Enhancing Energy Efficiency and Fast Decision-Making for Medical Sensors in Healthcare Systems: An Overview and Novel Proposal

Ziyad Almudayni , Ben Soh * , Alice Li

Posted Date: 4 August 2023

doi: 10.20944/preprints202308.0310.v1

Keywords: Internet of Health Things; edge computing; fog and cloud computing; mist computing; energy efficiency; fuzzy logic; computing capacity; load balancing

Article

Enhancing Energy Efficiency and Fast Decision-Making for Medical Sensors in Healthcare Systems: An Overview and Novel Proposal

Ziyad Almudayni [1,2] and Ben Soh* [1] and Alice S. Li [3]

[1] Department of Computer Science and Information Technology, School of Computing, Engineering and Mathematical Sciences, La Trobe University, Bundoora, VIC 3086, Australia
[2] Department of Computer Engineering, College of Computer Science and Computer Engineering, Hail University, Saudi Arabia
[3] La Trobe Business School, La Trobe University, Bundoora, VIC 3086, Australia
* Correspondence: b.soh@latrobe.edu.au

Abstract: In the realm of the Internet of Things (IoT), a network of sensors and actuators collaborates to fulfill specific tasks. As the demand for IoT networks continues to rise, it becomes crucial to ensure the stability of this technology and adapt it for further expansion. Through an analysis of related works including Feedback-based Optimized Fuzzy Scheduling Approach (FOFSA) algorithm, Adaptive Task Allocation Technique (ATAT) and Osmosis Load Balancing algorithm (OLB), we identify their limitations in achieving optimal energy efficiency and fast decision-making. To address these limitations, this research introduces a novel approach to enhance the processing time and energy efficiency of IoT networks. The proposed approach achieves this by efficiently allocating IoT data resources in the Mist layer during the early stages. We apply the approach to our proposed system known as the Mist-based Fuzzy Healthcare System (MFHS) that demonstrates promising potential to overcome the existing challenges and pave the way for efficient Industrial Internet of Healthcare Things (IIoHT) of the future.

Keywords: Internet of Health Things; edge computing; fog and cloud computing; mist computing; energy efficiency; fuzzy logic; computing capacity; load balancing

1. Introduction

The Internet of Things (IoT) facilitates seamless communication between sensors and actors within a network, serving specific tasks across various work environments, including healthcare systems. The fundamental objective of IoT networks is to streamline workflows and enhance overall convenience. As a result, the demand for IoT networks is projected to experience significant growth in the forthcoming years, with estimates reaching 55.7 billion networks by 2025 [1]. Consequently, ensuring the stability and adaptability of this technology becomes crucial to meet the evolving demands. This paper introduces a novel solution called the Mist-based Fuzzy Healthcare System (MFHS), which aims to improve processing time and energy efficiency in IoT networks by strategically allocating IoT data resources in the early stages within the Mist layer. The paper comprises six sections, where the first section critically examines and analyses recent studies relevant to MFHS. The second section provides a comprehensive review of previous studies that leveraged fuzzy logic systems to address IoT network challenges. Section 3 offers an overview of the motivation and contributions of MFHS, while Section 4 presents the detailed methodology employed in MFHS. Section 5 showcases the results obtained from the implementation of MFHS, including a comparative analysis with existing approaches. Finally, the concluding section summarizes the key findings and highlights the implications of MFHS in advancing IoT network efficiency within healthcare systems.

2. Related work

The related work in this section consists of two parts:

 doi:10.20944/preprints202308.0310.v1

A) The load balancing of IoT networks among the four layers (Edge, Mist, Fog, and Cloud).
B) The effective use of fuzzy logic systems in networking.

A) Load balancing among the four IoT network layers

This section provides an overview of recent studies conducted between 2019 and 2022 that focused on workload balancing and task management within IoT systems across the Edge, Mist, Fog, and Cloud layers. The objective of these studies was to leverage the capabilities of all four layers and optimize their utilization within the system. However, this section also highlights the existing gaps in these studies and proposes potential solutions to address them.

In their work, Barik et al. (2021) [2] introduced a new algorithm known as the RAO-1 ALGORITHM, which aimed to minimize energy consumption within the Mist computing environment of IoT networks. The proposed approach consisted of two stages: reducing the number of unutilized microcomputers and effectively allocating tasks to suitable participating microcomputers. By implementing the algorithm using Python 3.7, the results demonstrated reduced power consumption and superior performance compared to the ECTC and MaxMaxUtil algorithms. However, from our perspective, simply reducing the number of unused microcomputers within the Mist layer may not provide an effective solution. It would be more beneficial for the authors to focus on effectively utilizing all available microcomputers and achieving workload balance to alleviate the burden on the Cloud and Fog computing environments. Furthermore, considering factors beyond just task completion time, such as data size, can contribute to making better resource allocation decisions.

