delivery. Additionally, packet loss may be measured by examining the sequence numbers on the receiver side. For instance, the loss of Clear to Send (CTS) packets may serve as an indicator of congestion.

Propagation Delay: High transmission delays, compared to transfer delays, may increase the likelihood of packet loss. Propagation delay refers to the time it takes for a packet to travel from the source to its destination.

Throughput: Throughput measures the amount of data that may be transmitted from the source to the target per second. It is typically expressed in kilobits per second (kb/s) and provides insights into the system's capacity.

Round-Trip Time (RTT): RTT represents the total time required for sending the first packet and receiving its corresponding response packet. It reflects the latency or delay in the communication between nodes.

Buffer Occupancy: Each sensor node in the network has a buffer that stores packets before wireless transmission. If the buffer occupancy exceeds a predefined threshold, a congestion warning is triggered. Monitoring the buffer threshold is a straightforward and effective indicator of congestion.

By considering these metrics, we have assessed the performance of our proposed approach, DNN-RBM, and compared it with the results obtained from DNN and Logistic Regression approaches.

Performance Analysis

Table 3.1 shows the comparative results with different machine learning techniques.

Due to the weight optimization of DNN using RBM, the prediction of congested packets is improved accuracy. The proposed DNN-RBM model having a high F1 score means that it maintains a strong balance between minimizing false positives and false negatives. This balance is desirable for many classification tasks, indicating that the model is reliable and accurate in its predictions.

Table 3.1 Comparative Analysis with Machine Learning Techniques

Techniques	Precision	Recall	Accuracy	F1 Score
Decision Tree	100	88.03	88.05	93.56
Random Forest	100	79.80	79.81	88.76
Linear Regression	98.9	80.81	80.82	89.10
DNN-GA	99.3	84.24	89.31	91.21
DNN-RBM	99.9	90.45	90.55	94.92

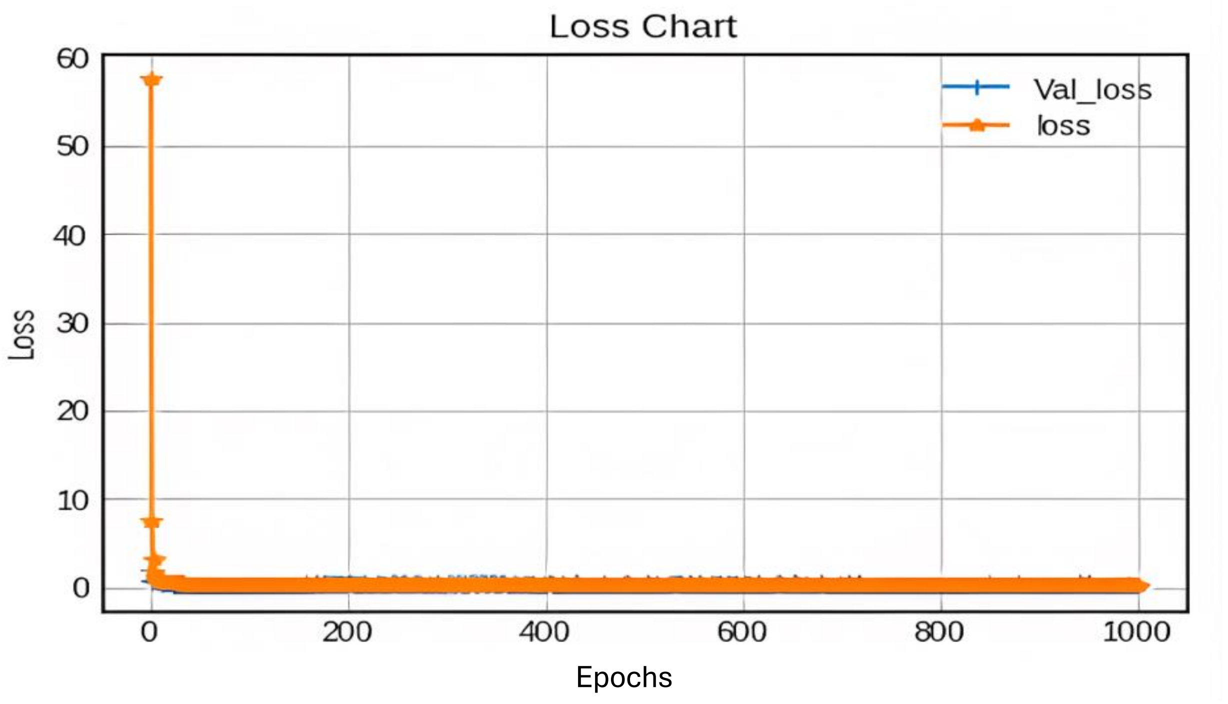

Figure. 3.3 :Loss Chart for Epoch=1000 (DNN-RBM)

The results, therefore, demonstrate that the DNN-RBM performed well, providing an overall accuracy of 99.9%, 90.4%, and 90.5%. In Figure 3.3, the loss chart displays the training and validation losses for the DNN-RBM model over an epoch count ranging up to 1000 and Figure. 3.4 the accuracy chart displays the validation and training accuracy for different epoch counts. The epoch count ranges up to 1000, and the line graph illustrates the trends of validation accuracy and training accuracy over these epochs.

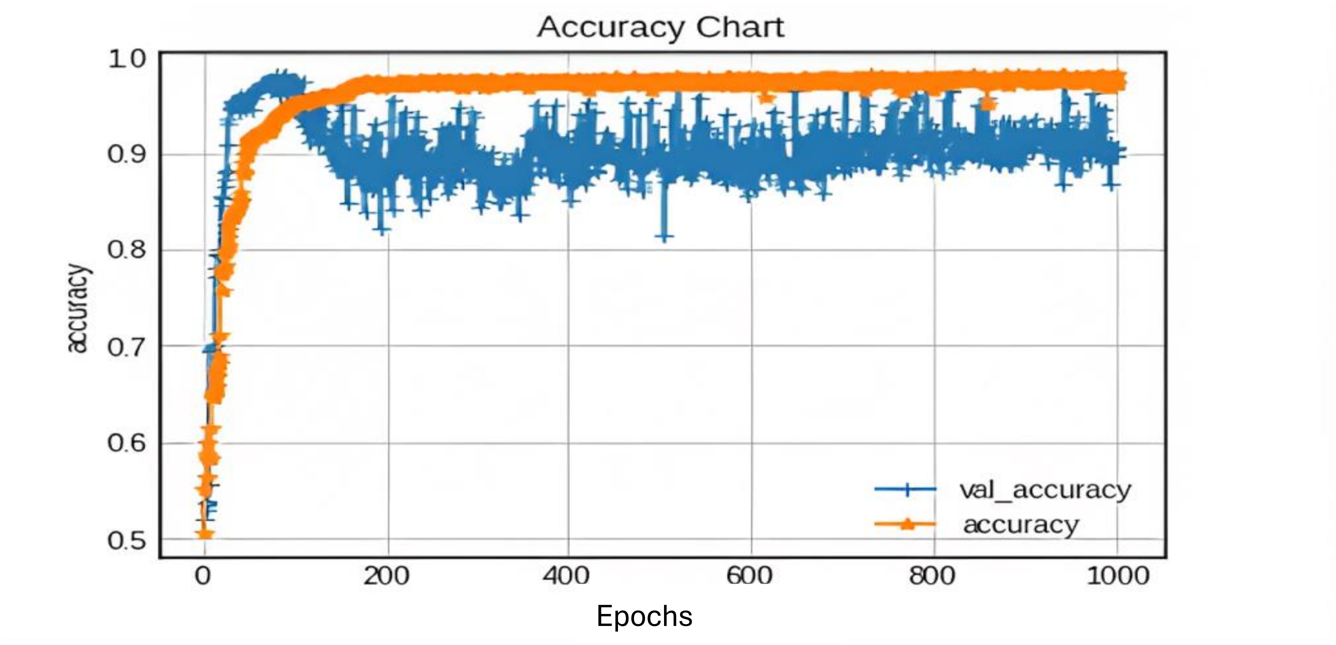

Figure. 3.4: Accuracy Chart for Epoch = 1000 (DNN-RBM)

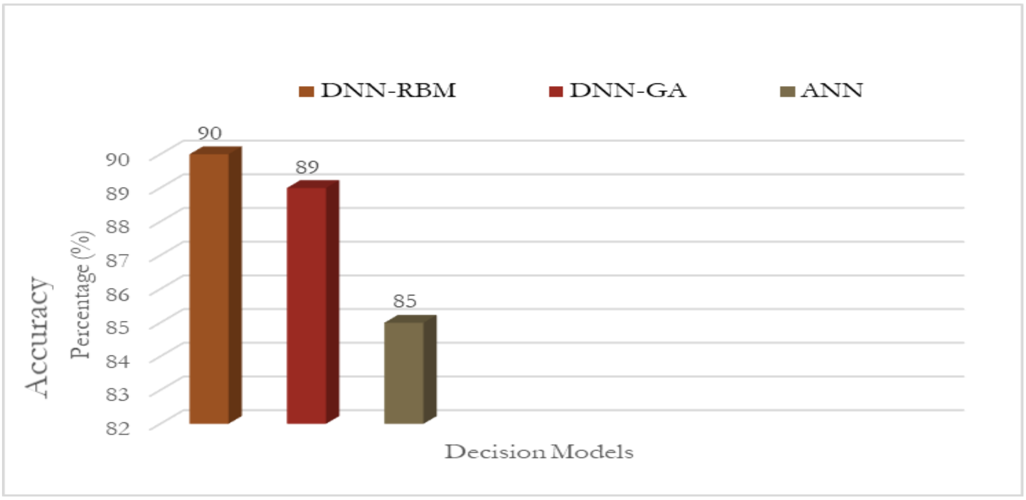

Figure. 3.5: Accuracy of Different Decision Models

Figure. 3.5 represents the Accuracy of different models for which the DNN-RBM model is with 90% accuracy, the DNN-GA model is with 89% accuracy and the ANN model is with 85% accuracy.

3.5. Summary

In this chapter, the DNN-RBM approach for predicting congested packets in WSN-based IoT is introduced. It combines the Restricted Boltzmann Machine (RBM) algorithm with a Deep Neural Network (DNN) to optimize weight values and improve performance. Input factors such as congestion window, throughput, delay, RTT, packets sent, and packet loss are used to predict congestion in nodes. The proposed DNN-RBM model achieves an accuracy of 90%, outperforming the DNN-GA (89%) and ANN (85%) models. Future work involves developing a robust algorithm to automatically adjust threshold values based on network type for congestion handling.

CHAPTER-4

DATA CONGESTION CONTROL USING DATA OFFLOADING IN IOT NETWORKS

In order to effectively handle the problem of congestion, congestion prediction serves as the crucial initial step, followed by congestion control to mitigate congestion and enhance network efficiency. In this work, we have specifically concentrated on the congestion control approach, which plays a pivotal role in addressing congestion.

Congestion control encompasses the implementation of mechanisms such as load balancing and regulation of data packet transmission rates to optimize network performance. By balancing the network load and modulating data transmission, congestion control aims to maximize network efficiency and minimize congestion-related issues.

Within the context of IoT networks, our approach focuses on offloading data packets to balance the load on individual nodes. This load-balancing technique aims to distribute the network traffic evenly across nodes, mitigating congestion hotspots and ensuring smoother network operation. The proposed approach delivered data packets with minimal delay and packet loss by effectively implementing congestion control approaches.

In summary, our work emphasizes the significance of congestion control in effectively managing network congestion. By focusing on offloading data packets, load balancing, and minimizing delays and packet loss, we have strived to improve network efficiency and ensure seamless communication in IoT networks.

4.1. Introduction

The Internet of Things (IoT) enables communication between devices without human intervention. It involves physical devices like sensors that collect and transmit real-time information. IoT aims to create a network environment for reliable communication among various devices, such as household appliances and vehicles, across different networks. It is predicted that there will be a significant increase in the number of IoT devices, with an estimated 10 billion connected devices by 2025. To ensure smooth communication in the face of increased traffic, congestion control is crucial for minimizing delays and packet loss.

The routing protocol, called RPL, handles the process of data transmission from nodes to border routers. However, the RPL protocol faces challenges when routing in heavily loaded networks due to its initial design based on low dynamicity and throughput.

In summary, IoT enables automated device communication, and congestion control is vital for efficient communication. The IETF has standardized IoT, including the RPL routing protocol, which faces difficulties in routing in heavily loaded networks.

4.2. Proposed Approach

In this chapter, we have addressed congestion issues by employing data offloading operations. Data offloading involves transferring the burden of received data from one node to another. When the traffic rate exceeds a node's processing capacity and the request generation is high, nodes become congested, resulting in a buffer overflow, packet loss, and increased delays that degrade network communication quality.

We have presented a congestion control offloading approach specifically designed for RPL. While previous approaches have focused on selecting reliable parent nodes and designing effective objective function metrics, they typically handle congestion after it has occurred. In these approaches, congested nodes notify their child nodes about the congestion, and the child nodes select new parent nodes to continue packet transmission. The new parent nodes then handle the packets of both their old and new child nodes, which introduces additional overhead and computational burden.

In our proposed approach, we have avoided the process of asking child nodes to select new parent nodes, thereby minimizing unnecessary changes in the network topology that may lead to delays and packet loss. Instead, we have calculated the congestion value of a node based on its parameter values and the impact of these parameters. Since RPL constructs a Destination-Oriented Directed Acyclic Graph (DODAG), with each node ranked based on Objective Function (OF) metrics, we have considered nodes with the same rank as potential neighbours. Once a node detects congestion, it starts searching for a suitable neighbour to offload some of its tasks. Rather than replacing the parent node entirely, the congested node treats the suitable neighbour as a helper and shares its load with them, effectively relieving congestion. This approach allows the node to overcome congestion by distributing the load with its neighbour.

