TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

THESIS NO: PUL075MSCSK014

Task Prioritization and Scheduling of Fog Computing Model in Healthcare Systems

by

Prakriti Pahari

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

LALITPUR, NEPAL

September, 2021

Task Prioritization and Scheduling of Fog Computing Model in Healthcare Systems

by

Prakriti Pahari

075MSCSK014

Thesis Supervisor

Prof. Dr. Subarna Shakya

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer System and Knowledge Engineering

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Tribhuvan University

Lalitpur, Nepal

September, 2021

DECLARATION

I declare that the work hereby submitted for Master of Science in Computer System and Knowledge Engineering (MSCSKE) at IOE, Pulchowk Campus entitled **"TASK PRIORITIZATION AND SCHEDULING OF FOG COMPUTING MODEL IN HEALTHCARE SYSTEMS"** is my own work and has not been previously submitted by me at any university for any academic award.

I authorize IOE, Pulchowk Campus to lend this thesis to other institution or individuals for the purpose of scholarly research.

Student Name: Prakriti Pahari
Roll Number: PUL075MSCSK014
Date: September, 2021

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled **"TASK PRIORITIZATION AND SCHEDULING OF FOG COMPUTING MODEL IN HEALTHCARE SYSTEMS"**, submitted by **Prakriti Pahari** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer System and Knowledge Engineering"**.

....

Supervisor: Prof. Dr. Subarna Shakya,
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

....

External Examiner: Dr. Dovan Rai,
Research Fellow,
Global Institute for Interdisciplinary Studies (GIIS)

....

Committee Chairperson: Assoc. Prof. Dr. Nanda Bikram Adhikari,
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

Date: September, 2021

DEPARTMENTAL ACCEPTANCE

The thesis entitled **"TASK PRIORITIZATION AND SCHEDULING OF FOG COMPUTING MODEL IN HEALTHCARE SYSTEMS"**, submitted by **Prakriti Pahari** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer System and Knowledge Engineering"** has been accepted as a bonafide record of work independently carried out by her in the department.

....

Prof. Dr. Ram Krishna Maharjan
Head of the Department
Department of Electronics and Computer Engineering
Pulchowk Campus, Institute of Engineering,
Tribhuvan University, Nepal.

ACKNOWLEDGEMENT

I express my deepest sense of gratitude to my supervisor **Prof. Dr. Subarna Shakya** for his valuable guidance and suggestions throughout the completion of this thesis. A very special and hearty thanks to program coordinator **Assoc. Prof. Dr. Nanda Bikram Adhikari** at the Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, along with the whole department and faculty members for providing us the necessary guidelines for this thesis work.

I have to present my sincere acknowledgements to Head of the Department Prof. Dr. Ram Krishna Maharjan, Assoc. Prof. Dr. Dibakar Raj Pant, Asst. Prof. Dr. Babu Ram Dawadi and other faculties for their precious comments and suggestions for the improvement of this thesis. Also, I would like to take a moment to thank my classmates for their cooperation throughout the development of this thesis.

ABSTRACT

Health-related applications are one of the most sensitive areas which should be delivered on time efficiently. For the storage and processing of enormous health data, Cloud Computing could not be efficient as Cloud Data Centers take a large time to process and send back the results. The new paradigm, called Fog Computing is applicable in cases like these. In this research, we utilize the sample time-critical healthcare system where the IoT sensors' data is divided into critical and normal tasks where critical tasks are prioritized over normal patients' data. To address their management, Fog Computing is used at the edge of the network. In this paper, a new fog-cloud-based algorithm called Prioritized Latency Aware Energy Efficient Algorithm (PLAEE) is developed by utilizing the existing algorithms in the fog system and also by process optimization of the core evaluation metrics, latency, and energy usage. This algorithm shows superiority to the existing algorithms in terms of performance metrics. The experimentation is performed using Blood Pressure data collected from the University of Piraeus. In terms of response time, the PLAEE is performing 36.40%, 14.82%, 14.70%, and 6.03% better than Cloud only, Edge-wards, Resource Aware, and SCATTER Algorithm respectively. In terms of Energy Consumption, the PLAEE is performing 23.85%, 14.96%, 10.84%, and 2.83% better than Cloud only, Edge-wards, Resource Aware, and SCATTER Algorithm respectively. Almost 98% of critical data are placed in the FNs according to the Tasks Managed value calculated where 91.70%, 6.28%, and 2.01% of Critical Tasks are placed in FZ1, FZ2, and CDC respectively.

Keywords: *Cloud Computing, Internet of Things, Fog Computing, Latency, Energy Consumption, Resource Aware, SCATTER.*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

FN	Fog Node
IoT	Internet of Things
SBP	Systolic Blood Pressure
DBP	Diastolic Blood Pressure
FSP	Fog Service Provider
FRM	Fog Resource Manager
CRM	Cloud Resource Manager
CDC	Cloud Data Center
MIPS	Million Instructions Per Second

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Healthcare is considered an important aspect of life that provides a variety of resources such as therapy, diagnosis, and prevention of disorders, disabilities, trauma, as well as psychological illness. It is found that technology closely interacts with the field of healthcare. To improve our lives, many healthcare systems are built with the aid of technology. Healthcare applications are the most crucial elements because of the fact that they directly affect the lives of patients. The one innovative model that is used in healthcare systems these days, is fog computing. Fog computing is a decentralized platform that serves at the network's edge, providing low latency, execution speed, anonymity, protection, and energy consumption, all of which are essential in healthcare applications. Because of its reduced latency and, in some situations, the surplus power of available devices, fog computing is seen to be much cost-effective than the centralized cloud paradigm. Fog Nodes (FN) should be available at the network's edge to perform the most important services locally and forward usual services to the network's core i.e., Cloud. A new article with emergency room doctors indicated that splitting focus into two areas while prioritizing the tasks was critical; time and performance. A key function is that of ensuring healthcare facilities are on time.

With the advent of the Internet of Things, much has been discussed as to how to properly execute all data processing brought on by smart devices, and this is precisely what fog computing aims to address. Its goal is to eliminate the need to transfer produced data to the cloud by processing it immediately at the device, or next to it, at the network boundary, in more capable equipment. CISCO coined the phrase "fog computing," which is described as an architecture that extends the cloud's processing and storage capabilities to the network's edge [1]. As a result, data may be gathered and processed locally, decreasing response time and bandwidth use. Reduced latency, better privacy, decreased bandwidth requirements, reliability, energy efficiency, and data security are the primary benefits cited in the literature for the fog computing paradigm. First, compared to other cloud-based designs, processing data at the network's border decreases latency since the physical distance is less. As a result, possible data center delays might be avoided. Another benefit of the fog computing idea is the ability to shift computation-intensive activities from low-power devices to higher-powered nodes [2].

The concept of Fog Computing is shown in the figure1.1.

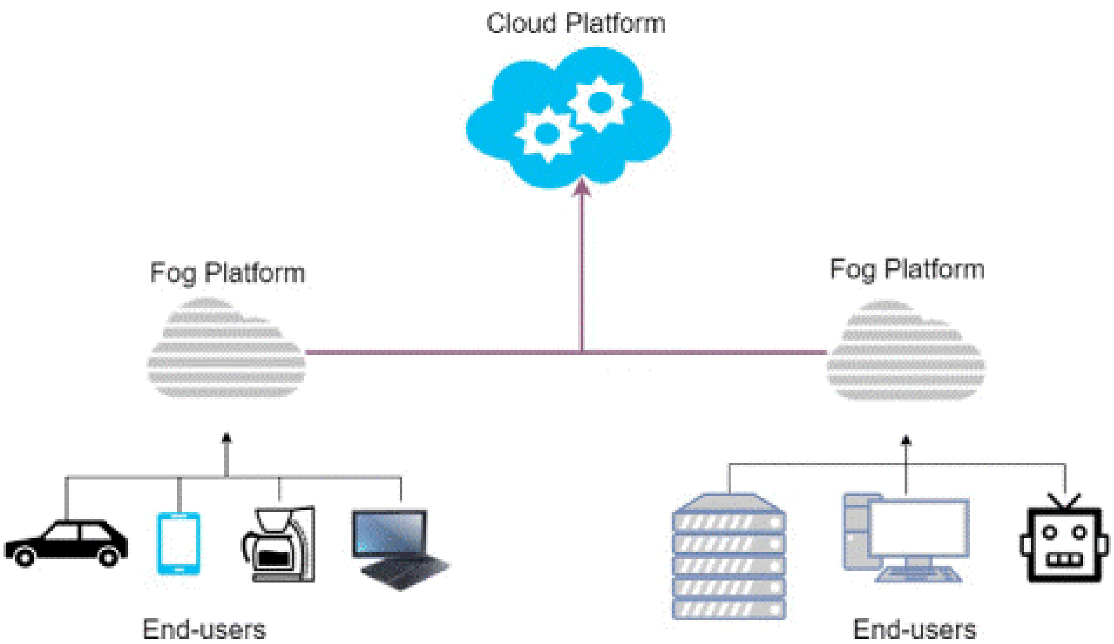

Figure 1.1: Fog Computing System

Secondly, unlike the cloud model, the fog method enables data analysis and processing on a local gateway rather than transmitting the information to the cloud, which may improve user data privacy [3]. Thirdly, because the fog approach shows data to be gathered and analyzed nearby the end-users, the amount of data sent to the cloud is decreased, resulting in lower network traffic costs. Fourth, by sharing the same functionality throughout the many distinct fog nodes, the fog paradigm can increase the system's reliability. As a result, data redundancy is improved. Furthermore, because the computing resources are closer to the end-users, the system may be less reliant on network connectivity. Finally, in the IoT world, energy is a critical component that must be thoroughly examined. Because most sensors are battery-powered, lowering energy usage is critical when discussing sensor devices. Fog nodes serving as gateways can help increase overall energy efficiency in these situations. While the sensor is resting, such gateways can handle requests or update procedures. It takes control of the entire program whenever it wakes up. Tasks requiring more intense power processing can also be offloaded to neighboring gateways from energy-constrained devices.

Fog computing is still in its infancy and has a long way to go. Despite all of these accomplishments, fog computing faces new problems that will need more study and attention [4] [5]. In fog computing, service placement is an optimization problem. Based on this optimization challenge, services should be more efficiently placed on computing resources close to the end-user. The Service Placement Problem (SPP) [6] is how this issue is referred to. Transferring application services from one fog device to another at a high or similar level may be required for service placement. While placing the service, various scheduling techniques are to be used. The inefficiency of scheduling algorithms leads to either less used or overly used resources causing the reduction of operational reliability or wastage of resources. Finding an optimized path for assigning n tasks to m processors is the aim of scheduling. Prioritizing the data and sending it to the nearest Fog Nodes (FNs) can perform a number of roles in order to facilitate healthcare models. Healthcare implementation tasks are likely to be produced at normal as well as critical categories which should be sent for processing as soon as possible. The management of resources is applied using the prioritization of data to FNs combined with the scheduling algorithms and sharing of resources. As a result, resource control is an important method for assuring that each healthcare application role is assigned to an effective and comprehensive device in the Fog layer. In particular, Scheduling is the method of ordering the operation of input tasks into predetermined resources, independent of the availability of computing resources or task criticality.

1.2 Problem Definition

In the case of time-critical healthcare systems, the IoT sensors generate enormous data that are responsible for emergency actions. To address their management, Fog Computing is used at the edge of the network. However, proper delegation of tasks is a necessity of healthcare applications. The Fog Devices or Nodes must select the tasks on the basis of emergency and other available information that includes the energy-efficient computing nodes. This research method aims to work on the management of critical and normal tasks and place them on Fog Nodes based on their priority and availability of energy-efficient fog nodes.

1.3 Objectives

The objective of this thesis is:

- To prioritize the critical and normal tasks efficiently, and to place those tasks to the optimal Fog Nodes.
- To compare the Task prioritized fog-based cloud architecture using the proposed Algorithm with baseline algorithms using the evaluation parameters of latency, energy consumption, tasks managed, and percentage utilization of the fog-cloud infrastructures.