In traditional IoT networks using the MQTT communication protocol, the MQTT brokers are distributed in the network and concentrated to Cloud nodes. Thus, the traditional MQTT protocol is inappropriate for time-sensitive applications because it does not support real-time communication between IoT devices. Therefore, Hmissi et al. 2021[3] proposed a new approach called MQTT-MBD to meet task deadlines and improve network efficiency by distributing MQTT brokers over Mist nodes. To ensure that the MQTT-MBD improves communication delay, the authors identify four steps: Registration, Selection, assignment and connection to select an appropriate broker. The broker selection is based on three criteria: enough energy, meeting the real-time constraint and CPU utilization. Finally, a task will be sent to the Mist broker only if the Cloud broker, the Fog broker, and Edge broker cannot answer the deadline. The simulation results showed that the MQTT-MBD achieved better results than Extended-EMMA and DM-MQTT in energy efficiency and delay. The only notable drawback that can be seen in this proposed approach is that the process of selecting an MQTT Mist broker to control tasks for the systems comes at a very late stage. This is because IoT tasks must travel to all layers in a decreasing order starting from the Cloud to the Mist layer and ask each computing layer if it can meet the deadline requirements for the task until it reaches the Mist layer. This process increases the execution time and consumes power; however, these circumstances can be avoided if selecting the Mist broker comes earlier.

Resource allocation and managing tasks among the Cloud, Fog and Mist computing in the IoT networks is a challenge. Therefore, Refaat 2020 [4] proposed a new model called Multi-Level IoT Tasks Scheduling (MLITS) to manage and allocate resources for IoT tasks and distribute these tasks over the Cloud, Fog and Mist computing environments. The criteria to distribute tasks among these computing environments is based on the task's deadline and the urgency to execute it. Using CloudSim 3.0.2, the proposed approach succeeded in achieving better results than Min-Min, CBS, EFDF algorithms in terms of the time to complete a task, less waiting time and more significant throughput . However, the MLITS has some drawbacks; the criteria to distribute tasks is limited, especially when it is known that the Fog, Cloud and Mist have different source capabilities. The second drawback is that all IoT tasks are first sent to the Mist, then if the Mist nodes are not capable of processing a task, it will be offloaded to the Fog nodes and so on until it reaches the Cloud. It is more optimal for a task to directly find its resources, as the offloading process incurs energy costs.

Due to the limited resources of Mist computing nodes, they are incapable of handling all IoT tasks, necessitating the offloading of these tasks to the Fog or Cloud for further processing. Hence, it

 doi:10.20944/preprints202308.0310.v1

is crucial to be aware of the node capabilities and the task requirements in all layers. Thus , Drosdov et al. 2019 [5] proposed a new model to extend the offloading process to Mist computing. The authors implemented two services to offload tasks from one to another accurately. First, the node summarizer (NS) collects information about the hardware capabilities. Second, the service summarizer (SS) gathers information about the service requirements. These two services help in selecting the optimum Mist node when offloading tasks. The proposed model was applied in three different experiments; the results showed that it successfully added 20 ms processing time to the system. In this proposed approach, the authors only focused on offloading tasks from one Mist node to another Mist node in the system and ignored all other layers. Furthermore, the task offloading criteria solely focus on analysing the Mist nodes, without taking into consideration any specific concerns related to the tasks themselves.

Hensh et al. 2021[6] studied and analysed the impact of extending the computing process through the Cloud, Edge and Mist layers to balance the workload among them and engage all layers to be a part of the computing process. The authors divided the priority of tasks into four types to select a resource. The classification is based on the time needed to complete a task and the response time's importance level. Using Visual Studio code, the authors completed four different experiments to compute tasks: Edge only, Cloud only, Fixed Edge-Cloud, collaborative Edge-Cloud and Mist-Edge-Cloud. The simulation results proved that the experiment of Mist-Edge-Cloud achieved better delay time than the other experiments. The authors did not introduce a new algorithm; instead, they conducted an experiment to highlight the significance of utilizing all four computing layers in data computation.