The main idea behind our approach have been divided into two steps. First, we have identified congested nodes by calculating their congestion values. Second, we have identified suitable neighbour nodes capable of assisting the congested node by carrying its data packets, thereby alleviating congestion.

In Figure. 4.1, we have presented a small network instance where nodes are connected in a DODAG, and communication and routing are performed using RPL. The red node represents the root node, and the dotted lines indicate the various intermediate levels between the root and child nodes. Let's assume that in this example, the congested node is represented by the pink node, which is identified based on its congestion value.

To alleviate congestion, we have selected neighbouring nodes with the same rank and coverage as the congested node. In Figure. 4.1, these neighbouring nodes are depicted in blue. Among the blue nodes, one node is chosen as the helper node for the congested pink node. The selection process considers factors such as distance and congestion value. The brown nodes represent the child nodes of the rank X nodes.

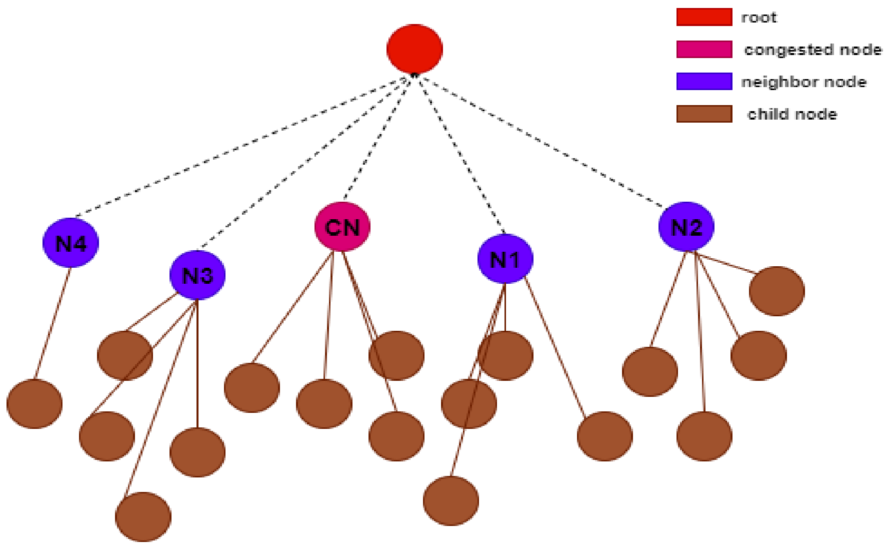

Figure. 4.1: Network Instance for Proposed Approach

In the figure, we have observed that there are only four neighbour nodes, N1 to N4, present for the congested node at level X. The selection of the helper node depends on various factors, including its distance to the congested node and its congestion value. The algorithm provides a more detailed explanation of the neighbour node selection process.

Please refer to the algorithm for a comprehensive understanding of how the selection process is carried out in practice. To identify the most suitable neighbour and predict congestion, several parameters of the node need to be considered. These parameters include buffer occupancy, Expected Transmission Count (ETX), remaining energy, and node load in terms of child nodes. By evaluating these parameters, the congestion value of a node has been calculated, enabling the prediction of congestion within the network.

The congestion value is computed periodically by each node. If a node detects that its congestion value exceeds a certain threshold, it initiates the congestion avoidance process. This proactive approach allows nodes to identify and mitigate congestion before it causes network degradation.

By considering parameters such as buffer occupancy, ETX, remaining energy, and node load in relation to its child nodes, nodes may assess their congestion status and take appropriate actions to alleviate congestion and maintain optimal network performance.

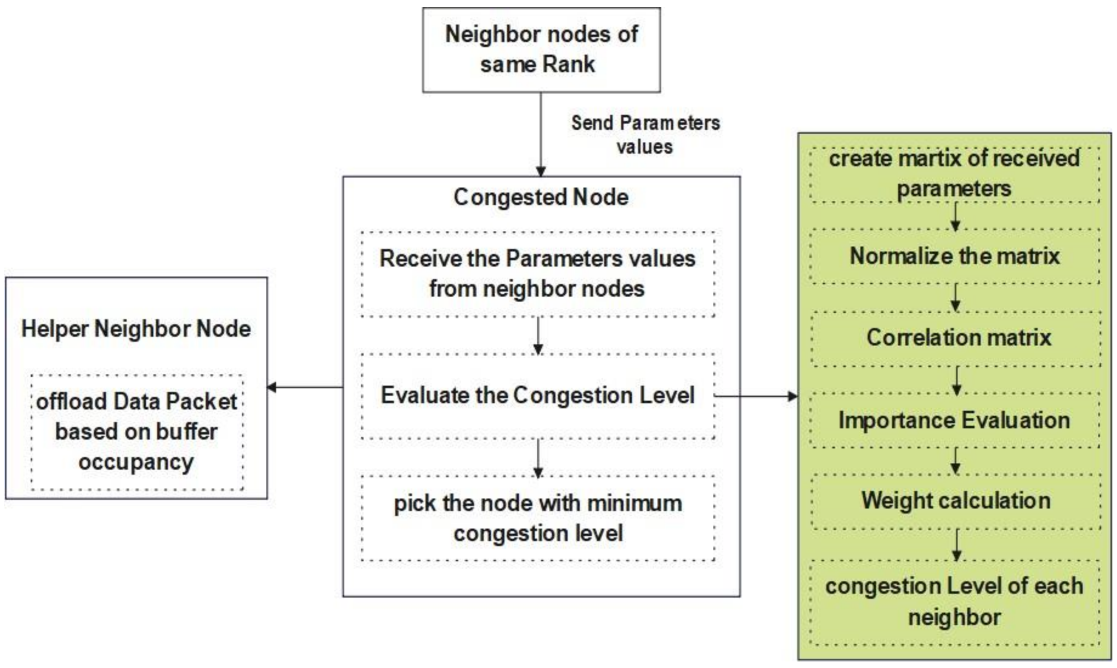
Neighbor nodes of same Rank
Send Parameters values
Congested Node
Receive the Parameters values from neighbor nodes
Evaluate the Congestion Level
pick the node with minimum congestion level
Helper Neighbor Node
offload Data Packet based on buffer occupancy
create martix of received parameters
Normalize the matrix
Correlation matrix
Importance Evaluation
Weight calculation
congestion Level of each neighbor

Figure. 4.2: Process of Data Offloading

The congestion avoidance is done by performing offloading. The congested node finds suitable neighbour to offload some of its burdens to handle the current situation. For the selection of the suitable neighbour node, the distance between the node and its neighbour as well as the congestion value are considered.

The parameter correlation method is used which is based on standard deviation for assigning the weights to the parameter. This method is used frequently for multiple-parameter decision-making processes. In this method to find the contrast intensity of each parameter, standard deviation is applied, and for conflicts or differences between parameters is considered as the coefficient correlation between parameters. This is applied to find the importance of each parameter. There are n parameters with each node and m alternative for the parent node neighbours.

The process of identifying the node Congestion value based on the parameter is followed by the listed steps.

Step 1. Initial matrix for the entire alternative with their parameters

$$y_{ij} = \begin{matrix} P_{11} & P_{1n} \\ P_{n1} & P_{mn} \end{matrix} \qquad i = 1,2,\ldots,m; j = 1,2,\ldots,n \qquad (4.1)$$

Where, y_{ij} represents the value of ith alternative for jth parameter.

Step 2: Normalization of all the parameter values by using equations (4.1) and (4.2). This is done to make all value to be in the same range so that it will be easy to compare and assign the weights to the parameter.

$$p_{ij} = \frac{y_{ij} - y_i^{min}}{y_i^{max_i}} \qquad i = 1,2,\ldots,m; j = 1,2,\ldots,n \qquad (4.2)$$

After the process of normalization, all the parameters are in the same range and now the impact of each parameter is calculated by the weight which is given to them.

Step 3: linear correlation matrix (4.3) is created. This process helps to identify the relation and the dependence of the parameter to each other.

$$V_{jk} = \sum_{i=0}^{m} \left(p_{ij} - \overline{p_j}\right) - \left(p_{ik} - \overline{p_k}\right) \Big/ \sqrt{\sum_{i=0}^{m} (p_{ij} - \overline{p_j})^2 \sum_{i=0}^{m} (p_{ik} - \overline{p_k})^2} \qquad (4.3)$$

Step 4: These weights are calculated by performing linear correlation and standard deviation between parameter values of the metrics. In equation (4.4) represents the amount of information obtained by parameter k of j alternative.

$$\beta_{kj} = \sigma \sum_{k=1}^{n} (1 - v_{kj}) \tag{4.4}$$

where, represents standard deviation of the kth parameter.

$$w_{lk} = \frac{\beta_{kj}}{\sum_{k=1}^{n} \beta_k} \quad \text{Where k=1….n} \tag{4.5}$$

In equation (4.5), represents the weight value for the kth parameter. The value of weight is higher if the standard deviation is high and correlation with each other is low. If the value is high means it is having a high amount of information hence weight assigned to it is also high. The rank value is calculated by considering the above-calculated weights for listed parameters to find the final value of the node.

Step 5: In this step node congestion value is calculated as,

$$Congestionvalue_j = \frac{\sum_{i=0}^{n} P_{ij}{}^{*} w_i}{\sum_{i=0}^{n} w_i} \quad \text{Where i=1….n} \tag{4.6}$$

This congestion value is calculated for every node periodically. When any node finds its congestion, value is greater than 70% it considers that node as a congested node. This percentage is set based on various simulations and analyzes to focus on the best result and better utilization of memory. After identification of congestion, it will find the neighbour node which may share its load without notifying about the congestion to their child nodes. This helps to have communication without changing the DODAG.

In the algorithm, we have input the set of neighbour nodes with same rank and in the node’s communication range. To reduce the load of comparison input is further reduced by removing the nodes with higher congestion value, for our case it is set to 60%. Once the input nodes find itself congested it beings the process to handle congestion. In this process, we have compared the entire neighbour node with each other in terms of distance and congestion value. The node which is having a minimum combination of both will be selected as a suitable node to help the congested node.

This process helps the node to handle the situation of high traffic and not overhead their child node for changing the topology. In the IoT networks, traffic situation occurs suddenly and generally for a small duration of time like the transfer of monthly or daily datasheet.

Algorithm: Find the least congested next node

Input:

P_n = node n; **C**: congestion threshold; **L_n**: Set of nodes at the same level as the of P_n and **within its range** and having **congestion value < C**

Output: Find the suitable node to pass the data information P_{sel}: selected node

Algorithm:

```
        $suitable\_value = \infty \quad suitable\_node = \infty$
For each $1 \in L_n$ do
        $suitable\_value\_node \leftarrow dis\,tan\,c\,e(l,n) + congestion\_value(1)$
If $\boldsymbol{suitable\_value\_node < suitable\_value}$ then
        $suitable\_value \leftarrow suitable\_value\_node$
        $P^{sel} \leftarrow 1$
End If
End For
Return $P^{sel}$
```

This process will occur short span of time, for this changing in the topology repeatedly may cause a high cost.

To avoid this proposed approach just select the helper neighbour. The parent node calculates the space in a buffer of the helper neighbour and offloads child data packet through tunnel without processing it to the helper neighbour node. This helps parent node to process their buffered packet and able to avoid the situation of congestion. The process to identify best suitable neighbour node for offloading is shown in algorithm.

4.3. Simulation Results and Comparative Analysis:

This part of the chapter estimates the performance of the proposed offloading approach for congestion control. To evaluate the performance, various network scenarios are tested with different numbers of nodes and traffic rates. The proposed work is compared to RPL, CoAR and CAFOR. The comparative analysis is done on the bases of various parameters such as packet loss rate, throughput, network lifetime and buffer utilization.

Network model and environment: The proposed approach is implemented in open source CONTIKI 2.7 operating system and COOJA is used for the simulation. It is used for various IoT applications. The network is composed in an area of 200 X 200 m2 and uses transmission speed of 250kbits/s. The network layer lies on the top of 1EE 802.15.4 MAC. The network consists of 1 gateway or receiver node which acts as the root node of DODAG and 50 sender nodes which are randomly distributed in the network. The time period for each simulation is set to 8 minutes run and the size of the packet is 56 bytes. Data packets are transmitted by the source nodes after 60 seconds from the start of the simulation. In each simulation, the source node is selected randomly and independently. The results are calculated by performing each simulation 10 times and the average is considered. In Table 4.1 parameter setting is shown.

Table 4.1 Parameters Value for Simulation

Parameter	Value
Topology area	200 x 200 m^2
Transmission range	60 m
Interface range	70 m

Mote type	Tmote Sky
Data link	CSMA; Contiki MAC(RDC)
Radio Model	Unit Disk Graph Medium
Queue Type	FIFO
Queue size	12 packets
Adaptation	6LoWPAN
Channel Check Rate	16 Hz

In this approach, we have compared the performance based on various parameters like energy consumption, packet loss ratio, packet receive ratio throughput and end-to-end delay. This is done by considering the different numbers of nodes in the network and the variable traffic load. All the parameters are explained and show the result in comparison to RPL, CoAR, and CAFOR.

Energy consumption: In this, we have considered the consumption of energy from source to destination for every transmission of packets. The proposed approach consumes lesser energy as compared to others. As in the proposed approach, we have not changed the parent after the occurrence of congestion instead we have avoided the case of congestion by predicting a probability of congestion and offloading the packet to a neighbour without changing the actual initially assigned parent. This helps in avoiding the energy usage for changing the parent node as well as it reduces the number of retransmissions and reduces energy consumption.

Table 4.2 Performance Comparison of Different Approaches

Parameter	RPL	CoAR	CAFOR	Proposed
Packet Loss	59%	18%	14%	8%
Energy Consumption	40%	23%	22%	14%
Throughput	19%	26%	27%	28%
Delay	38%	23%	21%	18%

The number of retransmissions is higher in CAFOR, CoAR, and RPL as compared to the proposed approach which results in higher energy consumption. Figure. 4.3 shows the energy consumption with different traffic load situations. When there are 480 packets per minute as traffic load then also its energy consumption is lesser than other approaches.