1.4 Scope

The scope of this thesis is limited to task prioritization and scheduling in the fog-based cloud architecture to handle the emergency health response. The areas that would be considered within the proposed algorithm are to evaluate the cloud-only and fog-based cloud by utilizing task management processes in order to develop a system that optimizes the services managed, latency, energy consumption, and utilization of the environment. This uses the concept of bringing the network to the edge of the devices in order to efficiently manage time-critical applications i.e., healthcare in this case. The system will not only reduce the latency and energy usage but would also categorize the types of tasks based on their criticality.

1.5 Originality of this Work

This thesis introduced a new hybrid algorithm called PLAEE which considers the criticality of healthcare-related data and also tries to optimize the QoS metrics (delay, response time, and Energy usage) and the percentage of cloud server utilization. It also tries to maximize the number of Critical Tasks Managed at Fog Nodes that are closer to the edge rather than at the Cloud Layer.

1.6 Structure of the Thesis

The thesis is structured into five chapters.

In chapter 1, a brief overview of the problem is given along with the motivations and the importance of doing research in this particular field.

Chapter 2 states the related works regarding this work and presents the research gap in those already done works.

Chapter 3 describes the theories related to understanding the concept behind the work and the research methodology for performing the work to meet the objectives.

Chapter 4 presents the experimental settings and implementation results along with some analytical discussions. Finally, chapter 5 draws conclusions of the thesis and mentions some future work that can be done in this subject area.

CHAPTER 2

LITERATURE REVIEW

In the case of time-critical applications, fog computing shows great efficiency because of fewer network hops for processing. However, the researches in this area are still at the infancy stage.

2.1 Fog-based Cloud System

Mahmoud, M presented the fog-based IoT system and compares it with the traditional Cloud-only policies. This model proves to be more energy efficient than the traditional cloud datacenters. It proposes an energy-aware placement strategy to focus more on the energy consumptions metrics rather than response time and network usage.[7]

2.2 Resource Management and Scheduling

Many scholars have concentrated exclusively on resource management for critical task implementations, such as the concept given by Kumar M. who grouped all concepts relevant to scheduling and provisioning of resources under resource management.[8] However, the most important issue, according to the authors in Gupta H. and Buyya R., [9]is the design of resource management techniques to schedule which analytics framework modules are deployed to each edge system in order to minimize latency and maximize efficiency. They concentrated on the tasks, albeit from a resource standpoint. The words load balancing, prioritization, and resource availability each contribute to improving task scheduling efficiency.

Aldegheishem A. and Bukhsh R. proposed a system in which multiple agents are used to collaborating between various elements of the energy system in smart cities and a Fog based model was developed by utilizing scheduling algorithms in home appliances to minimize the cost of global power and computation, as well as to address the issue of stored energy exceeding potential demand, which undermines the model.[10] Cristescu G. and Dobrescu R. implemented a system to boost edge computer processing capabilities by sharing tasks across a group of emerging edge computing dimensions for adaptive distributed networks.[11] Auluck N. and Singh A.[12] developed a scheduling strategy called Security Aware scheduling in near real-time in the Fog network. In this research paper, the agents generate instances that are unique to each task in all the Fog Nodes in the system.

Mehta M. and Kavitha V. discovered that the priority assignment of tasks before scheduling produces better results compared to various schedulers. The tasks are ranked after the calculation of priority for each task based on the agent's condition and their response. The task with the highest priority is selected in the beginning. The priority assessment takes place on the basis of the criticality and significance of the task.[13] The task assignment and scheduling strategies in the healthcare domain are also studied in various research papers. It is seen from the literature that the research on task prioritization is a new concept in the fog computing domain. Most of the studies are focused on placing tasks to the cloud and fog layer to reduce delay. Aladwani T. proposed a scheduling strategy combined with the prioritization of tasks based on their deadlines along with the MAX-MIN scheduling algorithm. In this research, the author has divided the tasks into most important, medium, and least important tasks based on their deadlines given. The tasks are then sent to VMs based on the resource capacity of each VM. For the evaluation process, execution time is considered in this paper. However, no such calculation on energy usage has been carried out. Moreover, the tasks are sent to random fog nodes even if they are at a longer distance from the edge device.[14] In addition to this, most of the papers utilize the concept of traditional hierarchical architecture which has been now replaced by few papers by introducing the concept of clusters. Khosroabadi F. developed a fog orchestrator or manager that consists of a resource manager, planner, and status monitor. On top of this, the orchestrator interacts with a cloud data center and other fog nodes in charge of locally storing and/or computing data of the IoT. [15] This helps in better scalability of the fog-based cloud architecture.

The prioritization of important tasks can help to ensure efficient scheduling, with priority scheduling being the most effective of other scheduling methods. However, no such healthcare applications have been suggested where the clustered fog-based cloud architecture is utilized to reduce the overall latency and energy consumption also by ensuring that a maximum number of emergency tasks can be handled by the fog layer and the utilization of cloud resources is reduced.

2.3 Research Gap

There are plenty of implementations done in healthcare time-critical applications using fog computing. Few implementations also consider the priority of tasks based on their deadlines, which can be called deadline agnostic approaches. But there is no implementation of any other new methods which consider prioritization based on criticality along with process optimization of latency and energy usage to maximize the number of services handled within the fog layer. Priority-based scheduling is one of the algorithms which is considered best along with the optimization of QoS metrics for time-critical applications. This thesis utilizes the concept of priority, resource management, and process optimization to adjust QoS metrics all at a single time for healthcare-related applications using the new paradigm of clustered fog-based cloud architecture.

CHAPTER 3

METHODOLOGY

This thesis is conducted by utilizing an efficient task management model where the Proposed Prioritized Latency Aware Energy Efficient Algorithm (PLAEE) in fog based system is implemented to assist healthcare emergencies.

3.1 Data Collection

The dataset used in this thesis is provided by the Department of Digital Systems, University of Piraeus. It utilized the IoT consumer-grade devices that collected blood pressure data. This data was collected over a period of 61 days, using iHealth Track, Withings BPM, iHealth View, and iHealth Clear. These records contain 26560 data of both the Systolic as well as Diastolic Blood pressure values, SBP, and DBP respectively. All these IoT devices produced the same data i.e., time as well as the date of the data collection, unique device id, DBP value and unit, SBP value, and unit. Following is the description of the dataset used for algorithm development and experimentations.
Table 3.1 shows the description of the dataset.

Table 3.1: Dataset Description

Attributes	Description
Date	Format: yyyy/mm/dd
Time	Format: hh/mm/ss
DBP Value	Range constraints
DBP Unit	mmHg
SBP Value	Range constraints
SBP Unit	mmHg
device-id	unique identifier

Similarly, UTEM Clinical Dataset is collected from their university site [16] for the validation purpose consisting of 6740 records of Blood pressure data.

Following is the description of the dataset used for the validation.

Table 3.2: Table showing descriptions of datasets for validations

User ID
Systolic
Diastolic
Heart Rate

3.2 Prioritization of Data

In this thesis, the priority of the task would be decided by the task's severity and relevance, i.e., the blood pressure value. At all times, the critical patients must be addressed before the normal patients.

The group of sensors is classified based on all incoming tasks from the sensors on the basis of their criticality. In the case of blood pressure monitoring, it is to be understood is that this value of blood pressure is not enough for clinical diagnosis, but helps to distinguish emergency conditions for immediate further treatment. The priority ranges are modeled as *Critical = High Priority* and *Normal = Low Priority.*

The figure 3.1 shows the categorization of different critical and non-critical levels of Blood Pressure.

Input: Systolic Blood Pressure (mmHg)	Input: Diastolic Blood Pressure (mmHg)	Output: Blood Pressure Category
140	90	High
180	120	Hypertension (Critical)
40	60	Low
90	80	Normal
117	80	Normal
142	90	High
30	10	Hypotension (Critical)

Figure 3.1: Categorization of Tasks

As considered by American Heart Association, the value greater than 180/120 is considered as Hypertensive Emergency and the value less than 30/10 is considered as Critical Hypotension which needs immediate further diagnosis, and the value greater than 200/140 as Hypertensive Urgency is the most Critical value since it leads to Target Organ Damage in case of patients of any age group.[17] However, if only one of the blood pressure values either systolic or diastolic is high, it is called Isolated Systolic Hypertension or Isolated Diastolic Hypertension which is considered less critical than the above emergency cases. [18] This condition is termed critical than the other normal blood pressure ranges so it is scheduled based on critical priority.

The algorithm to prioritize the data is shown here.

Task Prioritization Algorithm

```
if Patients' Data P = Critical then do,
    Push the data into stack S in decreasing order
    while the stack S is not empty do,
        if top(S) is not the highest critical then
            S ←the highest critical data
        else,
            P ←top(S)
            Pop top(S)
        endif
    endwhile
else,
    Push the Normal data into the other stack in decreasing order.
endif
```

3.3 System Architecture

The model consists of three levels: Cloud layer, Fog Layer, and Client Layer. A Cloud Layer is made up of a group of nodes housed in a single Cloud Data Center (CDC). The Fog Layer considers two distinct Fog Zones: the Main Fog Zone (FZ1) and the Neighbor Fog Zone (FZ2). The Client Layer is made up of a variety of devices or endpoints that are geographically spread across the environment. Each

IoT device in this architecture is linked to a fog gateway (FG) as shown in Figure 3.2.

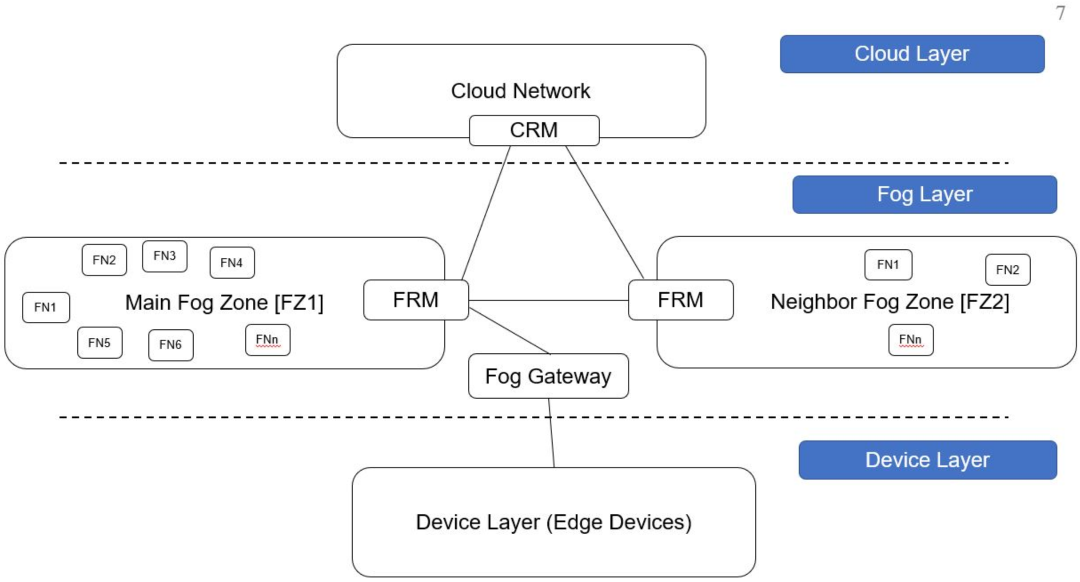

Figure 3.2: System Architecture

IoT devices communicate with their FGs by sending requests. If a Fog Gateway is capable to handling requests on its alone, it will process the request and return the result. Otherwise, it sends the request to the fog manager, who will look for a suitable compute node to host it. There are two fog service providers (FSPs) in the fog environment, each of which contains a group of devices such as access points, routers, Arduinos, Raspberry PIs, microservers, access points, and so on. Compute nodes, also known as fog nodes (FNs), serve as both computing and packet relay devices, whereas forwarding nodes simply serve as packet relay devices. Depending on its computational capability, an FN inside a fog zone (FZ) can host a number of virtual machines or container instances. This entity is responsible for managing each FZ, which is controlled by a fog resource manager (FRM) [19]. This entity is also responsible for hosting IoT services. FRM handles the process of placing a task based on data from edge devices over a set period of time. A data request can be sent to the Main Fog Zone (FZ1), a neighboring Fog Zone (FZ2), or a Cloud Data Center (CDC) in this operation. Many virtualized servers make up the cloud environment. Processing power, memory, and network usage are just a few of the qualities that any cloud server possesses. Cloud Resource Manager (CRM) handles the cloud servers in this scenario. This entity receives IoT requests delivered from the fog zones and places them on cloud servers. Cloud servers are linked to one another via a predetermined network architecture.