Shahid et al. 2021[7] implemented a new framework called IoTNetWar for military organizations to achieve better time and resource scenarios when using IoT service monitor troops. The authors take advantage of Machine Learning; specifically, they employed the delay-based K nearest neighbour algorithm to distribute IoT tasks among the Mist, Fog, Cloud, and application layers. By employing this ML algorithm, a task embarks on a journey from Mist computing to the Cloud, seeking an appropriate resource. EdgeCloudSim was used to validate the proposed framework on three different scenarios: Fog-computing (Load-balanced), Mist-computing (Load-balanced) and Mist computing delay-based K nearest neighbours (KNN). The results proved that the proposed framework Mist computing (KNN) achieved better results than Fog-computing (LB) and Mist-computing (LB) in terms of Service Time , Processing Time, Latency and CPU Utilization. It is noticeable from the framework that there is no resource allocation in the IoTNetWar algorithm as tasks search for a suitable resource to fit task requirements starting from the first layer (Mist layer) to the last layer (Cloud layer). The process of searching for resources for IoT tasks among all layers is very time-consuming, and this can be avoided if the resources are selected at an early stage without the need for searching.

Resource management and allocating resources for IoT tasks accurately is one of the keys to achieving better latency, bandwidth and energy efficiency for IoT networks. Therefore, Hosen et al. 2022 [8] proposed a new algorithm called MSRM-IoT to allocate resources for IoT tasks. All IoT tasks are first sent to Edge Broker (EB) in this algorithm. Inside this EB, there is an Application receiver and classifier (AC), and its function is to distribute these tasks among Mist computing, Fog computing and Cloud computing based on the input size, the number of tasks and the number of MIPS/workload of a task. In addition to the AC, there is a Mist Resource Manager in the EB to assist tasks to select the best Mist node. Moreover, there is a Fog Broker (FB) in the Fog computing layer, and its function is similar to EB. MATLAB19a was used to validate the proposed algorithm and compare it with three evolutionary algorithms: Router, FCFS and Short Job First. The results showed that the MSRM-IoT outperforms the three algorithms across all measures [8].The parameters mentioned in MSRM-IoT are mainly based on the computing size only to determine a resource for the IoT tasks, which is not efficient from our point of view. In allocating resources for IoT tasks, it is essential to determine tasks' importance and their timing requirements in real-time processing.

Ejaz et al. 2020 [9] analysed the difference between computing IoT tasks in the traditional Cloud-IoT model, Edge-Cloud-IoT model and local Edge-Cloud-IoT model. iFogSim was used to evaluate

 doi:10.20944/preprints202308.0310.v1

the three different scenarios. The simulation results showed that processing IoT tasks locally is the best with respect to connectivity and energy consumption compared with the first and second models. However, it is important to know that different IoT tasks might vary in different scenarios, and processing tasks locally is not always practical as local nodes have limited capabilities. Thus, Cloud and Fog assistance is always required to serve in all scenarios.

Tripathy et al. (2022) [10] introduced the Secure-M2FBalancer model to enhance the security and workload balancing of IoT networks through the integration of supervised learning and a genetic algorithm. The proposed model was implemented within a healthcare management system and comprised four layers: the IoT layer, Mist layer, Fog layer, and Cloud layer. In this model, the IoT layer forwards data to the Mist layer, which is responsible for resource allocation. If the Mist layer cannot process the data, it is offloaded to the Fog layer. Within the Fog layer, a technique is employed to determine the status of servers, distinguishing between overloaded and underloaded servers to enable accurate resource allocation. Furthermore, encrypted data is offloaded to or received from the Cloud layer for secure communication when additional processing or storage is required. The proposed approach was validated using MATLAB, and the results demonstrated superior makespan performance when compared to least association, round-robin, and weighted round-robin approaches. However, within the Secure-M2FBalancer model, the authors solely considered response time as the primary factor for resource allocation at the Mist layer. From our perspective, this single factor may not be sufficient for precise decision-making in server allocation. Therefore, we suggest the inclusion of additional factors to enhance the accuracy of the decision-making process. Furthermore, the criteria used to determine the state of Fog servers, whether overloaded or underloaded, are not explicitly defined in the study.

Mist computing can play a vital role in improving the data collection for IoT devices as the processing is at the Edge computing environment. Barik et al. 2017[11] developed a new platform called MistGiS for geospatial big data and applied it in two cases: Tourism Information Infrastructure Management and Faculty Information Retrial System. The Raspberry Pi microprocessor was utilised to build the framework. The study results found that Mist computing can assist Cloud and Fog computing in providing better analysis for geospatial big data. This platform serves as a means to only highlight the significance of Mist computing in evaluating other layers, thus lacking novelty in its approach.