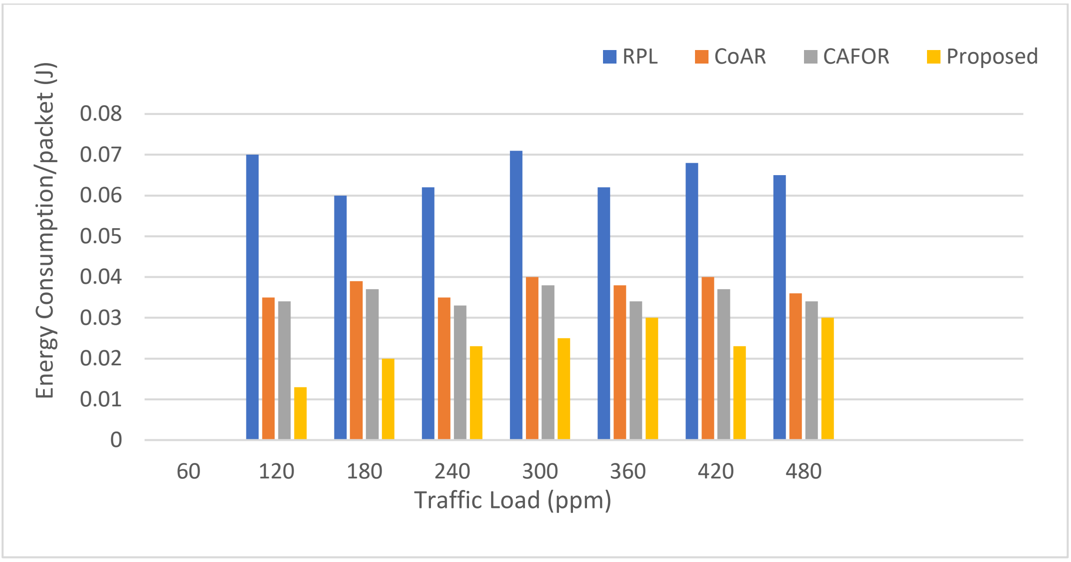

Figure. 4.3: Energy Consumption/ Packet According to the Traffic Load

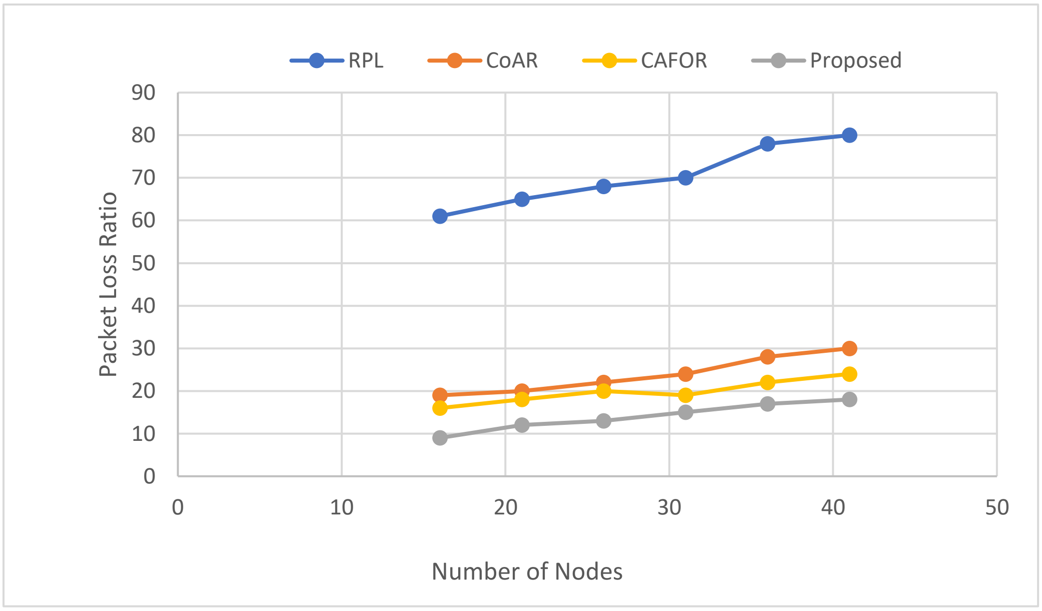

Figure. 4.4: Packet Loss Ratio with Number of Nodes

Packet loss ratio: The lost packet ratio with the total packet sent is given as packet loss ratio for variable traffic load and variable number of nodes. As shown in Figure. 4.4 and 4.5 with the increase of the number of nodes and traffic load the packet loss ratio is also increased respectively. The proposed approach causes lesser packet loss as compared to other approaches. When transmission data rate is set to higher value of 540 ppm then the percentage of packet loss ratio is 16% for proposed approach whereas another CAFOR is 22%, CoAR is 29%, and RPL is 65%. Similarly, the packet loss ratio with a number of nodes 40 is 80% for RPL, 30% for CoAR, 24% for CAFOR and 18% for proposed approach. Originally RPL did not have any approach to congestion control hence causing a higher rate of packet loss. The proposed approach is better than others because it finds the congestion value to identify the chances of congestion at a node and handles it by performing offloading. The process reduces the overall packet loss cases.

End to end delay: It measures the total time consumed for the data packet to travel from sender to the root node. It starts the time from the packet generation till it root node received it. To reduce the delay, we have performed offloading from the parent node instead of going back to the child node to select the new parent and perform retransmission.

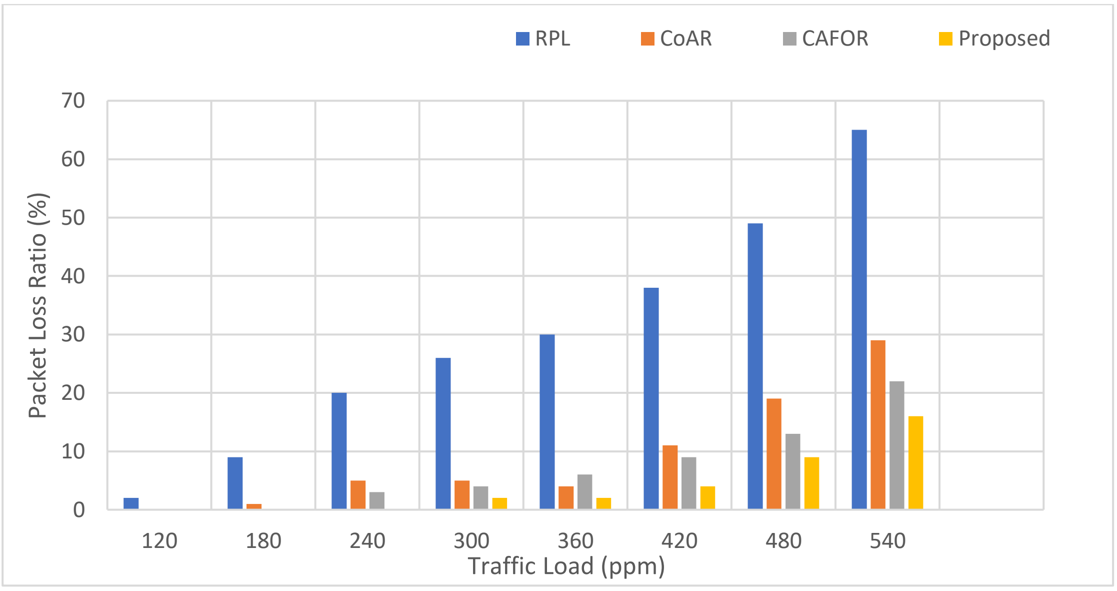

Figure. 4.5: Packet Loss Ratio According to Traffic Load

The RPL algorithm doesn't have any mechanism to control delay hence, causes a higher delay in comparison to CoAR, CAFOR and the proposed algorithm. Figure.

4.6 shows the delay per packet at different simulation times. It is observed that the variation in delay per packet for the proposed approach is between 1.5 to 2.5 which is low in comparison to RPL CoAR and CAFOR.

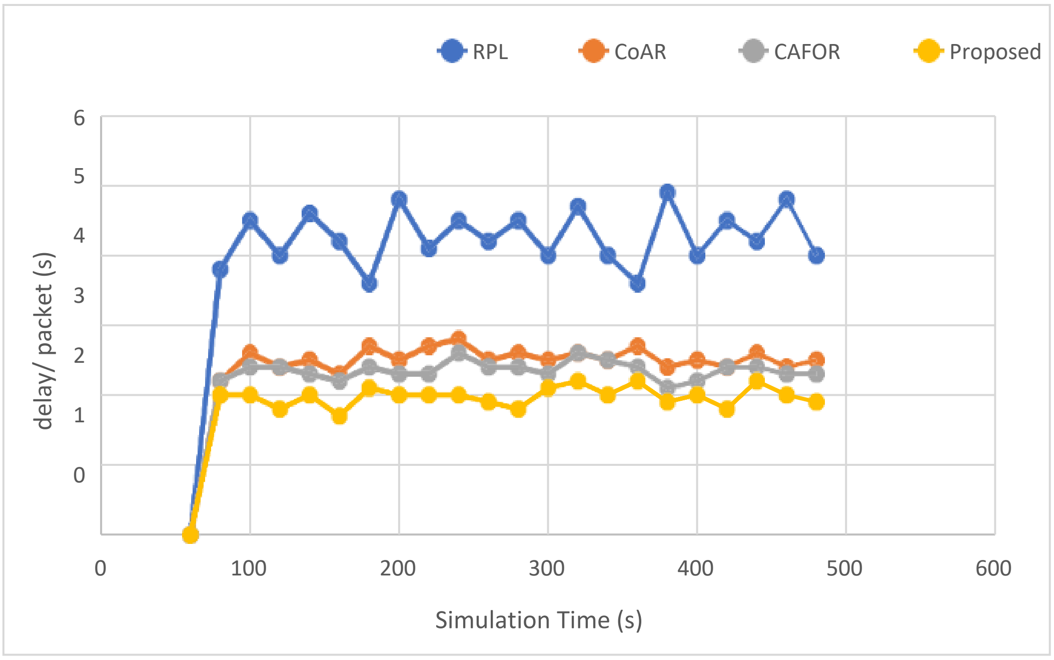

Figure. 4.6: Average Delay with Respect to Time

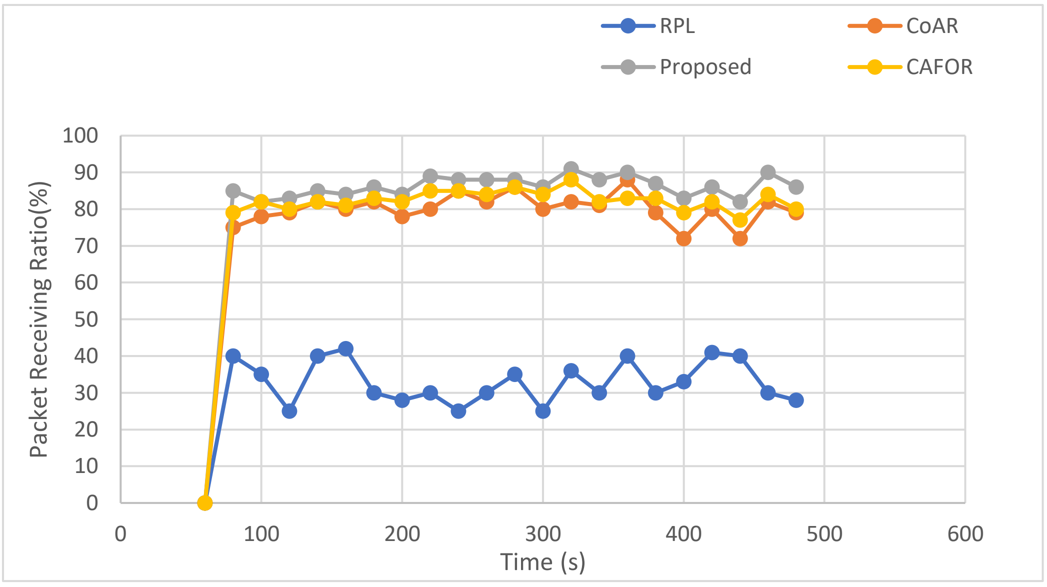

Figure. 4.7: Packet Receiving Ratio with Respect to Time

Packet Receiving Rate: It is used to analyze the number of packets received by the root with respect to the total sent packets by source nodes. There are various intermediate nodes in between source and the root so that communication have been carried out using multiple nodes.

Due to high traffic load and limited buffer size result in packet drop. This is the reason for the reduction of packet receiving rate and only packets received at destination. To analyze the performance of the proposed approach, packet receiving rate is calculated with different traffic load range between 60-540 packet/minute as shown in Figure. 4.7. With the increase in the traffic load, the rate of packet reception will decrease. The proposed approach gives better results as compared to others.