3.4 Problem Formulation

The terms requests, tasks, and services are used interchangeably throughout this paper. A typical IoT request is placed on a virtual machine or in a container generally. The placement of services across fog-cloud computing devices is the major emphasis here. Assume that a sensor represents a patient, and $P = \{p1, p2, \ldots.., pn\}$ represents a collection of patients' data supplied by IoT devices to Fog Layer via Fog Gateway (FG). Every task is represented as P_i with their own specifications including CPU, storage and size meausred in MIPS, MB and MI respectively. It is written as P_i^{cpu}, P_i^{mem} and P_i^{size} respectively.

A Fog Zone is represented as graph G = (F, E), in which F denotes the fog nodes, and E defines the group of connections among the computing fog nodes.It is supposed $F^m = \{F_1, F_2, \ldots., F_{|F^m|}\}$ and $F^n = \{F_1, F_2, \ \ldots., \ F_{|F^n|}\}$ be a set of $|F^m|$ and $|F^n|$ fog nodes respectively in FZ1 anf FZ2. For every $j \in \{1, 2, \ldots, |F^a| \cup |F^g|\}$, F_j^{cpu} and F_j^{mem} are utilized to indicate the CPU capacity and storage of the j^{th} Fog Node, respectively. F_j^{Pmax} and F_j^{Pmin} are notations that represent the upper and lower value of power consumption of fog nodes $F_j, \forall_j \in \ \{1, 2, \ldots, |F^a| \cup |F^g|\}$.

For any $(s, d) \in \ E., \delta_{sd}$ is used to indicate a delay between devices within a Fog Zones. It's worth noting that the value of δ_{sd} may be calculated by taking into account network congestion, transmission delay, network usage, and queuing time. Lastly, for the propagation delay between FRM of FZ1 and FZ2, the symbol $\delta_{M,N}$ and for the delay between FRM of FZ1 and CRM, the symbol $\delta_{M,C}$ is utilized.

The set of $|C|$ servers designated inside a CDC as $C = \{C_1, C_2, \ldots.., \ C_{|C|}\}$, where

C_l^{cpu} and C_l^{mem} represent the CPU capacity and storage of C_l, respectively, for every $l \in \{1, 2, \ldots, |C|\}$. C_l^{Pmax} and C_l^{Pmin} are the power consumptions of $C_l, \forall_l \in \{1, 2, \ldots, | C|\}$. Because the major emphasis of this thesis is focused on a fog-based approach, homogeneous cloud servers are used for simplicity. In this model, the FRM chooses the best location for received services, whether it's on the FZ1, FZ2, or CDC. Computing fog nodes run the services after being provided to them for execution and sends back the acknowledgment to the FRM. Consequently, this Fog Resource Manager sends the data to the appropriate edge devices. In the optimization model, the three variables are used that decides the placement approach that may be specified as follows:

$$a_{ij} = \begin{cases} 1 & \text{if service } P_i \text{ is placed within FZ1} \\ 0 & \text{otherwise,} \end{cases} \quad (3.1)$$

$$b_{ik} = \begin{cases} 1 & \text{if service } P_i \text{ is placed within FZ2} \\ 0 & \text{otherwise,} \end{cases} \quad (3.2)$$

$$c_{il} = \begin{cases} 1 & \text{if service } P_i \text{ is placed within CDC} \\ 0 & \text{otherwise,} \end{cases} \quad (3.3)$$

For our proposed Algorithm, we need to calculate the propagation delay as well as energy consumption so that the critical tasks utilize the fog node with less propagation delay from FRM and the normal tasks utilize the fog node with less energy consumption. The main objective of this algorithm is to reduce the task response time for critical data due to which the critical tasks are kept on the fog nodes that are close as possible. This approach calculates the propagation delay which gives the distance from FRM to the Fog node where the task is placed where our major goal the major is to place the task to minimize propagation delay. We can find out the value of propagation delay by calculating the round-trip time (RTT) and dividing it by 2. The equation is written as:

$$RTT(ms) = 0.03 * distance + 5[20] \quad (3.4)$$

This equation is used to calculate the propagation delay among the Fog Nodes from the FRM. In this thesis, the topology of the Fog Zone is generated using the random graph generator during every experiment within the simulation platform.

For the overall time taken to complete the tasks, we calculate the response time of each task and add up to calculate the total response time. The delay between when an edge device submits data and when it gets back the result is specified as the response time of P_i in the fog-based architecture. We'll suppose that the Fog Gateway G_i which is directly linked to the edge devices sends the data P_i to the Fog Resource Manager in Main Fog Zone. FRM accepts all requests and finds suitable compute nodes to place them using the suggested prioritization method which will be discussed in the section of the proposed algorithm. The response time of a task P_i is estimated in the following way, depending on its location:

$$\begin{aligned} L_i = (2\delta_{G_i,M} + 2\delta_{M,F_j} + t_{i,j}).a_{ij} + \ (2\delta_{G_i,M} + \delta_{M,N} + \ 2\delta_{N,F_k} + t_{i,k}).b_{ik} + \\ (2\delta_{G_i,M} + \quad \delta_{M,C} + 2\delta_{C,C_l} + t_{i,l}).c_{il}, \end{aligned} \tag{3.5}$$

where $\delta_{G_i,M}$ = delay between Fog Gateway G_i and the FRM of Main Fog Zone,

δ_{M,F_j} = delay between the FRM and Fog Node F_j of Main Fog Zone where P_i is hosted

δ_{N,F_k} = delay between the FRM and Fog Node F_k of Neighbor Fog Zone,

δ_{C,C_l} = delay between CRM and Cloud Server C_l ,

$t_{i,j}, t_{i,k}$ and $t_{i,l}$ are waiting time (execution time and scheduling delay) for service P_i at Fog Nodes F_j, F_k and Cloud Server C_l, respectively.

The total response time of the model for all the data is:

$$L^{tot} = \sum_{i=1}^{n} L_i \tag{3.6}$$

The power consumption of every processing node must first be evaluated for normal data so that the most energy-efficient node can be selected. The power model adopted from [21] [22] is utilized in which the use of power is explained as a linear model of CPU usage. The model for FNs is given here. This equation is used for calculating the power usage in Cloud servers with this model as well.

$$P_j = \begin{cases} \mathrm{F}_j^{Pmin} + (F_j^{Pmax} - F_j^{Pmin}) \times \mathrm{z}_j, & \text{if } z_j > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.7}$$

where z_j is the CPU usage of FN F_j. The energy usage of Fog Node F_j throughout the processing of given tasks is calculated as follows:

$$E_j = P_j * \sum_{i=1}^{n} a_{ij} \cdot \frac{P_i^{size}}{P_i^{cpu}}, \forall_j \in \{1, 2, \ldots, |F^m| \cup |F^n|\} \tag{3.8}$$

Deriving from the above equations, the total energy usage of the architecture is formulated by the following:

$$E^{tot} = \sum_{j=1}^{|F^m|} E_j + \sum_{k=1}^{|F^n|} E_k + \sum_{l=1}^{|C|} E_l \tag{3.9}$$

The major objective is to solve the fog–cloud system in such a manner that the latency and energy consumption is reduced as a whole by considering the task criticality as well. As a result, the problem is expressed as the function:

$$min[\alpha L^{tot} + (1 - \alpha) E^{tot}] \tag{3.10}$$

where α is a binary value in which it is denoted as 1 for critical tasks and 0 for normal tasks s.t.,

$\sum_{j=1}^{|\mathcal{F}^m|} a_{ij} + \sum_{k=1}^{|\mathcal{F}^n|} b_{ik} + \sum_{l=1}^{|C|} c_{il} = 1, \forall_i \in P,$

which shows that each task is hosted on a single processing node amongst the two zones or cloud data centers considered.

3.5 Proposed Algorithm

The data is divided into two categories: critical and normal, based on the blood pressure range in order to provide good QoS in case of critical data and minimal energy usage in case of Fog Service Providers. Critical data should be placed on fog devices that have a minimal communication latency i.e., propagation delay from FRM to reduce time. To do this, the Fog Nodes are arranged in both the Main and Neighboring Fog Zone in increasing order of propagation delay based on the distance from the Fog Resource Manager. For this purpose, the calculation of propagation delay is done using equation 3.4. This guarantees that δ_{M,F_j} and δ_{N,F_k} are reduced, minimized, resulting in a decreased response time for emergency cases.

In the case of normal tasks, the goal is to help Fog Service Providers save energy. As a result, these tasks are positioned on computing nodes with the lowest possible value in the system's energy usage using equation 3.8. An optimized hybrid algorithm is proposed for this goal.

In the proposed hybrid algorithm, we use the concept of process optimization where the Critical Services P^C tries to reduce the latency and Normal Services P^N tries to reduce the energy usage. The program configures this based on the application (i.e., blood pressure range in this case). The hybrid algorithm is divided into two sections for two categories of data namely Latency Aware Algorithm and Energy Efficient Algorithm to develop PLAEE as a whole. The Latency Aware algorithm is utilized for emergency services, which seeks to reduce latency, and the Energy Efficient Algorithm, which focuses on reducing energy usage. The integration of Proposed Algorithms is implemented as given in the flowchart in figure 3.3.

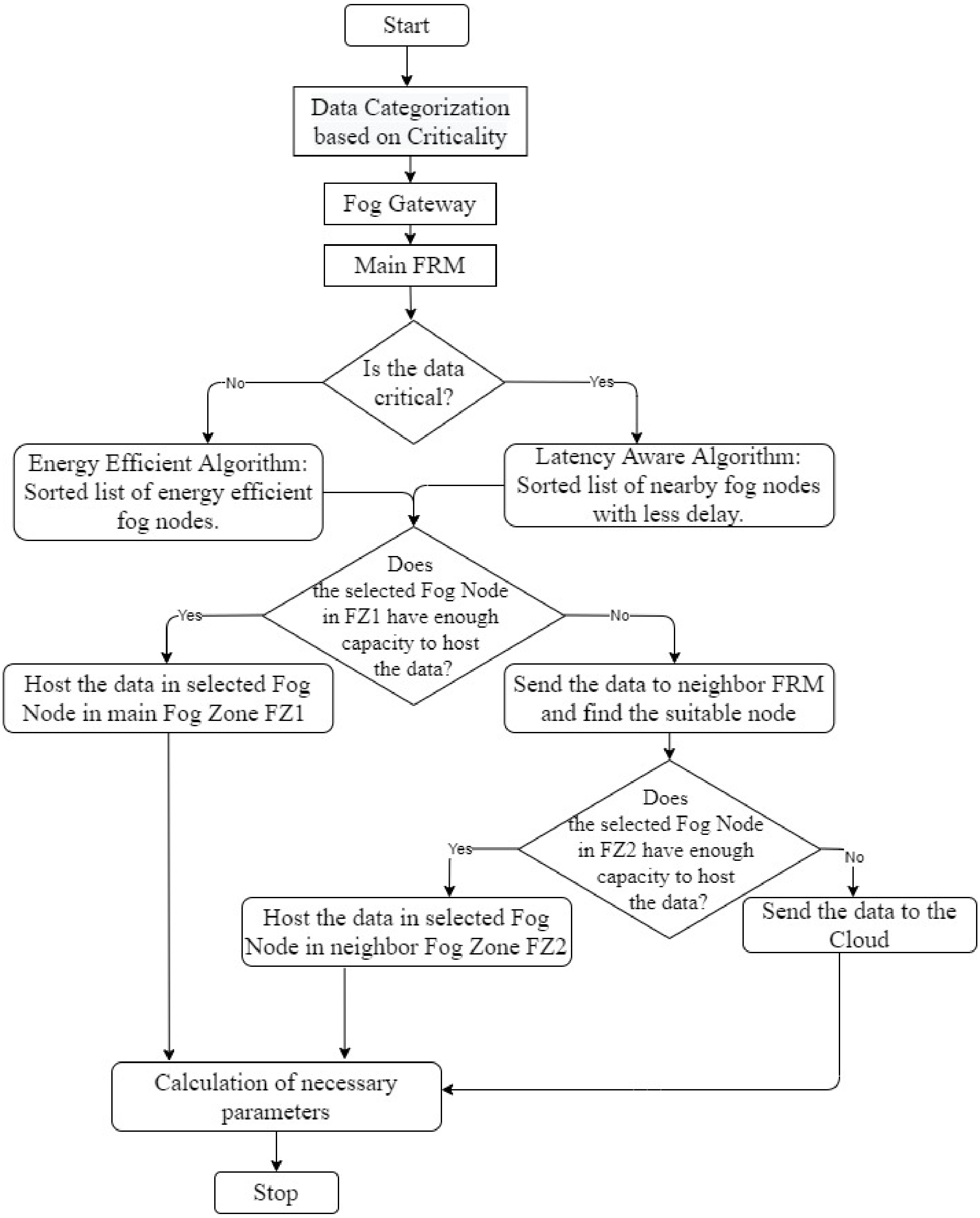

Figure 3.3: Proposed Algorithm Implementation

3.5.1 Latency Aware Algorithm

The Critical Services (C) are fetched based on decreasing order of criticality. The FRM captures the FZ's network traffic before deploying Critical services. The algorithm then loads the list of emergency services that are sorted onto FNs that have a shorter latency with the FRM, i.e., are closer to the edge using the propagation delay. Services will be dispatched to the FZ2 if none of the FNs in the FZ1 have enough resources. As a result, the neighboring FRM is in charge of sending services on its FNs. For the neighboring FRM, the same placement approach is considered. Lastly, when the fog environment's resources are insufficient, the services are transmitted to the CRM. To deploy services on Cloud, CRM randomly places the services on cloud servers.