Stavrinides et al. (2021) introduced a scheduling heuristic that considers security and cost for real-time IoT data processing, taking into account various security requirements. The algorithm is divided into three stages:

1. In the task selection stage, tasks are prioritized based on their deadlines.
2. The VM Filtering stage aims to select the appropriate security level.
3. The VM selection stage determines the suitable VM for task processing based on the earliest estimated finish time.

To validate their approach, the researchers implemented a custom discrete-event simulator in C++. The results of the study demonstrated that the proposed approach outperformed the baseline policy of security-aware heuristics (SAH) in terms of deadline miss ratio, total cost of Cloud resources, and average response time. However, we believe that additional factors, such as task size, should be considered when prioritizing tasks.

B) Fuzzy logic and load balancing

In this section, we provide an overview of previous studies conducted between 2018 and 2021 that leveraged the fuzzy logic system to enhance decision-making in their respective approaches. These studies have been compiled to showcase the efficacy of employing fuzzy logic in making accurate decisions pertaining to task scheduling and load balancing in IoT systems.

Ali H et al. 2021[13] proposed a fuzzy logic algorithm called Real-time Task Scheduling (FLRTS) to enhance the execution of the tasks of IoT applications at the Fog layer. The algorithm works as a filter to divide the tasks into two categories: Fog group and Cloud group, to select the environment to execute data to either the Fog or the Cloud. Furthermore, the algorithm works inside the Fog broker

at the Fog layer for the purpose of classification. The algorithm incorporates five inputs: CPU Utilization, Storage Utilization, Bandwidth Utilization, Task Deadline, and Network Latency, to determine the optimal execution environment for data, whether it should be processed in the Fog or the Cloud. To evaluate the proposed algorithm, the IFogSim simulator, a Java-based simulation toolkit, was employed. The simulation results showed that the proposed approach outperforms the existing algorithms First In First Out (FIFO) and Short Job First (SJF) scheduling algorithms and the Real-Time Task Processing (RTP) in terms of makespan, delay, success ratio and average turnaround time. To achieve improved response time and decision-making, it is crucial for the filtering process to be situated closer to IoT sensors rather than at the Fog layer. Filtering should ideally take place at an early stage, and in our perspective, the Mist layer is the most suitable for task filtering since it is the layer closest to the Edge layer (which represents the sensors layer).

Das A et al. 2020 [14] proposed a User Categorization using Fuzzy Logic (UCFL) algorithm to improve the overall performance of IoT networks. The framework consists of three layers; the second layer (Fog devices) works as a chine to allow the first layer (users) to collect data from the third layer (sensor nodes). Due to the power constraints in the IoT network, the sensor nodes are not involved in the authentication process to save power. Instead, only nodes with higher capacity, specifically the Fog nodes in this framework, are engaged in the authentication process. The algorithm takes into account three inputs: user experience, user knowledge, and user recommendation, to generate an output that classifies users into three categories: high trusted, medium trusted, and low trusted. This classification plays a crucial role in reducing the number of authentication phases required. High trusted users do not require authentication phases as they have high experience, knowlEdge and recommendation, while medium trusted users only require one phase of authentication. On the other hand, low trusted users require two phases of authentication as they have low experience, knowlEdge and recommendation. To evaluate the proposed algorithm, a Raspberry Pi was employed as a user, while a laptop served as a Fog node to facilitate the communication process. The simulation results clearly demonstrated that the proposed approach outperformed other algorithms such as CoAPMicro, CoAPBlip, and HTTP/UDP in key metrics including handshake duration, memory consumption, average response time, and computation cost. In the UCFL framework, the authors achieved performance improvements by reducing certain users' authentication processes and adjusting security levels. However, it is important to note that, from our perspective, enhancing network performance should not compromise the system's security, either directly or indirectly.

In their work, Reddy A et al. (2021)[15] proposed an algorithm called feedback-based optimized fuzzy scheduling approach (FOFSA) to enhance power efficiency, execution time, and makespan in IoT networks while meeting Quality of Service (QoS) requirements. The algorithm intelligently allocates appropriate resources from Cloud resources, including Virtual Machines, applications, and data centres, to compute tasks. Resource estimation for Cloud computation takes place at the Fog layer. The proposed algorithm was evaluated using MATLAB 2017b, with a focus on applying it to a rainfall prediction mechanism. The algorithm takes five inputs: Clouds, Tasks, Expected Completion Time, User Set Priority, and Decision Making, and generates rainfall predictions by estimating suitable VMs. Simulation results demonstrated that the FOFSA approach effectively reduced power consumption, execution time, and makespan compared to existing algorithms such as optimized fuzzy bee-based scheduling algorithm (OFBSA), Dynamic Duty Scheduling for green sensor Cloud applications (DDSA), an adaptive tasks distribution method for green Cloud computing, and osmosis load balancing algorithm (OLB). However, one limitation of the FOFSA algorithm is the lack of criteria and measurements to determine input values. For example, when considering the time to complete a task, there is no standard measurement to classify it as low or high, which may impact the accuracy of the algorithm. Additionally, the computing capacity of resources was not explicitly considered in the algorithm.