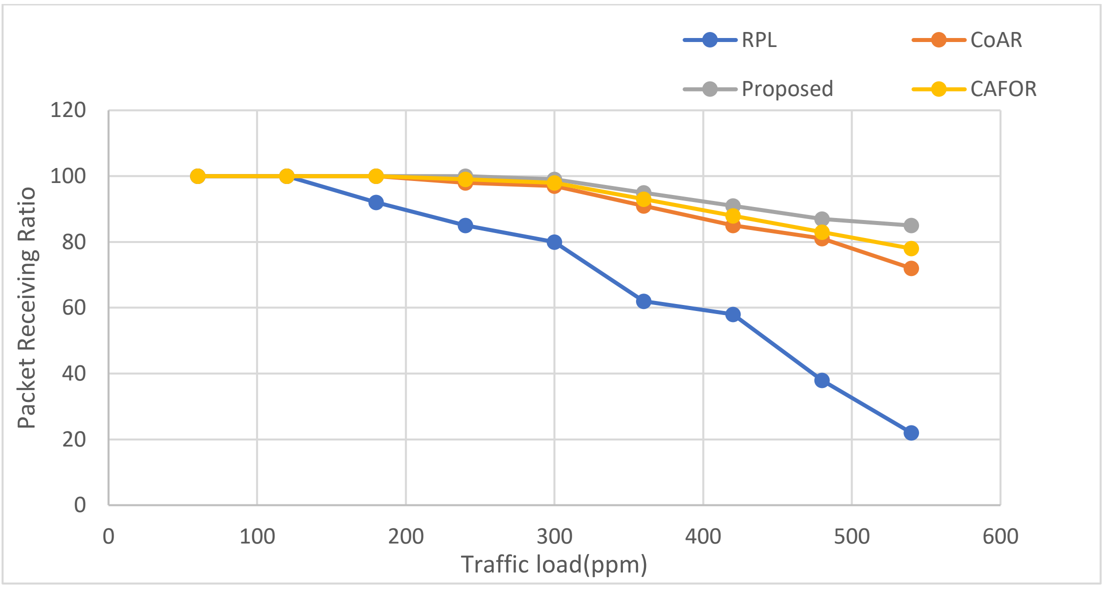

Figure. 4.8: Packet Receiving Ratio with Respect to Traffic Load

In Figure. 4.8, different simulation time is shown for the packet receiving ratio and the proposed approach remains constant with an 80% to 90% receiving ratio. Whereas for CoAR and CAFOR it varies between 70%-90%. In Figure. 4.9, the packet-receiving ratio is shown for different numbers of nodes. When the number of nodes reaches to 40 then also it maintains the packet receiving ratio of 70% as shown in Figure. 4.9 and CoAR and CAFOR have achieved 50% and 60% respectively when number of nodes is 40.

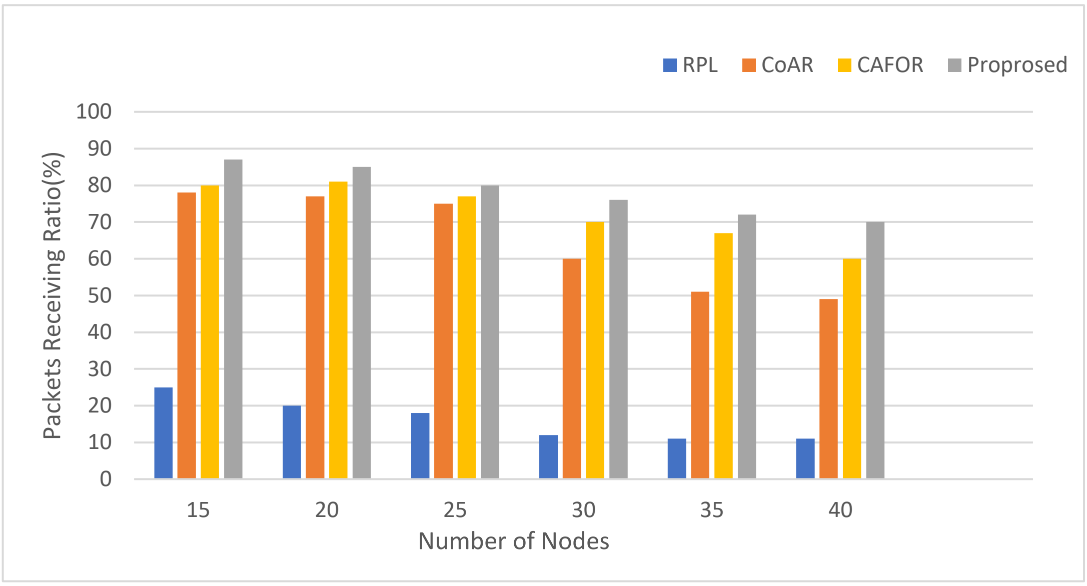

Figure. 4.9: Packet Receiving Ratio with Respect to Number of Nodes

Throughput: it measures the number of bytes received by the root node in a unit of time. As shown in Figure. 4.10, throughput is increased when there is less traffic load but the proposed approach is able to touch higher throughput even with increased traffic load as compared to other approaches. The RPL is not able to work effectively with the higher traffic load as it doesn't have any method for congestion control.

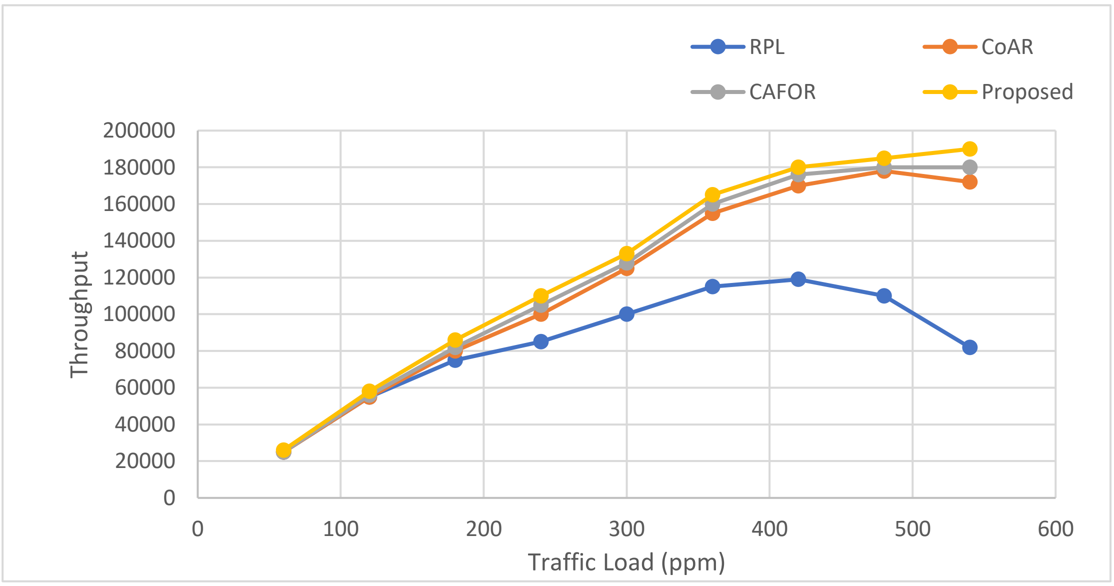

Figure. 4.10: Throughput with Respect to Traffic Load

The proposed approach, CAFOR and CoAR are effectively working in high network traffic conditions. In Figure. 4.11, throughput is mapped with the number of nodes in the network. The proposed approach performed really well as compared to others and increased throughput with the increase in network size. Overall throughput is enhanced by the proposed approach in both cases larger number of nodes and higher data traffic. While comparing the proposed approach with other approaches it is visible the results are in favour of our approach and this is possible because it starts working by analyzing the congestion instead of waiting for buffer overflow and causing packet loss.

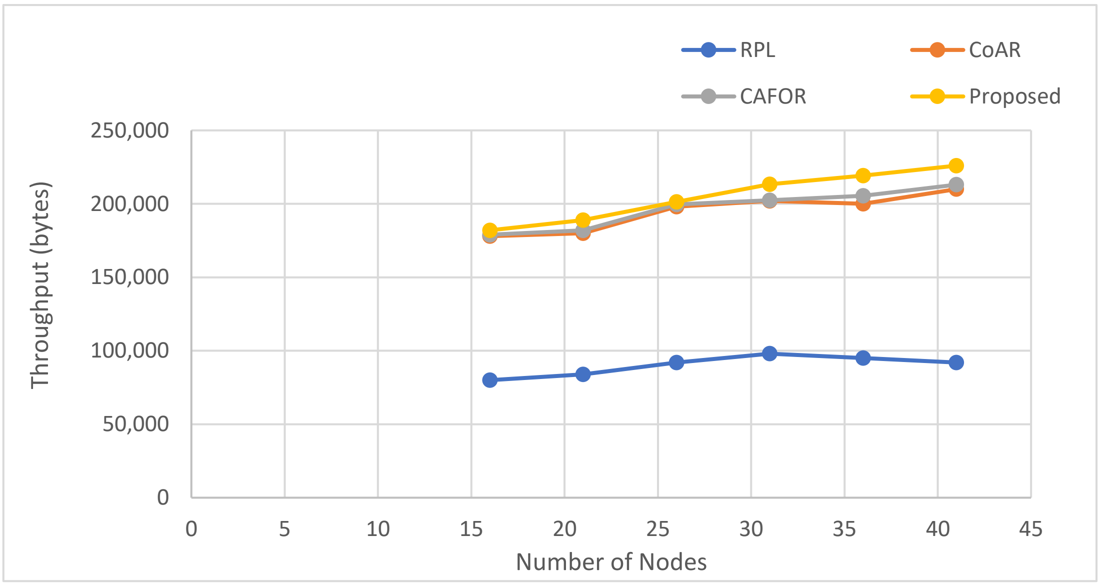

Figure. 4.11: Throughput Concerning the Number of Nodes

The parent node itself starts taking care of its packet and handles the congestion by performing offloading. The proposed approach led to reliable communication by reduction of delay and packet loss rate. It also saves energy by avoiding the requirement of retransmission and hence increase the network lifetime.

4.4. Summary

In this chapter, we have proposed a congestion control approach for RPL-based networks using an offloading mechanism. It identifies congested nodes based on past traffic, congestion level and buffer occupancy, and offloads their load to nearby nodes with lower congestion values. The proposed approach improves throughput, packet receiving ratio, and reduces packet loss, delay, and energy consumption compared to existing approaches. It achieves this by maintaining the initial network structure and balancing the load of congested nodes through offloading. Based on the simulation results compared to RPL, CoAR, and CAFOR, the proposed protocol outperformed in terms of Packet Loss (8%), Energy Consumption (14%), Throughput (28%), and Delay (18%).

CHAPTER-5

CONGESTION-AWARE DATA TRANSMISSION IN IOT NETWORKS

In order to mitigate resource consumption in constrained IoT devices and optimize network performance, congestion-aware data transmission techniques are employed to minimize the overhead caused by delayed packet retransmission and packet loss. Typically, congestion control measures are initially implemented to address network congestion, followed by the application of congestion-aware data transfer techniques to reduce the impact of packet loss or delayed packet retransmission. However, in this work, we have specifically focused on enhancing congestion-aware data transmission within the constrained IoT networks by introducing a hop-to-hop approach.

Our approach emphasizes selecting the next hop for data transmission based on various parameters, including the distance, hop count, residual energy, link quality and buffer occupancy of potential receiving nodes. By considering these parameters of individual nodes, congestion and node condition are known and help to achieve congestion-aware and efficient data transmission by selecting the appropriate next hop.

By incorporating hop-to-hop congestion awareness, we have striven to optimize data transmission within constrained IoT networks. This approach enables us to dynamically adapt the data routing path, bypassing congested nodes and selecting less congested or more capable nodes for data transmission. This way, we have aimed to minimize the impact of congestion, reduce packet loss, and enhance the overall efficiency of data transmission in the network.

In summary, our work emphasizes the importance of congestion-aware data transmission in addressing the resource consumption of constrained IoT devices. By introducing a hop-to-hop approach, we have selected the next hop for data transmission based on congestion levels and other relevant parameters. This framework aims to optimize data transmission efficiency, minimize the impact of congestion, and enhance the overall performance of IoT networks.

5.1. Introduction

This chapter aims to address the issue of resource consumption in constrained IoT devices by introducing congestion-aware data transmission techniques. The primary objective is to reduce the overhead associated with delayed packet retransmission and packet loss. By minimizing these inefficiencies, the network have ensured optimal resource utilization and prevent the strain on constrained IoT devices.

We have employed an improved Analytic Hierarchy Process (AHP) approach to select the most suitable node based on multiple parameters such as the distance, hop count, residual energy, link quality and buffer occupancy. This approach enables efficient hop-to-hop data communication while considering congestion levels and network efficiency. This process will continue this the node reaches the destination.

5.2. Proposed Approach

During the hop-to-hop transmission process, the next node to be selected is chosen based on minimizing the probability of congestion. The selection of the best suitable node involves considering multiple parameters, and their prioritization is achieved using an improved Analytical Hierarchy Process (AHP) approach.

When a node wants to transmit a data packet, it sends a request to its neighbouring nodes within its communication range to obtain the parameter values. The AHP model is then utilized to determine the weights for all the different parameters. Based on these weights, the efficiency probability of each node is calculated to identify the best suitable neighbour node.

This selected node is used as the next hop for the data packet until it reaches its destination. The primary focus of this approach is to enable congestion-aware data transmission while ensuring network efficiency is taken into account. By optimizing the choice of neighbour nodes based on congestion probabilities and network efficiency, the proposed approach aims to enhance the overall performance of data transmission in the network.

5.2.1. Selection Parameters

The parameters considered in the present approach are distance, hop count, residual energy, link quality and buffer occupancy.

1. **Distance**

The difference in the distance between neighbour nodes is compared to find the most suitable node among the list of neighbours. To find the distance, Euclidean distance is measured between the source and every possible neighbour.

$$\big((x, y)(r, s)\big) = \sqrt{(x - r)^2 - (y - s)^2} \tag{5.1}$$

Where, (x,y) and (r,s) represent the position of the source node and one of the neighbour nodes respectively. D is the distance between them.

2. **Hop Count**

The number of hops needed to transmit the packet from a node to an IoT gateway and is defined by equation (5.2).

$$hopcount(X) = \frac{\text{distance from source to IOT gateway}}{\text{transmission range}} \tag{5.2}$$

Additionally, the count of hops needed from neighbours needs to be less than X. The neighbour node which reduces the count of hops will be considered most.