Latency Aware Algorithm

Input: P^C, Fog Layer (Main, Neighbor) and Cloud Layer
Output: Positioning of Critical Tasks from list C
Fetch the P^C in the decreasing order of criticality placed in the list C.
Sort the Fog Nodes within FZ1 and FZ2 in ascending order of propagation delay from FRM using equation 3.4
for $\forall P^C \subseteq P$ do
    for $\forall F_j \in F^m$ do,
        if $P_i^{cpu} \leq F_j^{cpu}$ and $P_i^{mem} \leq F_j^{;mem}$ then
            place P_i^C on F_j
            update F_j^{cpu} and F_j^{mem}
            break;
        end if
    end for
    if (no enough resources to host P_i^C on $F_j \in F^m$) then
        Ask FRM of FZ2 to host P_i^C within FZ2;
        if $\exists \mathrm{F}_k \in F^n$ then
            send P_i^C to the FRM of FZ2 to host on F_k
            update F_k^{cpu} and F_k^{mem}
        else
            send P_i^C to the CRM to host on C_l
            update C_l^{cpu} and C_l^{mem}
        end if
    end if
end for

3.5.2 Energy Efficient Algorithm

This aims to use as little energy as possible by putting normal services on FNs, which have the smallest impact on overall FZ's energy usage. The normal services are prioritized first in this algorithm that means for non-emergency cases of patients, the energy of FSPs is intended to reduce by using energy-efficient FNs. Normal service with a normal blood pressure range receives high priority in this case. As a result, such a service will have a better probability of running in a fog environment. If there are many FNs for one normal service with equal energy requirement, this system chooses the one with the least network latency from an energy efficiency standpoint. When service cannot be placed on the FZ1 due to a lack of resources, the main FRM requests its neighbor Fog Zone's FRM to host the service, which is the same as the process executed in Critical Services process discussed.

Energy Efficient Algorithm

Input: P^N, Fog Layer (Main, Neighbor) and Cloud Layer
Output: Positioning of Normal Tasks from list N.
Fetch the P^N in the decreasing order of blood pressure value placed in N.
for $\forall P^N \subseteq P$ do
    for $\forall F_j \in F^m$ do,
        if $P_i^{cpu} \leq F_j^{cpu}$ and $P_i^{mem} \leq F_j^{mem}$ then
            Calculate value of E_j
            If $\exists F_{least}$ with the least energy consumption value using equation 3.8 then
                F_{least} = selects the one with minimum propagation delay
                place P_i^N on F_{least}
                update F_{least}^{cpu} and F_{least}^{mem}
                break;
            end if
        end if
    end for
    if (no enough resources to host P_i^N on $F_j \in F^m$) then
        Ask FRM of FZ2 to host P_i^N within FZ2;
        if $\exists \mathrm{F}_k \in F^n$ then
            send P_i^N to the FRM of FZ2 to host on F_k
            update F_k^{cpu} and F_k^{mem}
        else
            send P_i^N to the CRM to host on C_l
            update C_l^{cpu} and C_l^{mem}
        end if
    end if
end for

3.6 Scenario Description

3.6.1 Compared Algorithms

1. Only-Cloud [9] This only-cloud approach is defined as a standard policy of cloud computing in which all the data that are requested are routed through the Centralized Cloud. In this research, it is found to be assuming that, as compared to the fog environment, the cloud environment provides higher computing power to the service.

2. Edge-ward [9] Services are processed in the fog environment in the case of the edge-ward placement approach. When the fog computing nodes' resources are insufficient to host a service, then only it forwards the data to the centralized cloud. The MFC is the sole factor considered in this method, whereas the NFC is ignored.

3. Resource-Aware [Micro-benchmark] [23] This algorithm, integrates three other algorithms known as Module Mapping, Lower bound, and Compare Algorithm which collectively gives the most efficient nodes by sorting the nodes based on their capacity and placing them on the eligible node for every iteration of tasks.

4. SCATTER Algorithm [15] In this paper, a heuristic algorithm is suggested which uses the cluster concept comparable to the fog zones in this thesis. It utilizes the small-scale experiment and the concept of scalability is not exercised.

3.6.2 Experimental Setup

The simulation scenario is created using Java and iFogSim using Eclipse IDE to test the proposed policy's performance. The two FZs are simulated, each with a distinct amount of FNs, along with a datacenter with homogeneous physical machines i.e., Cloud servers. The four distinct FNs and Cloud Servers are analyzed whose specifications are adapted from [9] shown in Table 3.3 and 3.4. For each experiment, an average of 10 independent runs was conducted.

Table 3.3: Fog Nodes' Characteristics

F_j^{cpu},MIPS	F_j^{mem}, MB	F_j^{Pmax},W	F_j^{Pmin},W
300	256	88.77	82.7
1400	2048	103	83.25
1600	1024	87.53	82.44
3000	4096	107.339	83.433

Table 3.4: CDC Characteristics

C_l^{cpu},MIPS	C_l^{mem}, MB	C_l^{Pmax},W	C_l^{Pmin},W
10000	10240	412	333

3.6.3 Evaluation Metrics

This includes the evaluation metrics on which the comparison of given task prioritization and scheduling model in fog-based cloud computing is evaluated against the traditional cloud computing model.

A. QoS metrics:

i. Total Response Time:

It is basically known as the time of response which is calculated as the sum of queuing delay and processing delay for the given request sent by an application instance.

ii. Delay:

It is calculated as the time taken before processing on some specific fog node. The interval from sending the data from the edge node to placing it on some computing node is called delay.

iii. Energy Consumption:

Energy consumption is basically known as the usage of energy which is calculated as the sum of transmission energy and processing energy for the given request sent by an application instance.

B. Tasks Managed:

Tasks managed is the measure to demonstrate the placement of Critical and Normal tasks in Cloud, Main Zone, and Neighbor Zone. The percentage of Critical Services, as well as Normal Services, handled in these 3 system environments, are calculated as:

$$Tasks\ Managed\ (\%) = \frac{Number\ of\ Critical\ Tasks\ in\ (FZ1/FZ2/CDC)}{Total\ Critical\ Tasks} * 100 \tag{3.11}$$

The goal of tasks managed is to demonstrate the importance of utilizing Fog nodes to expand the capabilities of Cloud servers. The number of critical tasks handled, the number of Cloud servers and the number of Fog Nodes may all be used to evaluate performance. In the Fog layer, this measure compares the number of emergency tasks handled in the Main Fog Zone and Neighbor Fog Zone to the total number of emergency services. It measures the effectiveness of the proposed algorithm in terms of the quality of service delivered to consumers.

C. Percentage Utilization of Cloud and Fog Resources:

This measure is calculated to demonstrate the usage of System Environment i.e., Cloud, Main Fog Zone, and Neighbour Fog Zone in this case. The placement of both Critical and Normal Tasks in Cloud, Main, and Neighbor Zone shows how much of the resource is utilized in the Proposed Algorithm compared to the ones that are existing. The percentage of tasks handled in these 3 system environments are calculated as:

$$Utilization\,of\;(FZ1/FZ2/CDC)(\%) = \frac{Number\ of\ Tasks\ in\ FZ1/FZ2/CDC}{Total\ Tasks} * 100 \tag{3.12}$$

The goal of this metric is to observe whether the proposed Algorithm has reduced the usage of Cloud Data Center so that it can be utilized for long-term analysis and storage only which expands the abilities of Cloud Server. Cloud servers are only utilized in case of inefficient fog nodes in the Fog layer. It measures the utilization of system resources and how the use of the proposed algorithm delivers high quality of service by reducing cloud server usage.

3.7 Tools and Resources

- IDE(Eclipse)
- Programming Language: Java
- Simulation Platform: CloudSim, ifogSim

CHAPTER 4

RESULTS, ANALYSIS AND COMPARISON

In this section, the results obtained from the Proposed Algorithm PLAEE are presented along with the detailed analysis of those results. The algorithm was implemented on the dataset discussed in Section 3.1. The assumptions made regarding different parameters values during the simulation are listed in Table 3.3 and Table 3.4.

Section 4.1 presents the experimental results obtained by varying patients' data while keeping the Fog Nodes constant. Section 4.2 presents the experimental results obtained by changing the number of Fog Nodes in the Main Fog Zone only. Section 4.3 presents the experimental results by altering the ratio of Fog Nodes in Neighbor Fog Zone with respect to the Fog Nodes in Main Fog Zone. Section 4.4 presents the experimental results by changing the number of critical data compared to the whole data. All these results are compared with the results obtained of the compared algorithms discussed in Section 3.6.1. Section 4.5 presents the validation results after applying another set of data to the algorithms.

4.1 Effect of varying patients' data

The performance of the Proposed Algorithm is evaluated with various comparison algorithms in this experiment by changing the amount of data from 1000 to 26560 and keeping the nodes within the Main Fog Zone constant at 800. In all aspects, the Proposed Algorithm outperforms the other methods, as shown in Figure 4.1, 4.2, 4.3 and 4.4.

In the instance of Energy Consumption, the proposed method outperforms most of the compared algorithms except for Resource-Aware as shown in Figure 4.1. The following is the major cause for this reduction in energy usage. Firstly, the

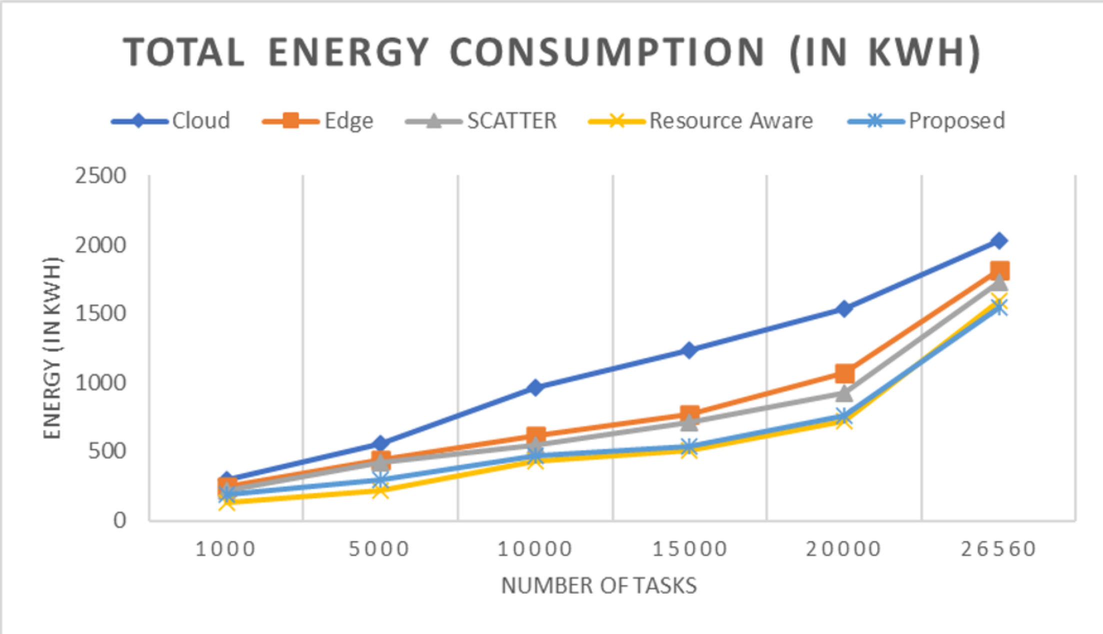

Figure 4.1: Energy Consumption by varying the number of tasks

fog computing system uses more energy-efficient nodes than the centralized cloud system, putting more requests on Fog Nodes will result in increased energy efficiency, which is something the algorithm considers. Secondly, the proposed method chooses comparatively energy-efficient Nodes in Fog Zone for placing data that might result in lower energy usage, which is more visible when the amount of data is low. However, the Resource-Aware algorithm is mainly focused on decreasing the energy consumption rather than the latency due to which it performs slightly better than the proposed algorithm PLAEE when the number of data is low. But when the data is increased, PLAEE performs slightly better than the Resource-Aware algorithm.