Cuka M et al. [16] 2018 proposed two fuzzy logic algorithms to select IoT devices in opportunistic networks for task computing. The challenges that opportunistic networks face are considered to determine the inputs for the two fuzzy logic algorithms, namely FBS1 and FBS2. In FBS1, there are three inputs: the waiting time, the storage and the remaining energy to assist in

 doi:10.20944/preprints202308.0310.v1

generating an output for the IoT device to process in an opportunistic network. In FBS2, one new parameter is added to the three inputs in the FBS1 to achieve the same objective. The FBS1 is less complex than the FBS2; however, the simulation results demonstrated that the FSB2 achieved better results than the FBS1 in selecting IoT devices . One drawback of this approach is that the authors primarily focused on selecting a suitable source for data computation without considering the characteristics of the data itself.

In their work, Haripriy A et al. (2019) [17] introduced a lightweight fuzzy logic algorithm called Secure-MQTT to enhance the security of IoT applications, specifically in healthcare monitoring, by utilizing the MQTT communication protocol. The algorithm incorporates an efficient intrusion detection system to safeguard IoT applications against denial of service (DoS) attacks. The Secure-MQTT algorithm takes two inputs, namely the Connection Message Ratio (CMR) and Connection Acknowledgment Message Ratio (CAMR), to detect anomalous behaviours and generate an output indicating whether the behaviour is normal, abnormal, or indicative of an attack. To evaluate the proposed approach, the Contiki simulator COOJA was employed. The simulation results demonstrated that Secure-MQTT exhibited superior performance in detecting attacks compared to existing methods, thereby enhancing the overall security of IoT applications.

In their study, Khalil A et al. (2019) [18] presented a fuzzy logic algorithm designed to assess the trust level of nodes for data collection purposes. The algorithm's evaluation of node trust is based on three inputs: Device Physical Security, Device Security Level, and Device Ownership Trust. These inputs are processed by the fuzzy logic system to generate an output, which represents the trust rating of a node on a scale from one to ten. A rating of one indicates the lowest level of security, while a rating of ten indicates the highest level of security. To assess the proposed approach, the researchers utilized FISpro 3.5 and conducted evaluations on various scenarios involving different input values. The simulation results revealed that IoT nodes experienced increased trust levels when the inputs had higher parameter values, signifying better security attributes for the selected nodes.

Sankar S et al. 2018 [19] proposed a fuzzy logic-based energy-aware routing protocol (FLEARPL) to prolong the network lifetime and reduce the power consumption of IoT applications while using RPL. The algorithm assists in selecting the best route during transmission via RPL. It uses three inputs: the routing metrics load, residual energy (RER) and expected transmission count (ETX) to estimate the quality of routes for better selection. The Cooja simulator was used to evaluate the proposed algorithm. In comparison with RPL, MRHOF-RPL and FL-RPL, the simulation results showed that the proposed algorithm outperforms them in terms of the packet delivery ratio and network lifetime.

Ramkumar K [20] proposed a Fuzzy-based Relay Node Selection and Energy Efficient Routing (FRNSEER) to enhance QoS performance in WSNs. The FRNSEER selects the best sink node among active relay nodes to collect data from sensor nodes; sensor hubs were applied to link sensor nodes to the sink. Energy and Utility are the parameters to determine active nodes. After defining the active nodes, fuzzy logic rules are applied to select the best node from the active nodes to work as a sink node. The selection of the sink node is based on three inputs: Virtual Distance (VD), Virtual Energy (VE) and Node bandwidth. A network simulator (NS2) was employed to evaluate the proposed approach. The simulation results demonstrated that the proposed approach achieved better results than the Fuzzy-based Hyper Round Policy (FHRP) and Neural Network-based Localisation Scheme (NNBLS) in terms of energy efficiency and packet rate.