3. **Residual energy**

The node shows the lifetime with the help of residual energy. The node with the higher residual energy will survive more in the network and the chances of the node being more to be selected as the next neighbour node for the data packet transmission. To measure the residual energy (RE), initial energy (IE) is subtracted from the utilized energy (UE).

$$RE_P = IE_P - UE_P \tag{5.3}$$

4. **Buffer occupancy Ratio**

The memory is required to hold the packet before it may be forwarded to next node so that the difference in transmission speed and receiving speed may be managed. This memory is named as a buffer. A node consists of some amount of buffer to store and hold the packet so that the losses of the packet may be reduced. The Buffer occupancy Ratio is the ratio of the currently occupied buffer $N_P$and the actual size of the buffer. If the buffer is highly occupied mean, it may soon get overflowed and cause packet loss.

$$BO_R = \frac{N_P}{Q_S} \tag{5.4}$$

5. Expected Transmission

The expected transmission count parameter indicates the quality of the link between nodes as it shows the count of transmission needed to deliver the packet successfully. The lower value of ETX indicates the better quality of the link for the transmission as it took a lesser number of transmissions.

$$ETX_{i,j} = \frac{1}{P_{data_{ij}} - P_{ack_{ij}}} \tag{5.5}$$

Where represents the data packet probability of successful delivery from (i) to (j) and represents the acknowledgement packet's successful delivery from (j) to (i).

Improved AHP model

AHP (Analytic Hierarchy Process) is a divide-and-conquer methodology that helps to determine the relative importance of various parameters needed for decision-making. As the name suggests, it incorporates the use of a hierarchical structure where the relative importance of sub-elements has been determined by comparing them to their parent elements.

The basic steps of Improved AHP for the congestion-aware multiple parameter-based node are:

Step 1: The parameters used to calculate the next best neighbour are represented as yi where i range equals 1, 2, 3, 4 and 5 denoting buffer occupancy, ETX, hop count, residual energy and distance respectively.

Step 2: Subsequently a judgement matrix is established. A judgement matrix has been used to find relation between variables quantitatively. Let the judgement matrix M be represented as:

$$M = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \cdots & \cdots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{bmatrix} \tag{5.6}$$

Each element, m_{ij}= y_i/y_j (i and j are index variables), indicates the importance of the degree of variable estimating the probability of the event.

The Saaty's 9-Unit scale is used to determine elements m_{ij}as follows:

If y_i and y_j have equal importance, then $m_{ij}$and m_{ji} are both equal to 1. If y_i is little more dominating than y_j, then m_{ij} = 3 and m_{ji} = ⅓.

It is obviously more dominating than y_j, then m_{ij} = 5 and m_{ji} = 1/5. If y_i is much more dominating than y_j, then m_{ij} = 7 and m_{ji} = 1/7.

If y_i is absolutely more dominating than y_j, then m_{ij} = 9 and m_{ji} = 1/9.

If these important relationships for y_i and y_j are present between the relationships given above, m_{ij} choose as 3,5,7 and 9. i.e., any integer between 1 and 9 could be chosen.

Step 3: Following this, the consistency of judgement matrix will be checked.

The improved AHP computes the optimal weight using multiple judgement matrices for each element for the proposed approach. To dodge enormous fallacy between weights, the consistency of M ought to be checked.

If the judgement matrix M is completely consistent $m_{ij} = 1; \sum_{i=1}^{n} \lambda_i = \sum_{i=1}^{n} n\,; m_{ij} = n$ (λ is the eigenvalue of the judgment matrix M), a unique nonzero $\lambda = \lambda_{max}$exists. If judgement matrix is inconsistent, $\lambda_{max} \geq n$.

Now,

$$\sum_{i\neq max}^{n\Sigma} \lambda_i + \lambda_{max} = \sum_{i=1}^{n} n \tag{5.7}$$

Then,

$$\lambda_{max} - n = -\sum_{i\neq max}^{n\Sigma} \lambda_{i_{max}} \tag{5.8}$$

The mean value is used as index to validate the consistency of judgement matrix:

$$C.I = \frac{\lambda_{\max^-}}{n-1} = \frac{-\sum_{i=\max} \lambda_{\max}}{n-1} \tag{5.9}$$

When $\lambda_n = n$ and C.I = 0, M could be exactly consensus judged. The bigger the C.I is, the more awful the consistency of the judgment matrix will be. It is just necessitated that the consistency of the framework be sensible if C.I. ≤ 0.1.

The judgment matrix's dimension affects how consistent it is. Therefore, it becomes vital to loosen the consistency constraint while working with high-dimensional judgment matrices. In these circumstances, an adjusted Random Index (RI) might be employed to quantify the consistency of the judgment matrix instead of the more significant Consistency Ratio. It is commonly accepted that the judgment matrix displays satisfactory consistency when C.R. is less than 0.1. In situations when the judgment matrix contains a lot of dimensions, this enables a more realistic and trustworthy assessment of consistency.

Step 4: We have computed the weight intervals using the eigenvector approach. The weight of each element is calculated using the eigenvector techniques based on the judgment matrix that was previously established. The intervals indicating the weights of each constituent are created from these determined weights. These intervals are made up of separate weight points that show the relative importance of each component inside each interval.

After randomly choosing a sensible judgment matrix, a weight of variable is be determined by the eigenvector strategy. The computation formulations are provided in equation (5.10) to (5.12):

$$Z_{i,k} = \prod_{j=1}^{n} c_{ij,k} \tag{5.10}$$

$$W_{i,j} = \sqrt[n]{Z_{i,j}} \tag{5.11}$$

$$\vec{W_{i,k}} = \frac{W_{i,k}}{\sum_{i=1}^{n} W_{i,k}} \tag{5.12}$$

where i, represents the variable index (i=1, 2, 3, 4, 5) and k is the index of sensibly chosen judgment matrix$(k \in N^{+})$.$\vec{W_{i,k}}$is a weight value provided to ith index under the kth judgment matrix? Thus, there are k weights for every variable i. The k weights of every factor obtained are written as intervals, constituting to the weight interval for a variable. For any variable i, choosing k judgment matrices, a weight interval i may be acquired by utilizing equations and the interval contains k weights.

Step 5: Finally obtaining the optimal weights.

The final assessment result will more accurately represent the real conditions since the ideal weight considers all weights within the weight period. At the same time, the best weight combination effectively saves the tiresome task of calculating all possible combinations and makes it easier to find acceptable neighbour nodes. Consider the reasonable judgment matrix M = (M ∈ N+) and consider Wi as the weight for variable i. The necessary task is as equation (5.13):

$$min \sum_{i=1}^{5} \sum_{k=1}^{M} \left(\vec{W}_{i,j} - w_i\right)^2 \qquad (5.13)$$

The constraint condition for function will be:

$$\sum_{i=1}^{5} w_i = 1 \qquad (5.14)$$

On the above formulation basis, the optimal weight for the ith variable could be written as w_i^*.

6. **Process of Proposed Idea:**

In this research, we have introduced a congestion-aware data transmission approaches that employs hop-to-hop transmission to relay data to the IoT gateway. The primary objective of this approach is to achieve efficient data transfer while considering the congestion levels of nodes. To accomplish this, hop-to-hop transmission is utilized, which selects the next hop based on node efficiency and congestion, considering various parameters. The efficiency and congestion level of the network is determined by factors such as distance, direction, and residual energy, as well as link quality and buffer occupancy.

When a node intends to transmit a packet to the IoT gateway, it broadcasts a "hello" message to neighbouring nodes within its communication range. Nodes that receive this message and meet the initial threshold values respond by sharing their residual energy, buffer occupancy, link quality (ETX), location, and hop count with the sender of the "hello" message. The process of selecting the best next hop continues until the packet reaches the IoT gateway.

In the proposed model, IoT devices (nodes) are randomly deployed in the network, constrained by limited resources and battery power. All nodes communicate with the IoT gateway, and they are aware of their own location as well as the IoT gateway's location. The buffer size, initial energy, and communication range are standardized for all nodes. Nodes also computes their residual energy, buffer occupancy, and link quality at any given point.

The selection of the best suitable node is achieved using the improved Analytic Hierarchy Process (AHP) method, which provides better accuracy in assessing the impact of parameters compared to traditional AHP. The AHP model calculates weights for different parameters, and based on these weights, the probability of node efficiency for being the best suitable neighbour node is determined. This approach focuses on providing congestion-aware data transmission while optimizing network efficiency. For congestion awareness, buffer occupancy and expected transmission count are considered, while hop count, residual energy, and distance are considered for efficiency.

Once the optimal weights for each parameter are obtained using the improved AHP method, the node value is calculated as the sum of multiple parameter values with their corresponding weights. The node with the highest value is chosen as the next best neighbour node, and the data packet is forwarded to it. This process continues until the packet reaches the destination, which is the IoT gateway node. The resulting path is a congestion-aware optimal path in the mobile IoT networks, considering the nodes' movements at a lower rate.

Setup Phase

In our proposed model, IoT devices (nodes) are distributed randomly throughout the network. These nodes operate with constrained resources and have limited capabilities. We have assumed that the nodes are battery-powered and are equipped with energy status awareness. Each deployed node communicates information to the IoT gateway, which is located at a predefined position. Additionally, all nodes are aware of their own location and the location of the IoT Gateway. The buffer size, initial energy, and communication range are standardized across all nodes. Furthermore, the nodes possess the ability to calculate their residual energy, buffer occupancy, and link quality at any given moment.

Initialization Phase

When a node in the network intends to transmit a packet to the IoT gateway, it initiates the process by broadcasting a 'hello' control message containing initial threshold values to its neighbouring nodes within its transmission range. Upon receiving the 'hello' control packet, the neighbouring node checks if it lies within the initial threshold values. If the threshold conditions are met, the neighbour node shares its residual energy, buffer occupancy, link quality (ETX), location, and hop count with the sender of the 'hello' control message. This exchange of information enables the creation of an initial set of next hops based on the applied threshold for various parameters. These threshold values are then broadcasted to the communication range of the neighbour nodes using the 'hello' control packet.

The threshold value for residual energy is set at 20% of the initial energy. If a node's remaining energy falls below 20% of the initial energy, it is considered incapable of forwarding the packet, even if it is not a dead node. Additionally, the hop count condition requires that a node must be able to reduce the hop count needed for packet transmission. If a node cannot achieve a reduced hop count, it is not considered as a potential next hop, as moving in the opposite direction or at the same level would consume unnecessary energy and introduce delays in packet transmission. The hop count of the neighbour node must be one less than the hop count of the sender node.

AHP Phase

The process of selecting the most suitable node by considering multiple parameters and prioritizing them is accomplished using the improved Analytic Hierarchy Process (AHP) method, which offers superior accuracy compared to traditional AHP in parameter impact assessment. In this approach, the AHP model is employed to derive weights for the various parameters. These weights are then used to calculate the probability of a node's efficiency in becoming the best suitable neighbour node. The primary focus of this approach is to ensure congestion-aware data transmission while taking into account network efficiency.

For congestion awareness, the approach considers buffer occupancy and expected transmission count, while for efficiency, it takes into account hop count, residual energy, and distance as key factors. By assigning optimal weights to each parameter using the improved AHP method, the node value is calculated as the summation of the parameter values multiplied by their respective weights. The node with the highest value is considered the next best neighbour node, and the data packet is forwarded to that node.

This process continues until the packet reaches its destination, i.e., the IoT gateway node, in our approach. As a result, an optimal congestion-aware path is established in the mobile IoT networks, where the assumption is that nodes are moving at a lower rate. This approach ensures efficient data transmission with congestion avoidance capabilities, leading to improved performance and reliability in the network.

5.3. Simulation Results and Comparative Analysis:

The proposed technique was evaluated and analyzed through network simulations using MATLAB, which offers comprehensive capabilities for efficient implementation. In the simulation, nodes were randomly dispersed throughout an area, each equipped with the same initial energy and buffer size. The proposed approach was compared against various RPL-based approaches, including CoAR, RPR, CAFOR, and CQARPL, all specifically designed to address congestion-related issues.

Several key performance metrics were considered to demonstrate the superiority of the proposed strategy over existing methods. These metrics included network throughput, buffer overflow rate, average end-to-end delay for packet transfers, and packet delivery ratio. Through a thorough comparison of these metrics between the suggested approach and the RPL-based solutions, the performance enhancements and benefits of the proposed approach were carefully evaluated and determined.

The simulated network comprised 1000 nodes in a 1000×1000 m^2 area, with nodes randomly deployed and capable of mobility. The node's transmission range was set to 250 m, and its buffer capacity allowed it to hold up to 50 packets.

Table 5.1 Parameters Value for Simulation

Parameter	**Value**
Number of Nodes	1000

Area	1000×1000 m²
Node Deployment	Random
Node Transmission Range	250 m
Buffer Capacity	50 packets
Initial Energy Value	5 joules
Data Rate	250 kbps
Packet Size	250 bytes

All nodes had the same initial energy value of 5 joules, and data rate and packet size were set at 250 kbps and 250 bytes, respectively as mentioned in the Table 5.1.