In Figure 4.2 and 4.3, it is observed that this Algorithm delivers the best QoS, whereas cloud-only provides the lowest performance. This result is guaranteed because the algorithm seeks to host those data on nearer FNs as much as feasible. In the case of the only-cloud algorithm, all the services are hosted in centralized cloud servers owing to high latency. SCATTER, Resource Aware, and Edge-wards perform better than only-cloud since the emergency service requests that rely on FCs will be handled fast owing to less latency.

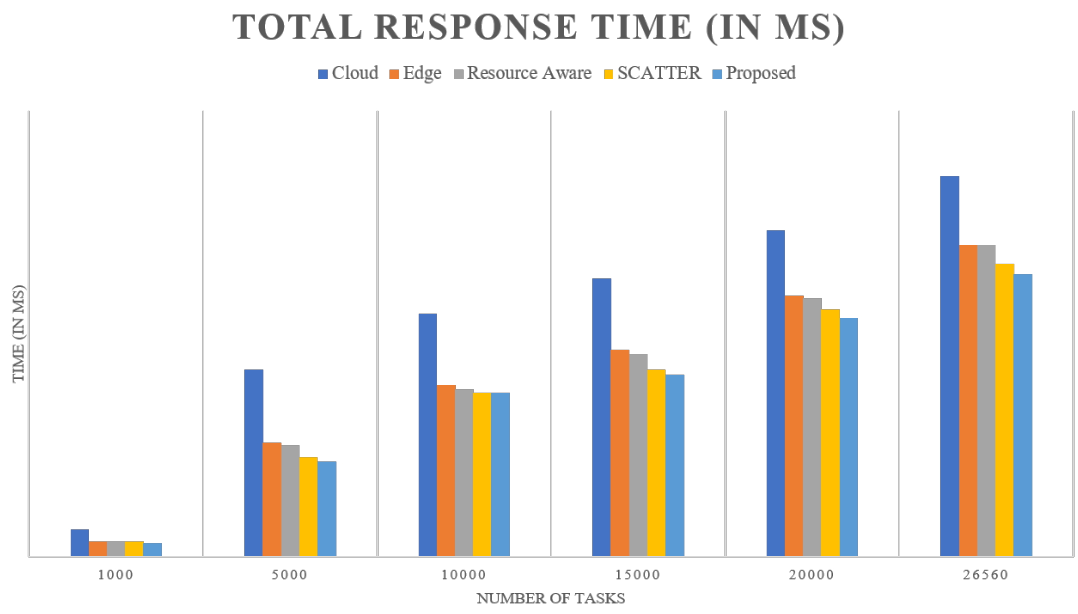

Figure 4.2: Total Response Time by varying the number of tasks

The figure 4.3 shows the delay of various algorithms when varying the nodes in Main Fog Zone. It depicts the delay encountered by IoT queries as the number of data grows.

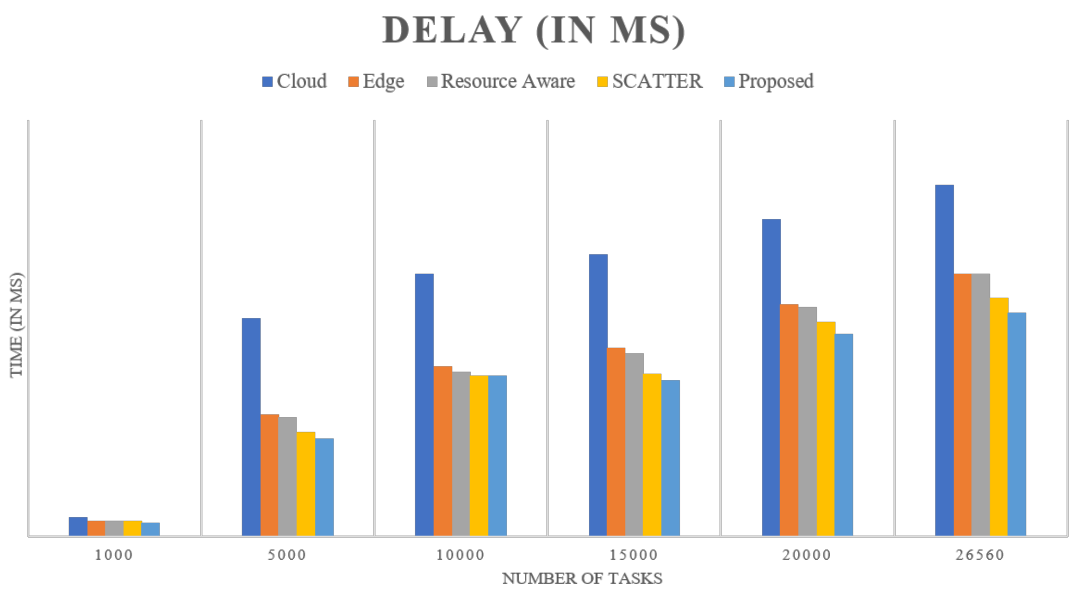

Figure 4.3: Delay by varying the number of tasks

When there are 26560 services, the proposed Algorithm places more than 80 % of emergency requests on FZ1 with the least delay, but this delay is much increased for other algorithms. The % of Critical Tasks Managed by this Algorithm is 91.70, 6.28, and 2.01 % in FZ1, FZ2, and CDC respectively. The Fog Layer handles almost 98% of the critical tasks compared to the existing algorithms which send both the critical and normal tasks to Cloud for execution as shown in Figure 4.4.

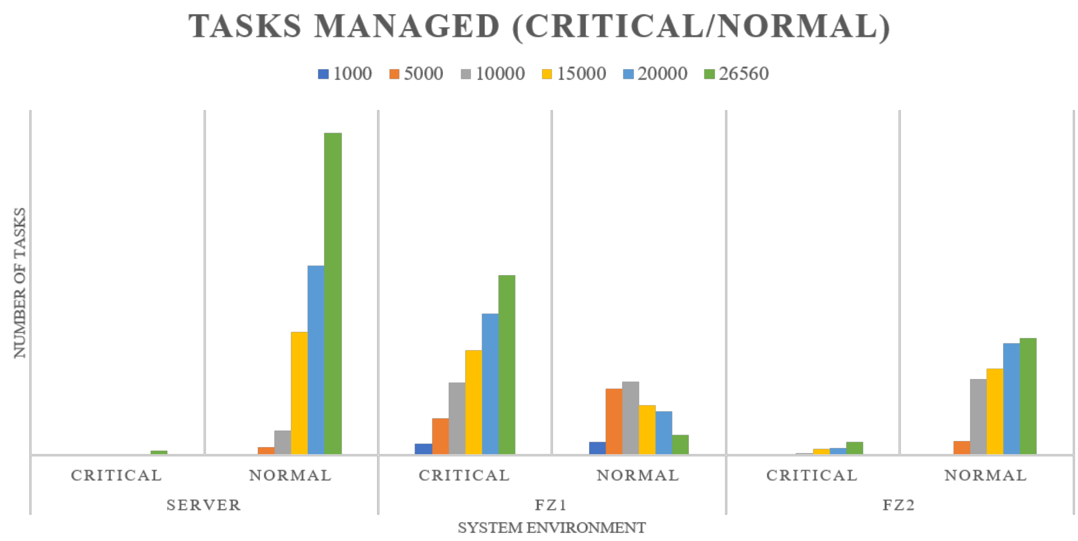

Figure 4.4: Tasks Managed by Proposed Algorithm while varying the number of tasks

In this experiment, for all data, we observe the proposed PLAEE performs 23.85%, 14.96%, 10.84 %, and 2.83% better than Cloud only, Edgewards, SCATTER, and Resource- Aware algorithm in case of Energy Consumption.

In this experiment, for all data, we observe the proposed PLAEE performs 36.40%, 14.82%, 14.70 %, and 6.03% better than Cloud only, Edgewards, Resource- Aware and SCATTER algorithm in case of delay.

4.2 Effect of varying Fog Nodes in Main Fog Zone (FZ1)

Using the whole dataset, the number of nodes inside the Main Fog Zone of Fog Layer is varied from 200 to 1200 in this experiment. The findings show that this proposed algorithm performs slightly better than the existing algorithms. Figure 4.5, in particular, shows how the suggested strategy is improved compared to other strategies in terms of overall energy usage but somehow similar to the Resource-Aware Algorithm. It can be observed from this graph that as the number of Fog Nodes in the Main Fog Zone grows, the overall energy usage is reduced in all algorithms except for the Only-cloud approach. Because a large number of Fog Nodes are considered in Fog Layer where a maximum number of tasks can be handled, which require less power compared to Centralized Cloud.

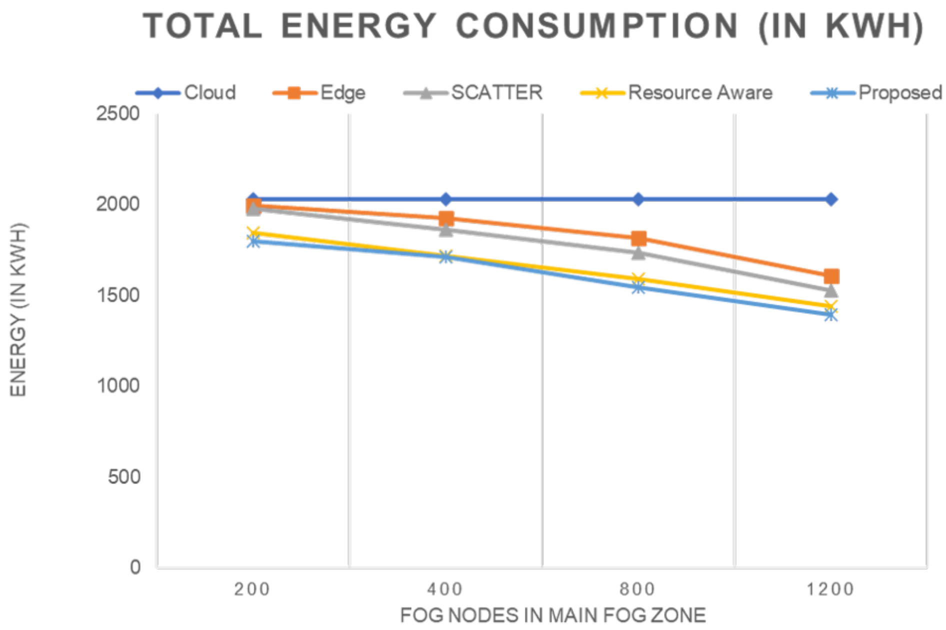

Figure 4.5: Energy Consumption by varying the number of Fog Nodes in Main Fog Zone

Furthermore, the Only-cloud approach is unaffected by this trial. By expanding the number of nodes inside FCs, datasets, particularly emergency services, have a better chance to achieve latency that is faster than it takes the services to reach Cloud. The fog–cloud-based methods begin to deploy all tasks on the fog layer when the nodes increase to 1200. However, this is not suggested since a large number of fog nodes is also not viable to use. So, the number of nodes is fixed to an optimum number of 800 only in this case for further experiments. The figure 4.6 shows the total response time of various algorithms when varying the nodes in Main Fog Zone. It depicts the total response time encountered by IoT queries as the number of Fog Nodes grows, is decreased rapidly compared to the cloud-only approach.