In their study, Radhika S et al. (2021) [21] introduced the Fuzzy Based Sleep Scheduling Algorithm, which incorporates Machine Learning techniques to enhance the energy efficiency of Wide Area Networks (WANs). The primary objective of the algorithm is to minimize the message transmission overhead, thereby reducing energy consumption and extending the overall network lifetime. This is achieved through a simplified cluster restructuring process. The algorithm employs Machine Learning techniques to reduce data transmission, along with two fuzzy logic algorithms that estimate cluster updates and sleep cycles. Inputs such as distance, residential energy, and data rate are utilized in the fuzzy logic algorithms to make accurate estimations. When the node detects similar data, it switches to sleep mode to conserve energy. The proposed approach was evaluated

 doi:10.20944/preprints202308.0310.v1

using MATLAB. Simulation results demonstrated that the proposed approach surpasses existing methods such as Data Density Correlation Degree (DDCD) and Energy Efficient Clustering with Correlation and Random Update (EECRU) in terms of network lifespan. The Fuzzy Based Sleep Scheduling Algorithm, with its integration of machine learning and fuzzy logic techniques, offers significant improvements in energy performance and network longevity.

3. Motivation and aims

Previous research in IoT systems, specifically in the areas of fog computing and cloud computing, has focused on task distribution and load balancing among computing layers. The primary goal has been to reduce power consumption and processing time by offloading tasks from one layer to another. However, the offloading process itself requires power, time, storage, and computing capacity. Therefore, it is crucial to minimize the offloading process in order to save energy, time, storage, and computing resources in the IoT system. However, this reduction must be achieved without negatively impacting the system, such as causing delays.

The proposed approach aims to reduce the offloading process by distributing tasks among computing layers in the early stages and leveraging this reduction. By prioritizing tasks and allocating resources to them at an early stage, we can achieve our objective, as tasks will be sent directly to their designated resources unless exceptional cases arise, such as when the resources are full. To achieve this, we allocate resources for tasks processed at the mist layer. The mist layer, located close to the edge layer (medical sensors), plays a crucial role in assisting patients with critical conditions and facilitating real-time monitoring for quick decision-making.

When allocating resources, we consider two main factors that help us make better decisions and minimize task offloading. The first factor relates to the health condition of the patients, while the second factor pertains to the capacity of the resources. In this context, we only consider the mist nodes and the fog nodes, and we do not take into account the capacity of the cloud since it is centralized. By leveraging fuzzy logic systems, which are powerful tools for decision-making and delivering accurate results, we can estimate and predict the optimal resource allocation for healthcare tasks. To study and analyse the proposed system, we will adopt healthcare systems as a model, focusing on the MFHS (Mist-Based Fuzzy Healthcare System).

Our proposed MFHS aims to achieve the following:

- Decision-making at the extreme edge of the network, facilitated by the mist broker, to enable fast decision-making and reduce processing time.
- Estimating patients' healthcare conditions and allocating resources based on their conditions.
- Prioritizing data packets for patients with critical conditions, ensuring they are served first.
- Minimizing transfer time by allocating resources at the mist broker, located at the extreme edge of the network.
- Reducing power consumption by eliminating the need for data offloading at all layers except the mist layer.

The proposed approach

The MFHS (Mist-based Fuzzy Healthcare System) operates across four layers: the Edge layer, the Mist layer, the Fog layer, and the Cloud layer, each playing a crucial role in processing data for healthcare systems. In the following section, we provide a detailed description of these layers and outline their functionalities within our approach. Additionally, Figure 1 provides a concise overview of the system design, illustrating all the components utilized in our proposed approach.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