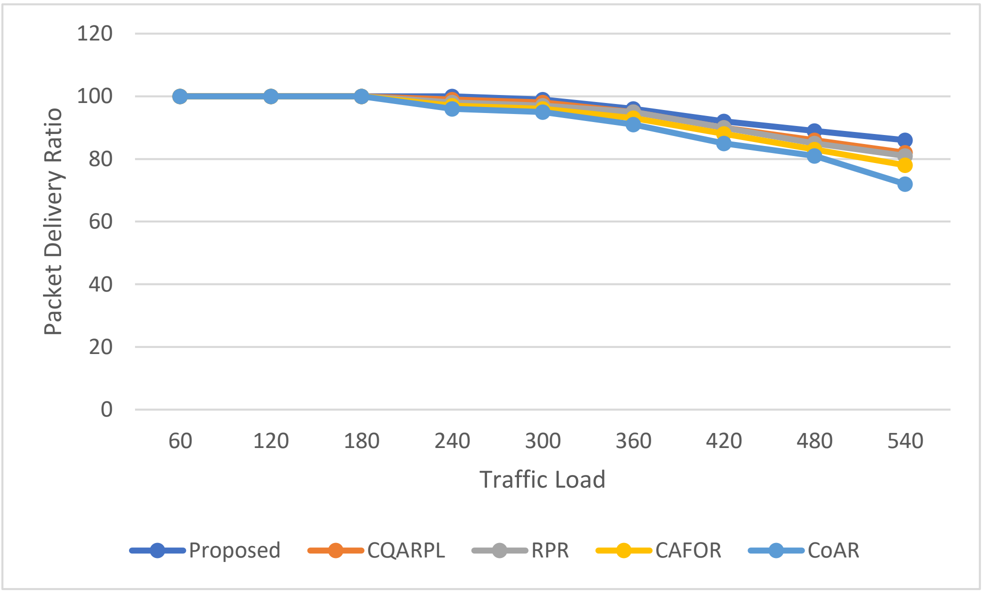

Figure. 5.1. Packet Delivery Ratio with Traffic Load

In Figure. 5.1, it is evident that the proposed approach achieves a higher packet delivery ratio compared to existing methods. Specifically, the proposed approach demonstrates a 3% improvement in packet delivery ratio over CQARPL, and even higher improvements of 3.5%, 6%, and 7.5% over RPR, CAFOR, and CoAR, respectively. The packet delivery ratio reflects the effectiveness of data transmission in the network, taking into account factors like packet loss and delay. It is noteworthy that initially when the node count is low, the delivery ratio remains similar for all approaches. However, as network traffic increases, the delivery ratio decreases in the compared approaches.

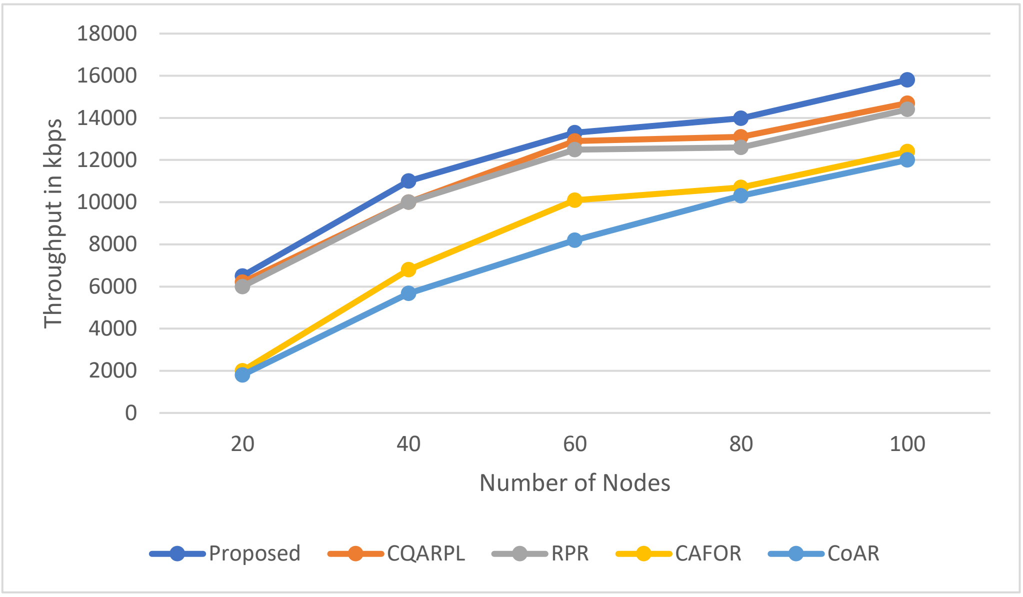

Figure. 5.2. Throughput of Network

The proposed approach significantly enhances network throughput by efficiently delivering packet bits to the gateway. As shown in Figure. 5.2, the network throughput (kbps) is influenced by the traffic load and the size of the network. Higher network density leads to increased traffic and subsequently reduces throughput.

However, the proposed approach demonstrates minimal impact on throughput in comparison to CQARPL, RPR, CoAR, and CAFOR. In fact, the throughput of the proposed approach is 4% better than the compared approaches.

The end-to-end delay is a critical factor in the IoT networks, as it directly impacts the usefulness of packet delivery. Therefore, reducing the end-to-end delay is essential. In the proposed approach, efforts were made to select the best suitable node by considering distance, hop count, and congestion value to minimize the end-to-end delay.

In Figure. 5.3, the average end-to-end delay is presented for different node counts in the network. The delay values are found to be 12% higher for CoAR, 10% higher for

CAFOR, 4% higher for RPR, and 3% higher for CAQRPL compared to the delay caused by the proposed approach.

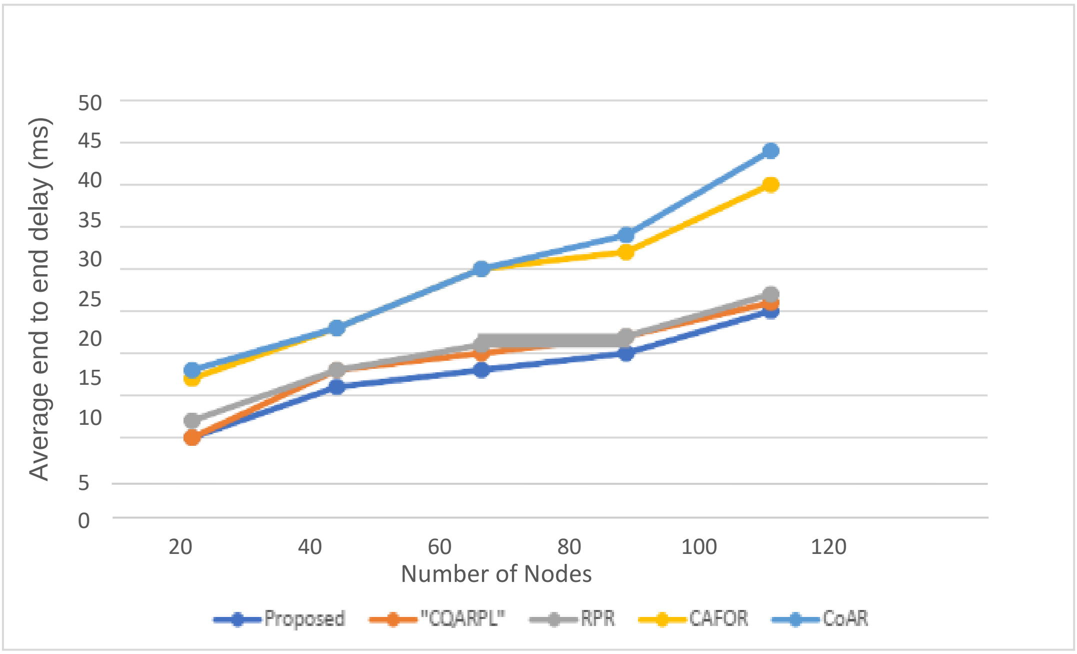

Figure. 5.3. Average End-to-End Delay (ms)

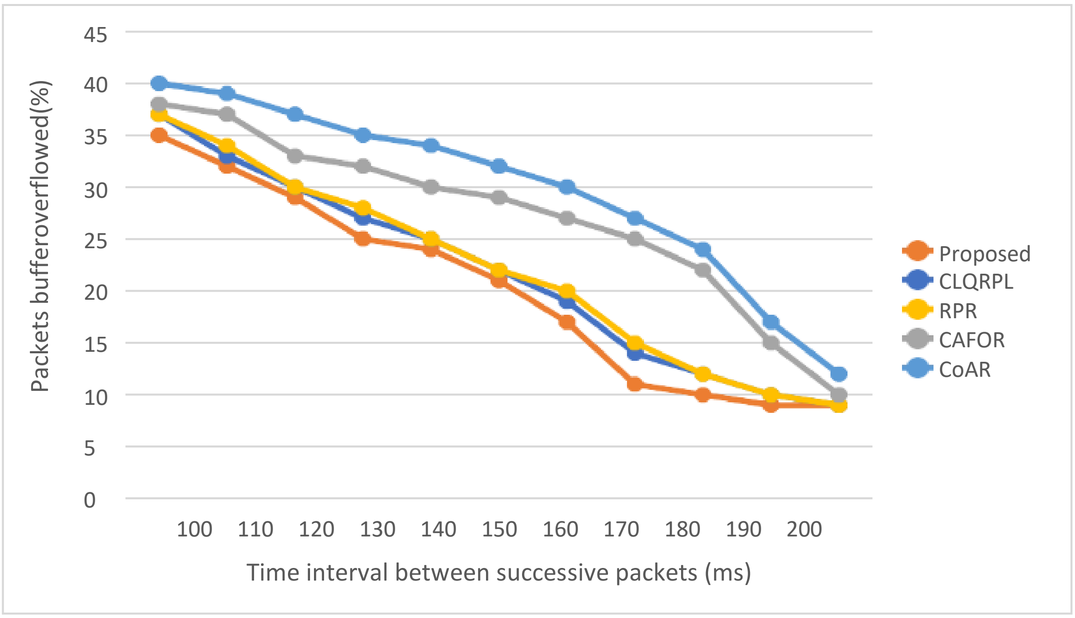

Figure. 5.4. Packet Overflow Percentage

This indicates that the proposed approach outperforms the other methods in terms of reducing end-to-end delay in the IoT networks. The rate of buffer overflow causes the packet loss in the network and it indicates the congestion of the node. The overflow occurs when the node is unable to handle the packet as the node transmission is slow as compared to receiving rate. It also occurs when a huge number of nodes are sent through the same node and resulting in congestion over the node. The buffer overflow is handled by selecting the next hop wisely.

5.4. Summary

In the chapter, a congestion-aware data transmission approach was provided for mobile IoT networks to address concerns with packet loss and delivery delay caused by congestion. The proposed approach used weighted criteria identified by the AHP model analysis to dynamically choose the best neighbour node. This approach, especially in high-traffic networks, greatly improved the packet delivery ratio, lowered end-to-end delay, and boosted total throughput by avoiding crowded nodes. The results of the proposed approach are compared with CoAR, RPR, CAFOR, and CQARPL methods, and the performance of the proposed method was better in terms of throughput by 6%, packet delivery rate by 7.5%, and reduced average delay by 4% as well as buffer overflow condition by 2%.

CHAPTER-6

CONCLUSION AND FUTURE SCOPE

In this comprehensive thesis, we have delved into the critical issue of congestion handling in IoT networks, presenting three novel mechanisms across chapters 3, 4, and 5. These mechanisms address different aspects of congestion handling and offer valuable insights into improving network performance, reducing packet loss, minimizing delays, and optimizing resource utilization.

Chapter 3 introduced the DNN-RBM (Deep Neural Network-Restricted Boltzmann Machine) approach, which leverages the dependence of the Restricted Boltzmann Machine algorithm on a deep neural network to predict congested packets in WSN-based IoT. The performance of the DNN has improved by optimizing the weight values using the RBM algorithm. In this proposed DNN- RBM, congestion window, throughput, propagation delay, RTT, number of packets sent, and packet loss are given as input. By utilizing these input factors, congestion in the nodes is predicted. The exhibition of the proposed DNN-RBM has been assessed regarding throughput, packet loss, RTT, and buffer occupancy. The Accuracy of Different models is predicted for which the DNN-RBM model is 90%, the DNN-GA model is 89% and the ANN model is 85%. Likewise, the presentation of the proposed DNN-RBM has contrasted that of the DNN-GA and DNN. As of now, a robust algorithm is also required to automatically change the threshold values depending on the network type for future work congestion handling schemes.

In Chapter 4, we have proposed a congestion control approach based on the offloading mechanism in RPL where we have proposed the mechanism to offload the load of the congested node to its neighbour which is having minimum distance and lower congestion value. The new metrics congestion value is calculated based on multiple parameters i.e., energy consumption, buffer occupancy, expected transmission count and Number of attached downward nodes. The node is considered congested by considering past traffic trends and its buffer occupancy.

Once the node is predicted congested, the congestion value and distance for all the neighbours in its range and level are compared to decide which neighbour node is suitable for offloading the buffer load and continuing the work without changing the initial DODAG. To calculate the weights of each parameter linear correlation and standard deviation are used. This weight shows the impact of the parameter of the congestion value. The proposed approach enhances the throughput and packet-receiving ratio as compared to the RPL, CoAR and CAFOR. The proposed approach also reduces the packet loss rate, delay, and energy consumption by using the initially designed DODAG and balancing the load of congested nodes by offloading.

Chapter 5 focused on a congestion-aware data transmission approach for mobile IoT networks. The approach is designed for the network where the nodes are constrained and timely delivery of the packet is crucial. In the existing approaches, routing in IoT networks is less effective when the network gets congested and congestion control measures are taken after the occurrence of congestion. The approach proposed in this study considers the problems of packet loss and delays in delivery which are majorly caused by congestion in the network. When the node wants to send a packet, it selects the best suitable next neighbour node reactively based on the various factors which are weighted and analyzed by the AHP model. In the AHP model, the parameters are arranged in the hierarchy based on importance, where buffer occupancy is at the top importance. In the presented approach, congestion is avoided by picking the best suitable node dynamically instead of having the prefixed parent node which may be congested and slow down the process. The results from the simulation of the network show that the proposed approach worked well for the high-traffic network and provided a higher packet delivery ratio as well as the minimum end-to-end delay in the network. The overall throughput is also increased by using the proposed approach. The approach may be further improved by prioritizing the packet to delivery at a higher speed. The approach may be experimented with in real-time scenarios to achieve better results and benefits in real IoT network communication.

In conclusion, our thesis contributed comprehensive insights and novel solutions to the challenging problem of congestion handling in IoT networks. The three mechanisms presented, including the DNN-RBM model, offloading mechanism in RPL, and congestion-aware data transmission approach, collectively offered valuable contributions to network performance enhancement, congestion mitigation, and resource optimization.