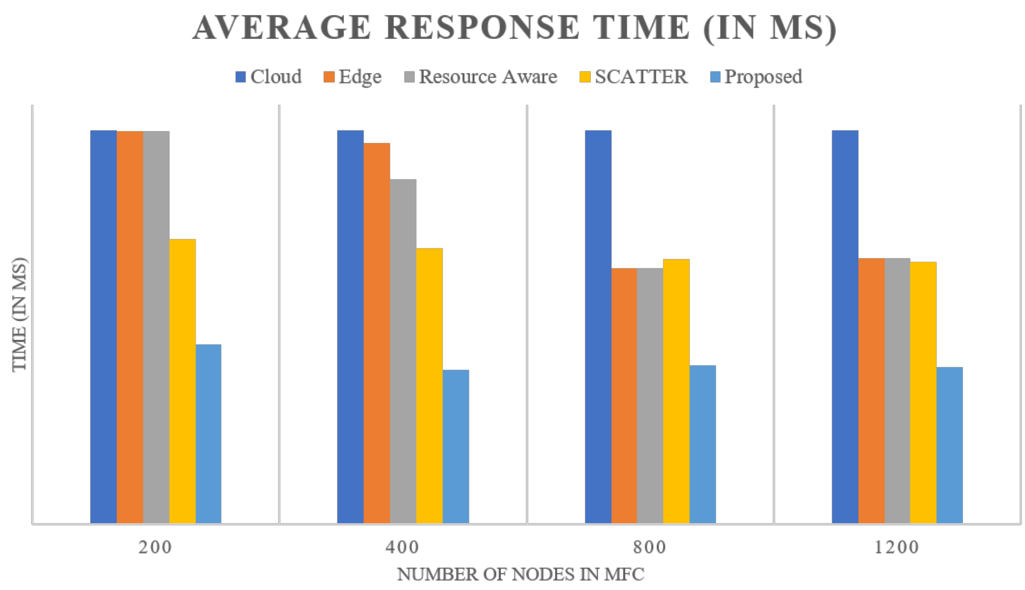

Figure 4.6: Total Response Time by varying the number of Fog Nodes in Main Fog Zone

The figure 4.7 shows the delay of various algorithms when varying the nodes in Main Fog Zone. It depicts the delay encountered by IoT queries as the number of Fog Nodes grows.

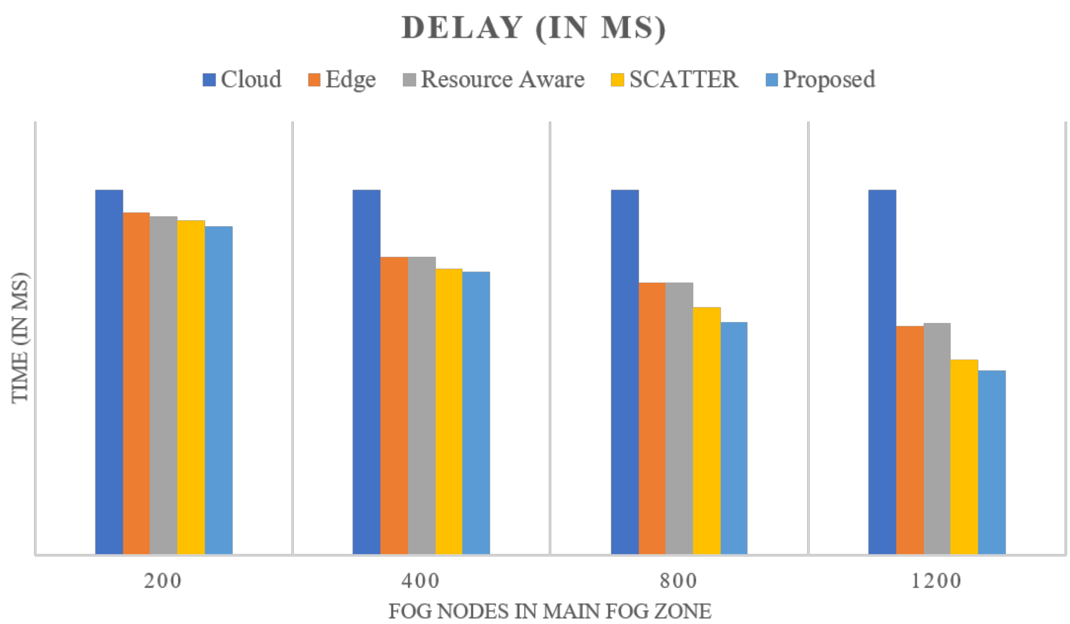

Figure 4.7: Delay by varying the number of Fog Nodes in Main Fog Zone

The number of tasks managed in different layers is another metric to observe how the proposed algorithm behaves compared to the other strategies. Here, in

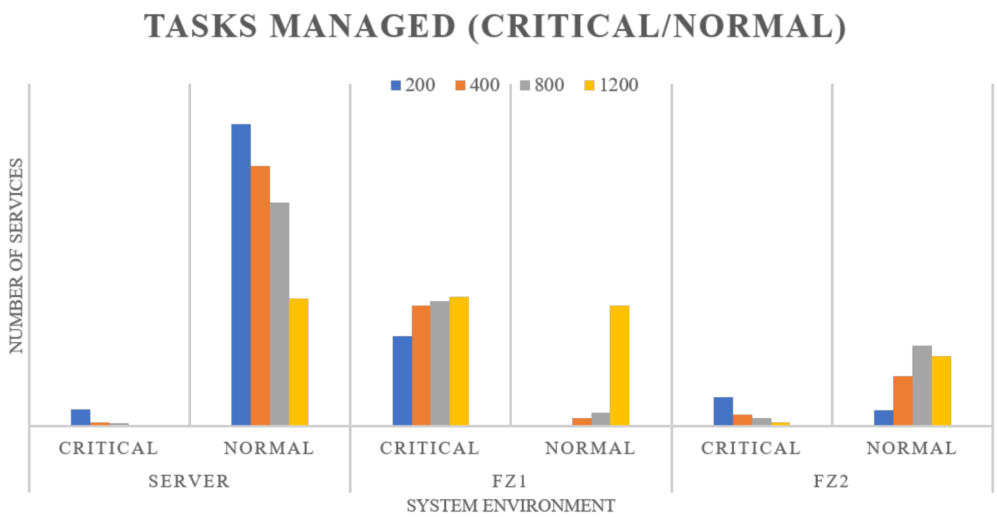

Figure 4.8: Tasks Managed while varying the number of Fog Nodes in Main Fog Zone

Figure 4.8, it is observed that when the number of nodes is 200, the Cloud Server handles almost half the critical services compared to other fog environments. This is because the fog layer does not have enough resources to handle the critical tasks which is why the request is sent to the cloud. Now, when the number of fog nodes is increased, it is seen that maximum of critical services i.e., more than 95 percent of critical services are handled by the fog environment. And very few critical requests are sent to the cloud.

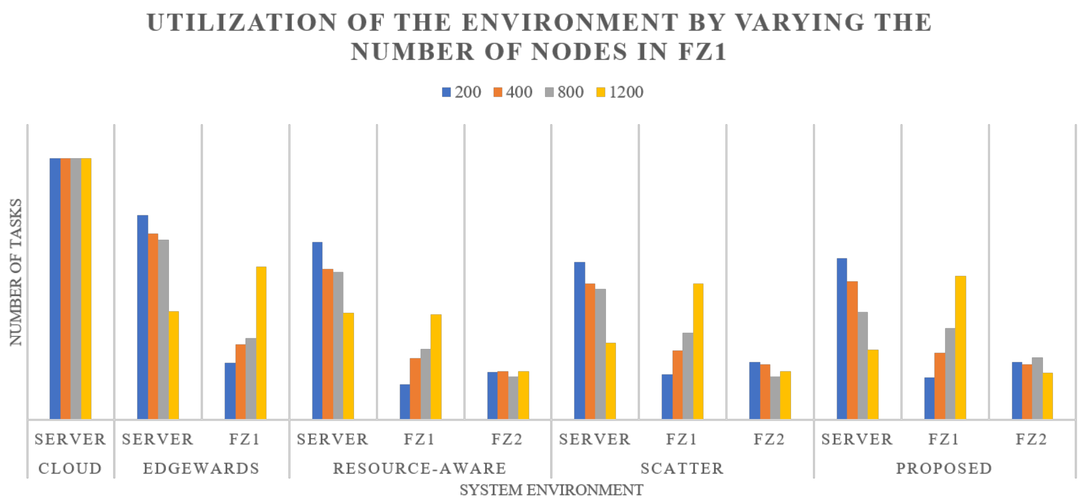

Figure 4.9: Utilization of the System Environment by varying number of Fog Nodes in Main Fog Zone

The utilization of Cloud Servers, Main and Neighbor Fog Zone in Fog layers is seen in the graph in figure 4.9 to see how the environment is being utilized when placing the tasks. It is found that the proposed Algorithm performs somewhat better than Cloud-only and Edgewards in nearly all aspects. However, in the case of Energy Consumed, the proposed algorithm is performing somehow similar to Resource Aware Algorithm, whereas, in the case of Response Time and Delay, it is near to the performance of SCATTER Algorithm.

4.3 Effect of different ratio of Fog Nodes in Neighbor to Main Fog Zone

Within this trial, the effects of altering the ratio of Fog Nodes in Neighbor to Main Fog Zone are observed. Here, the full dataset was taken and set the nodes inside MFC to 800, while varying the ratio of nodes within the NFC from 0.2 up to 1. The outcome of this trial is shown in Figure 4.10, 4.11 and 4.12. Changing the fog nodes in the Neighbor Fog Zone has no effect on the Only-cloud and Edgewards algorithms because they disregard the Neighbor Zone. However, this has a significant impact on the performance of the Proposed Algorithm, Resource Aware Algorithm, and SCATTER Algorithm.

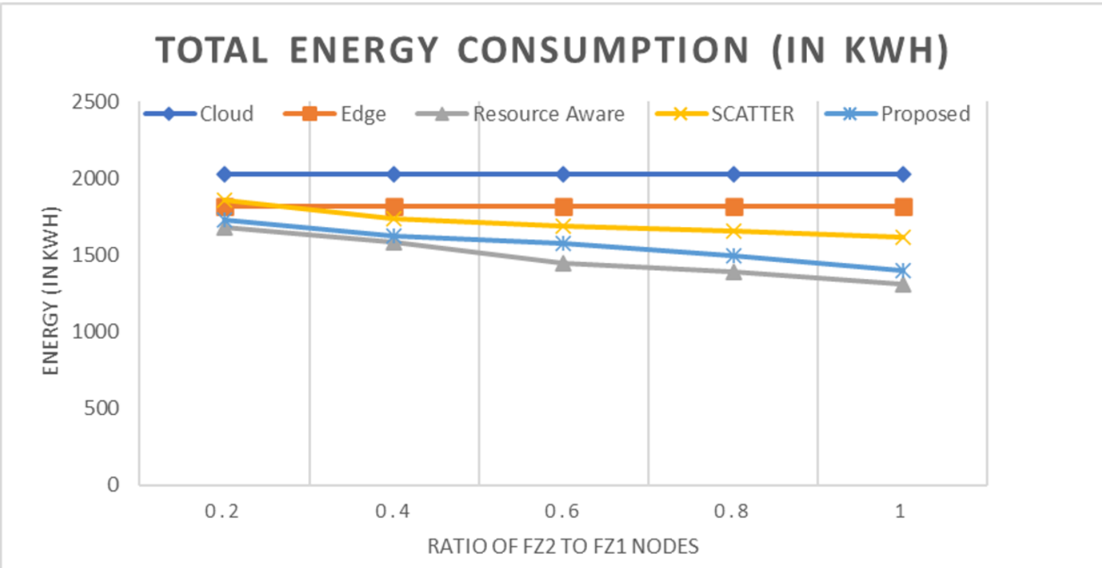

Figure 4.10: Energy Consumption by varying the ratio of FZ2 to FZ1 nodes

As shown in Figure 4.11, when the ratio is adjusted from 0.2 up to 1, the total response time of the Proposed Algorithm is reduced compared to the other baseline algorithms, however, somehow similar to SCATTER algorithm.