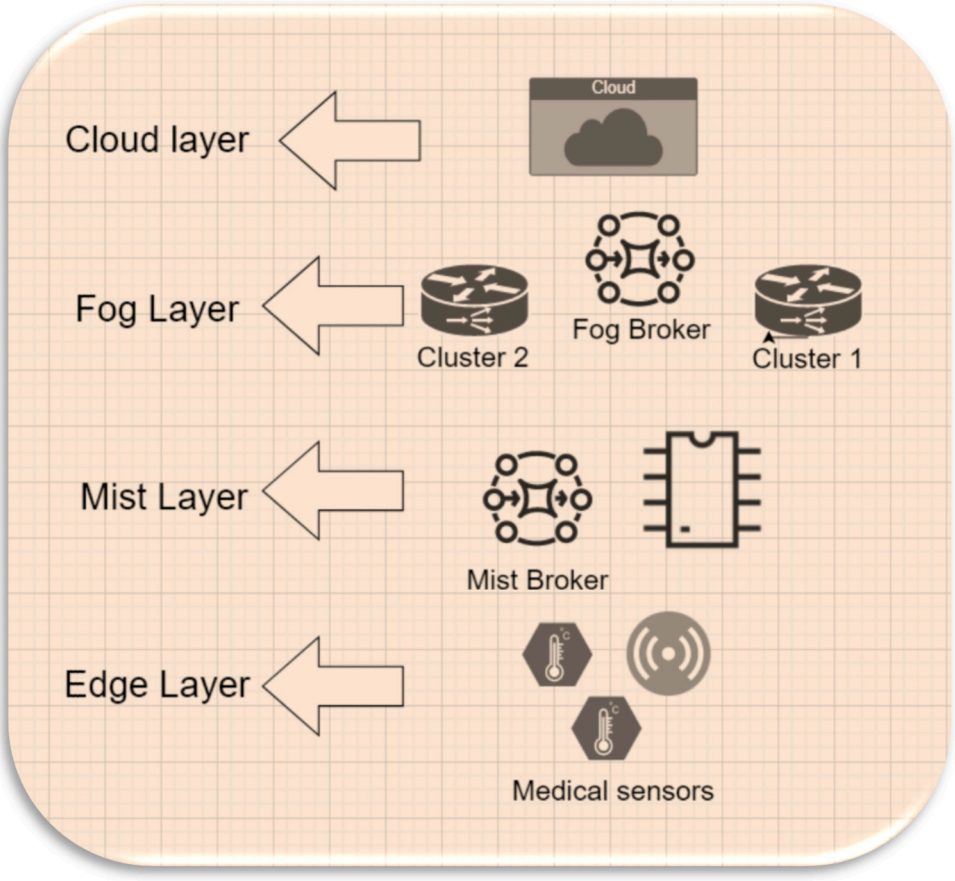

Figure 1. Proposal Design.

1) Edge layer: It collects medical data such as body temperature using sensor devices. The Edge layer sends the sensed data to the Mist layer, which categorises data based on the patient's condition. The Edge layer only sends the sensed data that a Mist layer requires for categorisation, which means some medical sensor devices can be removed without affecting the system.

2) Mist layer: The Mist layer receives the sensed data from the Edge layer. It categorises data based on the patient's health condition and the computing capacity of Mist using two fuzzy logic systems, namely MFHS1 and MFHS2. MFHS1 focuses on data categorization, where the Mist broker employs fuzzy rules to classify the data based on the patient's health condition and its priority. On the other hand, MFHS2 is responsible for estimating the computing capacity of the Mist nodes, enabling the system to determine whether the data should be processed in the Fog, Cloud, or within the Mist layer itself.

3) Fog layer: The Fog layer via the Fog broker is responsible for exceptional cases, such as when the Mist layer is unable to process data due to storage or capacity limitations. The Fog broker takes charge of distributing this data among the Fog nodes based on the clustering of these nodes

4) Cloud layer: The Cloud layer receives high-priority cases directly from the Mist layer for processing. Additionally, it acts as a recipient of data when the computing capacity of both the Mist and Fog nodes is insufficient to handle the workload.

4. Phases of the proposed approach

Before we go further into the design, it is essential to define the fuzzy logic system. Fuzzy logic is a predicting system to make accurate decisions based on fuzzy rules. Figure 2 shows how a fuzzy system works. The fuzzy logic system consists of two main phases: Fuzzification and Defuzzification.

Fuzzification converts crisp values to fuzzy set values; in this stage, it is essential to draw the membership function and fuzzy sets and use the established fuzzy rules to generate fuzzy set outputs

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

[22]. Many types of membership functions convert crisp data to fuzzy sets; however, in this design, the Triangular membership function is selected as it provides accurate results and suits the design.