The use of multiple parameters like buffer occupancy, ETX, node load, and energy level results in enhancing accuracy for congestion prediction however it is also observed that parameter like packet loss is not very effective for the congestion perdition as there are more false predictions. In this thesis, our focus is on performing congestion-avoiding actions that result in the enhancement of overall network performance. These findings serve as a significant foundation for future advancements in congestion handling schemes, ultimately facilitating reliable and efficient operations of IoT networks across various application domains.

This thesis also opens up several avenues for future research and advancements in congestion handling in IoT networks. One potential future scope is the exploration of more robust algorithms that may automatically adapt threshold values based on the specific characteristics and requirements of different network types. This would enable more dynamic and efficient congestion-handling schemes tailored to the unique needs of various IoT applications. Additionally, further investigation may be conducted to optimize the offloading mechanism in RPL by considering additional metrics and factors that impact congestion, such as network traffic patterns and node mobility.

Moreover, future studies may focus on enhancing the proposed congestion-aware data transmission approach by incorporating priority-based packet delivery mechanisms, ensuring the timely and efficient delivery of critical data. Real-time experimentation in diverse IoT network scenarios would also be beneficial to validate the effectiveness and performance of the proposed solutions. By addressing these future research areas, we may continue to advance congestion-handling techniques, enabling reliable and optimized operations of IoT networks across a wide range of applications.

REFERENCES

[1]. Li, X., He, W., & Li, S. (2014). Internet of things in industries: A survey. IEEE Transactions on Industrial Informatics, 10(4), 2233–2243. https://doi.org/10.1109/TII.2014.2300753

[2]. Gungor, V. C., & Hancke, G. P. (2009). Industrial wireless sensor networks: Challenges, design principles, and technical approaches. IEEE Transactions on Industrial Electronics, 56(10), 4258–4265.

[3]. Kushalnagar, N., Gabriel, M., & Christian, S. (2007). IPv6 over low-power wireless personal area networks (6LoWPANs): Overview, assumptions, problem statement, and goals. (pp. 1–11).

[4]. Kim, H.-S., Im, H., Lee, M.-S., Paek, J., & Bahk, S. (2015). A measurement study of TCP over RPL in low power and lossy networks. Journal of Communications and Networks, 17(6), 647–655. https://doi.org/10.1109/JCN.2015.000111.

[5]. Stojkoska, B. R., & Trivodaliev, K. (2017). Enabling the Internet of Things for smart homes through fog computing. In 2017 25th Telecommunication Forum (TELFOR) (pp. 1-4). IEEE.

[6]. Deore, R., Sonawane, V., & Satpute, P. (2015). Internet of Thing Based Home Appliances Control. In 2015 International Conference on Computational Intelligence and Communication Networks (CICN).

[7]. Govindraj, V., Sathiyanarayanan, M., & Abubakar, B. (2017). Customary homes to smart homes using the Internet of Things (IoT) and mobile applications. In 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon) (pp. 1059-1063). IEEE.

[8]. Rauscher, J., & Bauer, B. (2018). Safety and Security Architecture Analyzes Framework for the Internet of Things of Medical Devices. In 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom) (pp. 1-3). IEEE.

[9]. Laplante, P. A., & Laplante, N. (2016). The internet of things in healthcare:

Potential applications and challenges. IT Professional, 18(3), 2-4.

[10]. Chiuchisan, I., Chiuchisan, I., & Dimian, M. (2015). Internet of Things for e-Health: An approach to medical applications. In 2015 International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM) (pp. 1-5). IEEE.

[11]. Gupta, K., & Shukla, S. (2016). Internet of Things: Security challenges for next-generation networks. In 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH) (pp. 315-318). IEEE.

[12]. Qiu, T., Chen, N., Li, K., Atiquzzaman, M., & Zhao, W. (2018). How can heterogeneous internet of things build our future: A survey. IEEE Communications Surveys & Tutorials.

[13]. Almeida, V. A., Doneda, D., & Monteiro, M. (2015). Governance challenges for the Internet of Things. IEEE Internet Computing, 19(4), 56-59.

[14]. Terminology for constrained node networks. (2014). Retrieved from http://www.ietf.org/rfc/rfc7228.txt

[15]. Hahm, O., Baccelli, E., Petersen, H., & Tsiftes, N. (2015). Operating systems for low-end devices in the internet of things: A survey. IEEE Internet of Things Journal, 3(5), 720- 734.

[16]. Stankovic, J. (2014). Research directions for the Internet of things. IEEE Internet Things Journal, 1(1), 3-9. doi:10.1109/JIOT.2014.2312291.URL http://ieeexplore.ieee.org/xpls/absall.jsp?arnumber=6774858

[17]. Ghaffari, A. (2015). Congestion control mechanisms in wireless sensor networks: A survey. Journal of Network and Computer Applications, 52, 101-115.

[18]. Reena, P., & Jacob, L. (2007). Hop-by-hop versus end-to-end congestion control in wireless multi-hop UWB networks. In Advanced Computing and Communications, 2007. ADCOM 2007. International Conference on (pp. 255–261). IEEE.

[19]. Al-Kashoash, H. A., Kharrufa, H., Al-Nidawi, Y., & Kemp, A. H. (2018). Congestion control in wireless sensor and 6lowpan networks: Toward the internet of things. Wireless Networks, 1–30.

[20]. Al-Kashoash, H. A., Hafeez, M., & Kemp, A. H. (2017). Congestion control for 6lowpan networks: A game theoretic framework. IEEE Internet of Things Journal, 4(3), 760–771.

[21]. ITU internet reports. (November 2005). The internet of things. Retrieved from https://www.itu.int/net/wsis/tunis/newsroom/stats/The-Internet-of-Things-2005.pdf

[22]. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys & Tutorials, 17(4), 2347–2376.

[23]. Andrioli, L., da Rosa Righi, R., da Costa, C. A., & Graebin, L. (2017). Observing network performance and congestion on managing assets with RFID and cloud computing. Journal of Computer and Communications, 5(09), 43.

[24]. Ding, K., Jiang, P., & Jiang, P. (2018). RFID-based production data analysis in an IoT- enabled smart job shop. IEEE/CAA Journal of Automatica Sinica, 5(1), 128–138.

[25]. Ok, F., Can, M., Uc¸gün, H., & Y ¨uzgec¸, U. (2017). Smart mirror applications with Raspberry Pi. In Computer Science and Engineering (UBMK), 2017 International Conference on (pp. 94–98). IEEE.

[26]. Gupta, M. S. D., Patchava, V., & Menezes, V. (2015). Healthcare based on IoT using Raspberry Pi. In Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on (pp. 796–799). IEEE.

[27]. Gigli, M., & Koo, S. G. (2011). Internet of things: Services and applications categorization. Advances in Internet of Things, 1(2), 27–31.

[28]. Vandana, C., & Chikkamannur, A. A. (2019). Semantic ontology-based IoT-resource description. International Journal of Advanced Networking and Applications, 11(1), 4184–4189.

[29]. Web Ontology Language (OWL). (2012). Retrieved from https://www.w3.org/OWL/ [30]. Wajahat, H., & Kim, H. S. (2014). Efficient XML interchange for automated demand

response in smart grid networks. In Communications and Information Technologies (ISCIT), 2014 14th International Symposium on (pp. 398–399). IEEE.

[31]. Akpakwu, G. A., Silva, B. J., Hancke, G. P., & Abu-Mahfouz, A. M. (2018). A survey on 5G networks for the internet of things: Communication technologies and challenges. IEEE Access, 6, 3619–3647.

[32]. Mocnej, J., Pekar, A., Seah, W. K., & Zolotova, I. (2018). Network traffic characteristics of the IoT application use cases. School of Engineering and Computer Science, Victoria University of Wellington.

[33]. Al-Sarawi, S., Anbar, M., Alieyan, K., & Alzubaidi, M. (2017). Internet of things (IoT) communication protocols. In Information Technology (ICIT), 2017 8th International Conference on (pp. 685–690). IEEE.

[34]. Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F., & Alonso-Zarate, J. (2015). A survey on application layer protocols for the internet of things. Transactions on IoT and Cloud Computing, 3(1), 11–17.

[35]. Internet of Things - Architecture. (n.d.). Retrieved from www.iot-a.eu. [36]. IoTivity. (n.d.). Retrieved from https://iotivity.org/

[37]. Chen, S., Xu, H., Liu, D., Hu, B., & Wang, H. (2014). A vision of IoT: Applications, challenges, and opportunities with China perspective. IEEE Internet of Things Journal, 1(4), 349-359.

[38]. Mahmoud, R., Yousuf, T., Aloul, F., & Zualkernan, I. (2015). Internet of things (IoT) security: Current status, challenges and prospective measures. In 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST) (pp. 336–341). IEEE.

[39]. Somani, A. K., & Vaidya, N. H. (1997). Understanding fault tolerance and reliability.

Computer, 30(4), 45–50.

[40]. Power, A., & Kotonya, G. (2019). Complex patterns of failure: Fault tolerance via complex event processing for IoT systems. In 2019 International Conference on

Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (pp. 986–993). IEEE.

[41]. Hasan, M. Z., & Al-Turjman, F. (2017). Optimizing multipath routing with guaranteed fault tolerance in internet of things. IEEE Sensors Journal, 17(19), 6463–6473.

[42]. Ren, J., Guo, H., Xu, C., & Zhang, Y. (2017). Serving at the edge: A scalable IoT architecture based on transparent computing. IEEE Network, 31(5), 96–105.

[43]. Chun, S.-M., Kim, H.-S., & Park, J.-T. (2015). CoAP-based mobility management for the internet of things. Sensors, 15(7), 16060–16082.

[44]. Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A survey on security and privacy issues in internet-of-things. IEEE Internet of Things Journal, 4(5), 1250–1258.

[45]. Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., & Faruki, P. (2019). Network intrusion detection for IoT security based on learning techniques. IEEE Communications Surveys & Tutorials.

[46]. Faruki, P., Buddhadev, B., Shah, B., Zemmari, A., Laxmi, V., & Gaur, M. S. (2019). Droiddivesdeep: Android malware classification via low-level monitorable features with deep neural networks. In International Conference on Security & Privacy (pp. 125–139). Springer.

[47]. Santos, D. F., Almeida, H. O., & Perkusich, A. (2015). A personal connected health system for the internet of things based on the constrained application protocol. Computers & Electrical Engineering, 44, 122–136.

[48]. Farsi, M., Badawy, M., Moustafa, M., Ali, H. A., Abdulazeem, Y. (2019). A congestion- aware clustering and routing (CCR) protocol for mitigating congestion in WSN. IEEE Access, 7, 105402–105419.

[49]. Winter, T., Thubert, P., Brandt, A., Clausen, T., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, J. P. (2011). RPL: IPv6 Routing Protocol for Low

Power and Lossy Networks. draft-ietf-roll-rpl-19.

[50]. Thubert, P., et al. (2012). Objective function zero for the routing protocol for low-power and lossy networks (RPL).

[51]. Gnawali, O., Levis, P. (2012). The minimum rank with hysteresis objective function.

RFC 6719.

[52]. Pister, K., Dejean, N., Barthel, D. (2012). Routing metrics used for path calculation in low-power and lossy networks. RFC 6551.

[53]. Levis, P., Clausen, T., Hui, J., Gnawali, O., Ko, J. (2011). The trickle algorithm. Internet Engineering Task Force, RFC 6206.

[54]. Jaffey, T. (2014). MQTT and CoAP, IoT protocols. Eclipse Newsletter.

[55]. Masek, P., et al. (2016). Implementation of true IoT vision: survey on enabling protocols and hands-on experience. International Journal of Distributed Sensor Networks, 12(4), 8160282.

[56]. Betzler, A., et al. (2013). Congestion control in reliable CoAP communication. Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems.

[57]. Balandina, E., Koucheryavy, Y., Gurtov, A. (2013). Computing the retransmission timeout in CoAP. In Internet of Things, Smart Spaces, and Next Generation Networking (pp. 352-362). Springer.

[58]. Betzler, August, et al. "Congestion control for CoAP cloud services." Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA). IEEE, 2014.

[59]. Pramanik, A., Luhach, A. K., Batra, I., Singh, U. (2017). A systematic survey on congestion mechanisms of CoAP based Internet of Things. In Advanced Informatics for Computing Research (pp. 306-317). Springer.

[60]. Betzler, A., Gomez, C., Demirkol, I. (2015). Evaluation of advanced congestion

control mechanisms for unreliable CoAP communications. In Proceedings of the 12th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks (pp. 63-70). ACM.

[61]. Ancillotti, E., Bruno, R. (2017). Comparison of CoAP and CoCoA+ congestion control mechanisms for different IoT application scenarios. In IEEE Symposium on Computers and Communications, ISCC (pp. 1186-1192). IEEE.

[62]. Hasan, H. M., Ahmed, A. I. (2018). A comparative analysis for congestion mechanism in COAP and COCOA. Engineering Technology Journal, 36(8 Part (A) Engineering), 867- 877.

[63]. Bolettieri, S., Vallati, C., Tanganelli, G., Mingozzi, E. (2017). Highlighting some shortcomings of the CoCoA+ congestion control algorithm. In International Conference on Ad-Hoc Networks and Wireless (pp. 213-220). Springer.

[64]. Balandina, E., Koucheryavy, Y., Gurtov, A. (2013). Computing the retransmission timeout in CoAP. In Internet of Things, Smart Spaces, and Next Generation Networking (pp. 352-362). Springer.

[65]. Ludwig, R., Sklower, K. (2000). The Eifel retransmission timer. ACM SIGCOMM Computer Communication Review, 30(3), 17-27.

[66]. Bhalerao, R., Subramanian, S. S., Pasquale, J. (2016). An analysis and improvement of congestion control in the CoAP Internet-of-Things protocol. In Consumer Communications & Networking Conference (CCNC), 13th IEEE Annual (pp. 889-894). IEEE.