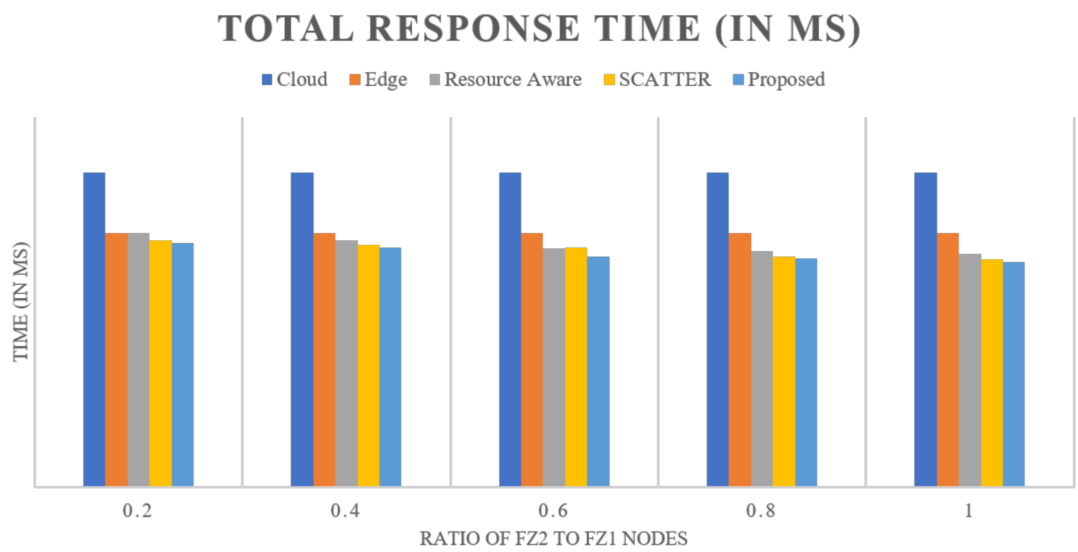

Figure 4.11: Total Response Time by varying the ratio of FZ2 to FZ1 nodes

The figure 4.12 shows the delay of various algorithms when varying the ratio of FZ2 to FZ1 nodes. It depicts the delay encountered by IoT queries as the ratio increases.

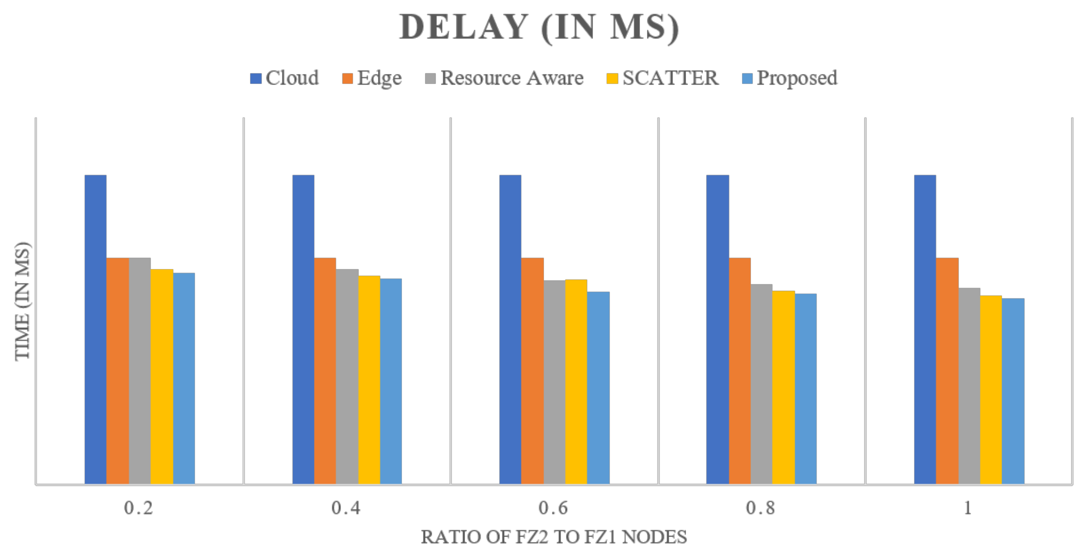

Figure 4.12: Delay by varying the ratio of FZ2 to FZ1 nodes

This experiment has no effect on the environment's percentage usage for only-cloud and Edgewards methods. This is to be anticipated, given that none of these techniques takes Neighbor Fog Zone into account throughout the positioning process.

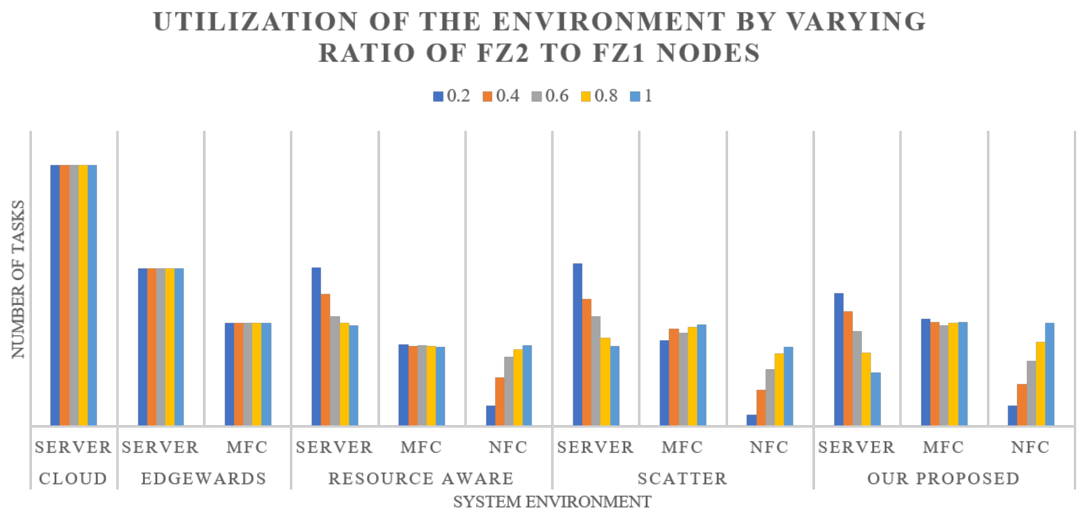

Figure 4.13: Utilization of the System Environment by varying ratio of FZ2 to FZ1 nodes

The Proposed Algorithm, SCATTER Algorithm, and Resource Aware Algorithm on the other hand, by raising the ratio, position a large number of services on FNs available in the Neighbor Zone while minimizing the use of Cloud Servers.

4.4 Effect of altering the ratio of critical tasks to all tasks

The ratio of emergency tasks to all tasks in the dataset was altered in this experiment. The ratio from 0 (all normal tasks) to 1 (all emergency tasks) is varied. The output from this experiment is shown in Figure 4.14, 4.15 and 4.16. In the proposed Algorithm, as shown in Figure 4.14, energy usage increases as the number of emergency services grow. It is because the critical tasks ignore the Energy Efficiency of nodes while placing the important tasks in nearer nodes for less latency. However, this increase in energy in the case of the proposed algorithm is still less than that of the energy produced in the case of Only-Cloud and Edgewards Algorithms.

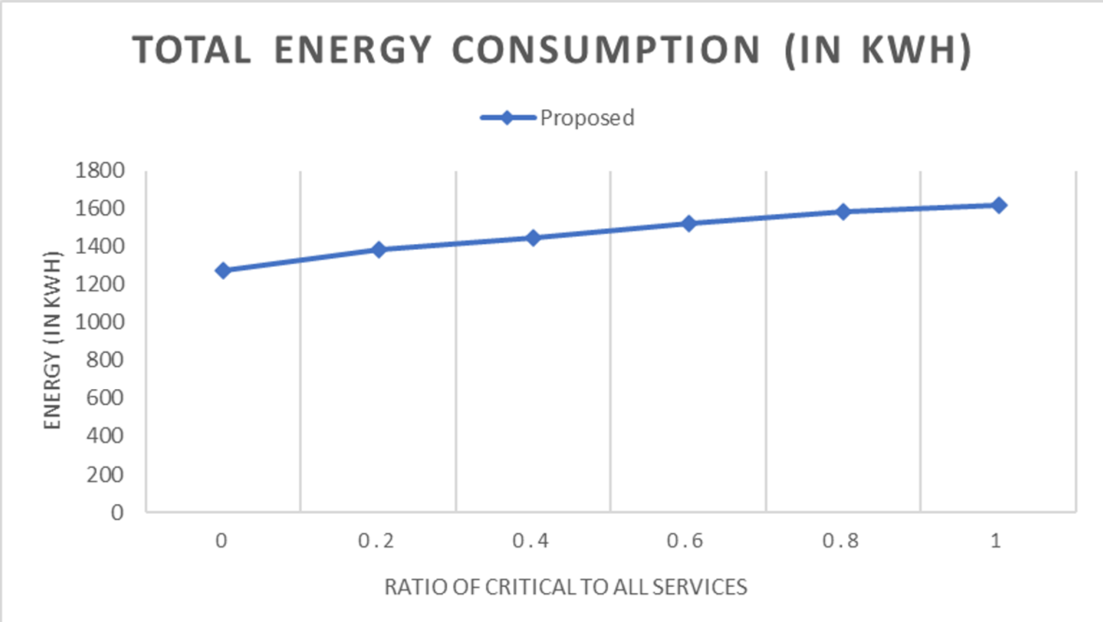

Figure 4.14: Energy Consumption by Proposed Algorithm while varying the Ratio of Critical to All tasks

The other metric, total response time is shown in Figure 4.15 for this particular experiment. It is observed that raising the emergency services to all services ratio from 0 up to 1 decreases this measure significantly. The major cause for this, once again, is the use of FNs for essential (emergency) task processing. As a result, the transmission time is reduced, and the response time of services is consequently reduced. It is to be noted that in the case of this experiment, no other strategies have any effect because they don't prioritize the jobs according to their critical condition and the datasets are dispersed randomly.

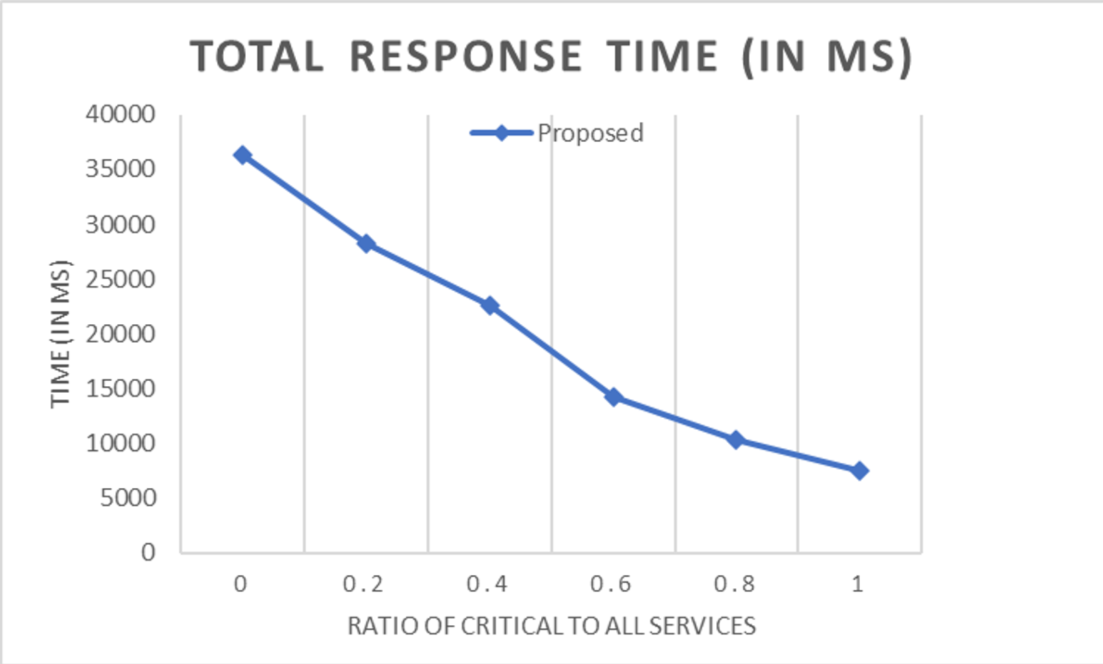

Figure 4.15: Total Response Time of Proposed Algorithm while varying the Ratio of Critical to All tasks

The figure 4.16 shows the delay of the proposed algorithm when varying the ratio of critical tasks to all tasks. It depicts the delay encountered by IoT queries as the ratio increases.