[67]. Hassan, R., Jubair, A. M., Azmi, K., Bakar, A. (2016). Adaptive congestion control mechanism in CoAP Application Protocol for Internet of Things (IoT). In Signal Processing and Communication (ICSC), 2016 International Conference on (pp. 121-125). IEEE.

[68]. Lee, J. J., Chung, S. M., Lee, B., Kim, K. T., Youn, H. Y. (2016). Round trip time based adaptive congestion control with CoAP for sensor network. In International Conference on Distributed Computing in Sensor Systems, DCOSS (pp. 113-115). IEEE.

[69]. Järvinen, I., Raitahila, I., Cao, Z., Kojo, M. (2018). Is CoAP congestion safe? In Proceedings of the Applied Networking Research Workshop (pp. 43-49). ACM.

[70]. Vallati, C., Righetti, F., Tanganelli, G., Mingozzi, E., Anastasi, G. (2018). ECOAP: Experimental assessment of congestion control strategies for CoAP using the WiSHFUL platform. In IEEE International Conference on Smart Computing, SMARTCOMP (pp. 423-428). IEEE.

[71]. Al-Kashoash, H. A. A., Al-Nidawi, Y., & Kemp, A. H. (2016). Congestion analysis for low power and lossy networks. In 2016 Wireless Telecommunications Symposium (WTS) (pp. 1-6). IEEE.

[72]. Pancaroglu, D., & Sen, S. (2021). Load balancing for RPL-based Internet of Things: A review. Ad Hoc Networks, 116, 102491.

[73]. Al-Kashoash, H. A. A., Hassen, F., Kharrufa, H., & Kemp, A. H. (2018). Analytical modeling of congestion for 6lowpan networks. ICT Express, 4(4), 209-215.

[74]. Al-Kashoash, H. A. A., Kharrufa, H., Al-Nidawi, Y., & Kemp, A. H. (2019). Congestion control in wireless sensor and 6lowpan networks: Toward the Internet of Things. Wireless Networks, 25(8), 4493-4522.

[75]. Al-Kashoash, H. A. A., Al-Nidawi, Y., & Kemp, A. H. (2016). Congestion aware RPL for 6LoWPAN networks. In Wireless Telecommunications Symposium (WTS) (pp. 1-6). IEEE.

[76]. Kim, H. S., Kim, H., Paek, J., & Bahk, S. (2017). Load balancing under heavy traffic in RPL routing protocol for low power and lossy networks. IEEE Transactions on Mobile Computing, 16(4), 964-979.

[77]. Kim, H. S., Ko, J., Culler, D. E., & Paek, J. (2017). Challenging the IPv6 routing protocol for low-power and lossy networks (RPL): A survey. IEEE Communications Surveys & Tutorials, 19(4), 2502-2525.

[78]. Kim, H. S., Paek, J., Culler, D. E., & Bahk, S. (2017). Do not lose bandwidth: Adaptive transmission power and multihop topology control. In 13th International

Conference on Distributed Computing in Sensor Systems, DCOSS (pp. 99-108). IEEE.

[79]. Shah-Mansouri, H., Wong, V. W. S. (2018). Hierarchical fog-cloud computing for IoT systems: A computation offloading game. IEEE Internet of Things Journal, 5(4), 3246- 3257.

[80]. Michopoulos, V., Guan, L., Oikonomou, G., & Phillips, I. (2012). DCCC6: Duty cycle- aware congestion control for 6lowpan networks. In 2012 IEEE International Conference on Pervasive Computing and Communications Workshops (pp. 278-283). IEEE.

[81]. Castellani, A. P., Rossi, M., & Zorzi, M. (2014). Back pressure congestion control for CoAP/6LoWPAN networks. Ad Hoc Networks, 18, 71-84.

[82]. Al-Kashoash, H. A. A., Hafeez, M., & Kemp, A. H. (2017). Congestion control for 6LoWPAN networks: A game theoretic framework. IEEE Internet of Things Journal, 4(3), 760-771.

[83]. Hellaoui, H., & Koudil, M. (2015). Bird flocking congestion control for CoAP/RPL/6LoWPAN networks. In Proceedings of the 2015 Workshop on IoT Challenges in Mobile and Industrial Systems (pp. 25-30).

[84]. Kim, H. S., Paek, J., & Bahk, S. (2015). Qu-RPL: Queue utilization based RPL for load balancing in large scale industrial applications. In 2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON) (pp. 265-273). IEEE.

[85]. Sheu, J. P., Hsu, C. X., & Ma, C. (2015). A game theory based congestion control protocol for wireless personal area networks. In 2015 IEEE 39th Annual Computer Software and Applications Conference (Vol. 2, pp. 659-664). IEEE.

[86]. Ma, C., Sheu, J. P., & Hsu, C. X. (2016). A game theory based congestion control protocol for wireless personal area networks. Journal of Sensors, 2016.

[87]. Ha, M., Kwon, K., Kim, D., & Kong, P. Y. (2014). Dynamic and distributed load balancing scheme in multi-gateway based 6LoWPAN. In 2014 IEEE

International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) (pp. 87-94). IEEE.

[88]. Tang, W., Ma, X., Huang, J., & Wei, J. (2016). Toward improved RPL: A congestion avoidance multipath routing protocol with time factor for wireless sensor networks. Journal of Sensors, 2016.

[89]. Lodhi, M. A., Rehman, A., Khan, M. M., & Hussain, F. B. (2015). Multiple path RPL for low power lossy networks. In 2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob) (pp. 279-284). IEEE.

[90]. Al-Kashoash, H. A. A., Amer, H. M., Mihaylova, L., & Kemp, A. H. (2017). Optimization-based hybrid congestion alleviation for 6LoWPAN networks. IEEE Internet of Things Journal, 4(6), 2070-2081.

[91]. Iova, O., Theoleyre, F., & Noel, T. (2015). Using multiparent routing in RPL to increase the stability and the lifetime of the network. Ad Hoc Networks, 29, 45-62.

[92]. Qasem, M., Al-Dubai, A., Romdhani, I., Ghaleb, B., & Gharibi, W. (2016). A new efficient objective function for routing in the Internet of Things paradigm. In 2016 IEEE Conference on Standards for Communications and Networking (CSCN) (pp. 1-6). IEEE.

[93]. Lodhi, M. A., Rehman, A., Khan, M. M., & Hussain, F. B. (2015). Multiple path RPL for low power lossy networks. In 2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob) (pp. 279-284). IEEE.

[94]. Nassar, J., Gouvy, N., & Mitton, N. (2017). Towards multi-instances QoS efficient RPL for smart grids. In Proceedings of the 14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks (pp. 85-92).

[95]. Nassar, J., Berthomé, M., Dubrulle, J., Gouvy, N., Mitton, N., & Quoitin, B. (2018). Multiple instances QoS routing in RPL: Application to smart grids. Sensors, 18(8), 2472.

[96]. Ha, M., Kwon, K., Kim, D., & Kong, P. Y. (2014). Dynamic and distributed load balancing scheme in multi-gateway based 6LoWPAN. In 2014 IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) (pp. 87-94). IEEE.

[97]. Moghadam, M. N., & Taheri, H. (2014). High throughput load balanced multipath routing in homogeneous wireless sensor networks. In 2014 22nd Iranian Conference on Electrical Engineering (ICEE) (pp. 1516-1521). IEEE.

[98]. Ganesan, D., Govindan, R., Shenker, S., & Estrin, D. (2001). Highly resilient, energy- efficient multipath routing in wireless sensor networks. ACM SIGMOBILE Mobile Computing and Communications Review, 5(4), 11-25.

[99]. Michel, M., Duquennoy, S., Quoitin, B., & Voigt, T. (2015). Load balanced data collection through opportunistic routing. In 2015 International Conference on Distributed Computing in Sensor Systems (pp. 62-70). IEEE.

[100]. Nassiri, M., Boujari, M., & Azhari, S. V. (2015). Energy-aware and load-balanced parent selection in RPL routing for wireless sensor networks. International Journal of Wireless and Mobile Computing, 9(3), 231-239.

[101]. Mamdouh, M., Elsayed, K., & Khattab, A. (2016). RPL load balancing via minimum degree spanning tree. In 2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (pp. 1-8). IEEE.

[102]. Guo, J., Liu, X., Bhatti, G., Orlik, P., & Parsons, K. (2014). Load balanced routing for low power and lossy networks. U.S. Patent Application No. 13/746,173.

[103]. Sebastian, A., & Sivagurunathan, D. S. (2018). Load balancing optimization for RPL- based emergency response using Q-learning. MATTER International Journal of Science and Technology, 4(2), 74-92.

[104]. Sankar, S., & Srinivasan, P. (2018). Fuzzy logic-based energy-aware routing protocol for the Internet of Things. International Journal of Intelligent Systems and Applications, 10(10), 11.

[105]. Gaddour, O., Koubâa, A., Baccour, N., & Abid, M. (2014). OF-FL: QoS aware fuzzy logic objective function for the RPL routing protocol. In 2014 12th International

Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt) (pp. 365-372). IEEE.

[106]. Aljarrah, E. (2017). Deployment of multi-fuzzy model-based routing in RPL to support efficient IoT. International Journal of Communication Networks and Information Security, 9(3), 457-465.

[107]. Maheshwari, A., Yadav, R. K., & Nath, P. (2022). Data congestion control using offloading in IoT network. Wireless Personal Communications, 1-20.

[108]. Maheswari, A., Yadav, R. K., & Nath, P. (2022). Data congestion prediction in sensors based IoT network. Journal of Scientific and Industrial Research (JSIR), 80(12), 1091- 1095.

[109]. Shreyas, J., Singh, H., Tiwari, S., Srinidhi, N. N., & Dilip Kumar, S. M. (2021). Cafor: Congestion avoidance using fuzzy logic to find an optimal routing path in 6LoWPAN networks. Journal of Reliable Intelligent Environments, 7(4), 325-340.

[110]. Guo, H., Liu, J., Qin, H. (2018). Collaborative mobile edge computation offloading for IoT over fiber-wireless networks. IEEE Network, 32(1), 66-71.

[111]. Guo, H., Liu, J., Zhang, J., Sun, W., & Kato, N. (2018). Mobile edge computation offloading for ultradense IoT networks. IEEE Internet of Things Journal, 5(6), 4977- 4988.

[112]. Dao, N.-N., Vu, D.-N., Na, W., Kim, J., & Cho, S. (2018). SGCO: Stabilized green crosshaul orchestration for dense IoT offloading services. IEEE Journal on Selected Areas in Communications, 36(11), 2538-2548.

[113]. Lyu, X., Tian, H., Jiang, L., Vinel, A., Maharjan, S., Gjessing, S., et al. (2018). Selective offloading in mobile edge computing for the green Internet of Things. IEEE Network, 32(1), 54-60.

[114]. Lee, H.-S., & Lee, J.-W. (2018). Task offloading in heterogeneous mobile

cloud computing: Modeling, analysis, and cloudlet deployment. IEEE Access, 6, 14908-14925.

[115]. Zhang, C., Zhao, H., Deng, S. (2018). A density-based offloading strategy for IoT devices in edge computing systems. IEEE Access, 6, 73520-73530.

[116]. Hasan, R., Hossain, M., Khan, R. (2018). Aura: An incentive-driven ad hoc IoT cloud framework for proximal mobile computation offloading. Future Generation Computer Systems, 86, 821-835.

[117]. Elbamby, M. S., Bennis, M., Saad, W. (2017). Proactive edge computing in latency- constrained fog networks. In 2017 European Conference on Networks and Communications (EuCNC).

[118]. Farahani, M., Rahbar, A. G. (2019). Double Leveled Unequal Clustering with Considering Energy Efficiency and Load Balancing in Dense IoT Networks. Wireless Personal Communications, 106(3), 1183-1207.

[119]. Gheisari, S., Tahavori, E. (2019). CCCLA: A cognitive approach for congestion control in the Internet of Things using a game of learning automata. Computer Communications, 147, 40-49.

[120]. Park, Y., Kim, S. (2015). Game-based data offloading scheme for IoT system traffic congestion problems. EURASIP Journal on Wireless Communications and Networking, 2015(1), 1-10.

[121]. Kuppusamy, P., Kalpana, R., Rao, P. V. (2018). Optimized traffic control and data processing using IoT. Cluster Computing, 22(1), 2169-2178.

[122]. Hussain, A., Manikanthan, S., Padmapriya, T., Nagalingam, M. (2019). Genetic algorithm-based adaptive offloading for improving IoT device communication efficiency. Wireless Networks.

[123]. Kim, H. (2017). A load balancing scheme with Loadbot in IoT networks. The Journal of Supercomputing, 74(3), 1215-122.

LIST OF PUBLICATIONS

Papers Published in International Journals:

1. Maheshwari, A., Yadav, R. K., & Nath, P. (2023). Congestion Aware Data Transmission in Mobile and Constrained IoT Networks. Wireless Personal Communications, 130(3), 2121-2136. (SCIE-2.017)
2. Maheshwari, A., Yadav, R. K., & Nath, P. (2022). Data Congestion Control Using Offloading in IoT Network. Wireless Personal Communications, 125(3), 2147-2166. (SCIE- 2.017)
3. Maheshwari, A., Yadav, R. K., & Nath, P. (2022). Data Congestion Prediction in Sensors Based IoT Network. Journal of Scientific and Industrial Research (JSIR), 80(12), 1091-1095. (SCIE- 0.6)

Papers Published in International Conferences:

4. Maheshwari, A., Yadav, R. K., & Nath, P. (2023). Enhanced RPL to Control Congestion in IoT: A Review. In International Conference on Internet of Things (pp. 1-13). Springer, Cham.
5. Maheshwari, A., & Yadav, R. K. (2020). Analysis of congestion control mechanism for IoT. In 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 288-293). IEEE.