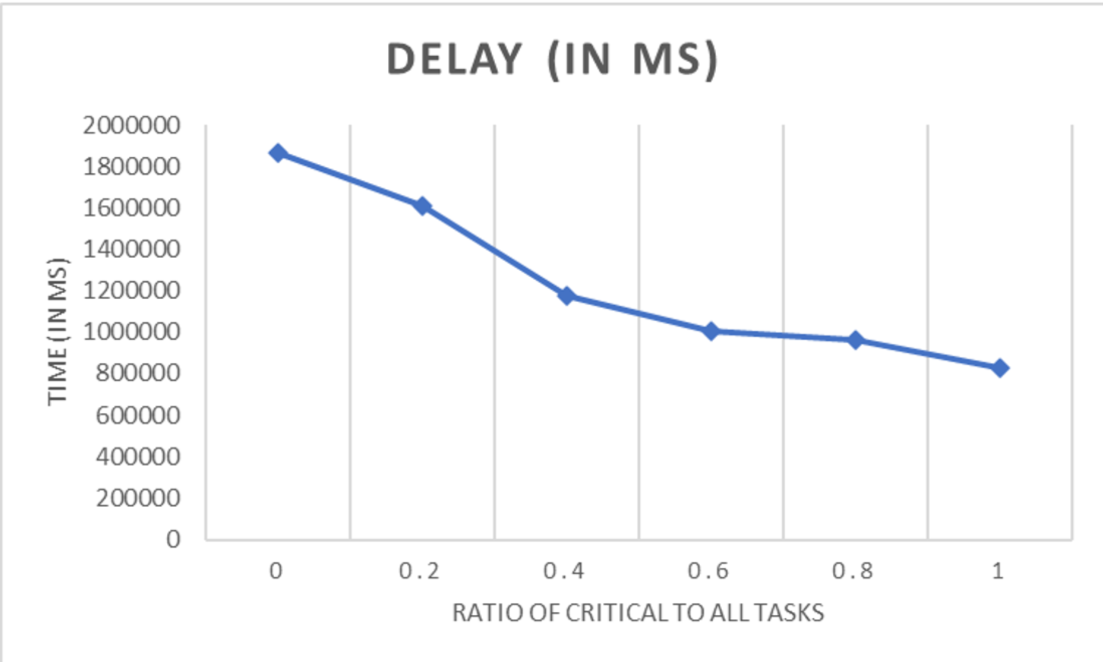

Figure 4.16: Delay by varying the ratio of critical tasks to all tasks

In summary, the proposed Algorithm minimizes overall energy usage in the fog–based cloud system while also providing improved QoS to demanded services, according to the comprehensive research above. This Proposed algorithm accomplishes these benefits by making better use of the infrastructure and prioritizing the data.

4.5 Validation Results

The following table shows the value of the validation results obtained performing the experiments using the algorithms described in the methodology section. The validation is done in different data than that of the experimentation given in the above section.

Table 4.1: Table showing average value of the validation results

Methods	**Total Delay(s)**	**Total Response Time (s)**	**Energy Usage (kWh)**
Only Cloud	1308.482	1698.34	568.72
Edgewards	983.173	1410.624	465.58
Resource Aware	933.436	1390.361	371.96
SCATTER	836.1325	968.304	423.10
PLAEE	776.63	932.172	384.23

Table 4.2: Table showing percentage utilization in validation results

Methods	**Main Fog Zone**	**Neighbor Fog Zone**	**Cloud Server**
Only Cloud	N/A	N/A	100
Edgewards	62.46	N/A	37.53
Resource Aware	33.38	37.09	29.52
SCATTER	63.5	20.24	16.24
PLAEE	59.19	30.26	10.53

As shown in the tables, PLAEE is performing better compared to other algorithms in the validation data as well.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

A new fog-based algorithm called Prioritized Latency Aware Energy Efficient Algorithm (PLAEE) is developed using the concept of existing popular algorithms. For that, the Fog-based Cloud platform is chosen for the implementation. As already mentioned, we have divided the data into Critical and Normal Data where critical data is given more priority. As shown from the above results, Critical data achieves high QoS by decreasing latency and Normal Data achieves less energy usage for FSPs.

Along with this, more data is handled by Fog Layer shown by the % of Tasks Managed, whereas the utilization of Cloud is reduced heavily making more utilization of fog resources near to the edge network. However, when we increase the critical tasks, energy consumption is to be maximum than other algorithms in the case of the proposed algorithm. This is due to the fact that the Proposed Algorithm is designed to work better in case of latency only when used for Critical Data.

As observed from the above experiments, in some of the instances of Energy Consumption, Resource- Aware seems to be performing the same or better than PLAEE mainly when the number of Fog Nodes is less. This is due to the fact that our algorithm PLAEE considers the tradeoff between Energy Consumption and Delay. This is justified because our objective is to optimize the QoS metrics for the data according to their criticality.

From the above experiments, considering 800 nodes in FZ1, 400 nodes in FZ2 and the whole dataset, the % of Critical Tasks Managed by PLAEE is 91.70, 6.28, and 2.01 % in FZ1, FZ2, and CDC respectively. The Fog Layer handles almost 98% of the critical tasks compared to the existing algorithms which send both the critical and normal tasks to Cloud for execution.

We observed that the proposed PLAEE performs 23.85%, 14.96%, 10.84%, and 2.83% better than Cloud only, Edgewards, SCATTER, and Resource- Aware algorithm respectively in the case of Energy Consumption. Similarly, we observe the proposed PLAEE performs 36.40%, 14.82%, 14.70%, and 6.03% better than Cloud only, Edgewards, Resource- Aware, and SCATTER algorithm respectively in case of delay.

The validation is done in a different set of data than that of the experimentation. PLAEE is performing better compared to other algorithms in the validation data as well.

5.2 Future Works

Fog Computing research is considered to be a new paradigm which is why this work can be further extended to generate more valuable results. A lot of effort in this thesis work has been put into resource and task management of the fog-based model. This thesis utilizes the static topology of devices which can be made dynamic for future enhancement by considering SDN for the mobility of devices to develop more real-time applications. The other area of extension that can be performed is integrating the Machine Learning algorithms in order to correctly classify the critical healthcare cases. This can be more effective in the case of real-time health applications.

REFERENCES

[1] Ketanpreet Kaur and Monika Sachdeva. Fog computing in iot: An overview of new opportunities. *Proceedings of ICETIT 2019*, pages 59–68, 2020.

[2] Thang Nguyen, Khiem Doan, Giang Nguyen, and Binh Minh Nguyen. Modeling multi-constrained fog-cloud environment for task scheduling problem. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, pages 1–10. IEEE, 2020.

[3] Morteza Dadashi Gavaber and Amir Rajabzadeh. Badep: Bandwidth and delay efficient application placement in fog-based iot systems. *Transactions on Emerging Telecommunications Technologies*, page e4136, 2020.

[4] José Santos, Tim Wauters, Bruno Volckaert, and Filip De Turck. Fog computing: Enabling the management and orchestration of smart city applications in 5g networks. *Entropy*, 20(1):4, 2018.

[5] OpenFog Consortium Architecture Working Group et al. Openfog reference architecture for fog computing. *OPFRA001*, 20817:162, 2017.

[6] Chenhua Shi, Zhiyuan Ren, Kun Yang, Chen Chen, Hailin Zhang, Yao Xiao, and Xiangwang Hou. Ultra-low latency cloud-fog computing for industrial internet of things. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2018.

[7] Mukhtar ME Mahmoud, Joel JPC Rodrigues, Kashif Saleem, Jalal Al-Muhtadi, Neeraj Kumar, and Valery Korotaev. Towards energy-aware fog-enabled cloud of things for healthcare. *Computers & Electrical Engineering*, 67:58–69, 2018.

[8] Mohit Kumar, Kalka Dubey, and Rakesh Pandey. Evolution of emerging computing paradigm cloud to fog: Applications, limitations and research challenges. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 257–261. IEEE, 2021.

[9] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K Ghosh, and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

[10] Abdulaziz Aldegheishem, Rasool Bukhsh, Nabil Alrajeh, and Nadeem Javaid. Faavpp: Fog as a virtual power plant service for community energy management. *Future Generation Computer Systems*, 105:675–683, 2020.

[11] Giorgiana Cristescu, Radu Dobrescu, Oana Chenaru, and Gheorghe Florea. Dew: A new edge computing component for distributed dynamic networks. In *2019 22nd International Conference on Control Systems and Computer Science (CSCS)*, pages 547–551. IEEE, 2019.

[12] Anil Singh, Nitin Auluck, Omer Rana, Andrew Jones, and Surya Nepal. Rt-sane: Real time security aware scheduling on the network edge. In *Proceedings of the10th International Conference on Utility and Cloud Computing*, pages 131–140, 2017.

[13] Malhar Mehta, Veeraruna Kavitha, and Nandyala Hemachandra. Price of fairness for opportunistic and priority schedulers. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 1140–1148. IEEE, 2015.

[14] Tahani Aladwani. Scheduling iot healthcare tasks in fog computing based on their importance. *Procedia Computer Science*, 163:560–569, 2019.

[15] Fariba Khosroabadi, Faranak Fotouhi-Ghazvini, and Hossein Fotouhi. Scatter: Service placement in real-time fog-assisted iot networks. *Journal of Sensor and Actuator Networks*, 10(2):26, 2021.

[16] Ammar Awad Mutlag, Mohd Khanapi Abd Ghani, Mazin Abed Mohammed, Mashael S Maashi, Othman Mohd, Salama A Mostafa, Karrar Hameed Abdulkareem, Gonçalo Marques, and Isabel de la Torre Díez. Mafc: Multi-agent fog computing model for healthcare critical tasks management. *Sensors*, 20(7):1853, 2020.

[17] Dimitris P Papadopoulos, Iordanis Mourouzis, Costas Thomopoulos, Thomas Makris, and Vasilios Papademetriou. Hypertension crisis. *Blood pressure*, 19(6):328–336, 2010.

[18] Desilu Mahari Desta, Dawit Zewdu Wondafrash, Afewerki Gebremeskel Tsadik, Gebremicheal Gebreslassie Kasahun, Segen Tassew, Teklu Gebrehiwot, and Solomon Weldegebreal Asgedom. Prevalence of hypertensive emergency and associated factors among hospitalized patients with hypertensive crisis: a retrospective cross-sectional study. *Integrated Blood Pressure Control*, 13:95, 2020.

[19] Shreshth Tuli, Redowan Mahmud, Shikhar Tuli, and Rajkumar Buyya. Fogbus: A blockchain-based lightweight framework for edge and fog computing. *Journal of Systems and Software*, 154:22–36, 2019.

[20] Asfandyar Qureshi. *Power-demand routing in massive geo-distributed systems*. PhD thesis, Massachusetts Institute of Technology, 2010.

[21] Dara Kusic, Jeffrey O Kephart, James E Hanson, Nagarajan Kandasamy, and Guofei Jiang. Power and performance management of virtualized computing environments via lookahead control. *Cluster computing*, 12(1):1–15, 2009.

[22] Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, and Charles Lefurgy. Optimal power allocation in server farms. *ACM SIGMETRICS Performance Evaluation Review*, 37(1):157–168, 2009.

[23] Mohit Taneja and Alan Davy. Resource aware placement of iot application modules in fog-cloud computing paradigm. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1222–1228. IEEE, 2017.

APPENDIX A

Similarity Index (Turnitin)

MSCKE_Final_Thesis_Report_for_SI/075Mscke_014_Prakriti_...

ORIGINALITY REPORT

13%

SIMILARITY INDEX

PRIMARY SOURCES

1	flipkarma.com Internet	308 words — 3%
2	www.mdpi.com Internet	115 words — 1%
3	citeseerx.ist.psu.edu Internet	75 words — 1%