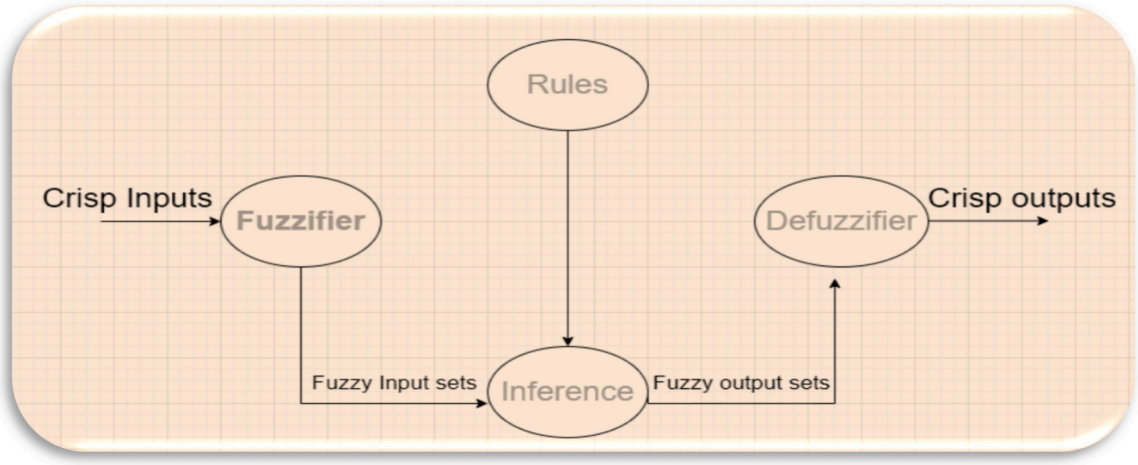

Figure 2. Fuzzy logic system structure.

Equation Triangular:

$$\mu_A(X) = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{m-a} & a < x \le m \\ \dfrac{b-x}{b-m} & m < x < b \\ 0 & x \ge b \end{cases}$$

Defuzzification converts the output values of the fuzzy set into crisp values by using some linguistic rules of If-Then and logical operators; in this design, we will use the AND operator. There are many equations to calculate defuzzification, and the centroid of area (CoA) is used in this design.

Defuzzification Equation:

$$x^* = \frac{\sum_{i=1}^{n} x_i * \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}$$

Our proposed approach consists of two phases. Phase 1 assists in data categorisation. Phase 2 assists in allocating resources depending on the categorised data in phase 1 and Mist node capacity.

Phase 1

The first phase of our proposed system occurs within the Mist broker, where data categorization takes place using a Fuzzy System (FS). This process aims to effectively handle patient data across various health conditions and determine the appropriate server resource from the Mist, Fog, and Cloud layers. The Edge layer focuses on three medical sensors, namely the body temperature (BT) sensor, glucose level (GL) sensor, and heart rate (HR) sensor, which record the patients' health data. These recorded data are initially transmitted to the Mist layer, specifically the Mist broker, where they undergo categorization using the FS. The Mist broker classifies the patients' health data into three priority levels: high priority (critical cases), medium priority (susceptible to disease), and low priority (healthy), based on the patients' health conditions.

The FS can accomplish that by converting the actual data (crisp inputs) into linguistic values (fuzzy input set) using the fuzzification method to determine and estimate patients' health conditions based on fuzzy rules to generate fuzzy output sets. Then these outputs are de-fuzzified to convert linguistic values to crisp values and count them as the results of the FS. The results of this defuzzification could be of one of the three priorities: high priority, medium priority and low priority. Table 1 represents how we estimate the health condition of patients.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

Table 1. The reading of the medical sensors.

BT	HR	GL
Low **35.5 C to 36.5 C**	Slow (<70 bpm)	Less (<60 mg/dl)
Normal **36.1 C to 37.2 C**	Average 60 bpm to 110 bpm	Normal 50 mg/dl to 140 mg/dl
High **37 C to 38 C**	Fast 100 bpm to 140 bpm	High 130 mg/dl to 240 mg/dl

The membership function for inputs BT, HR and GL is designed using the MATLAB Fuzzy Toolbox in Figures 3, 4 and 5, respectively.

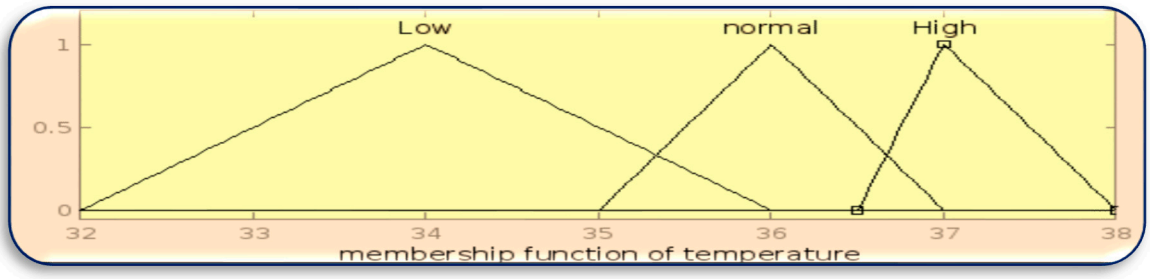

Figure 3. Membership function of BT.

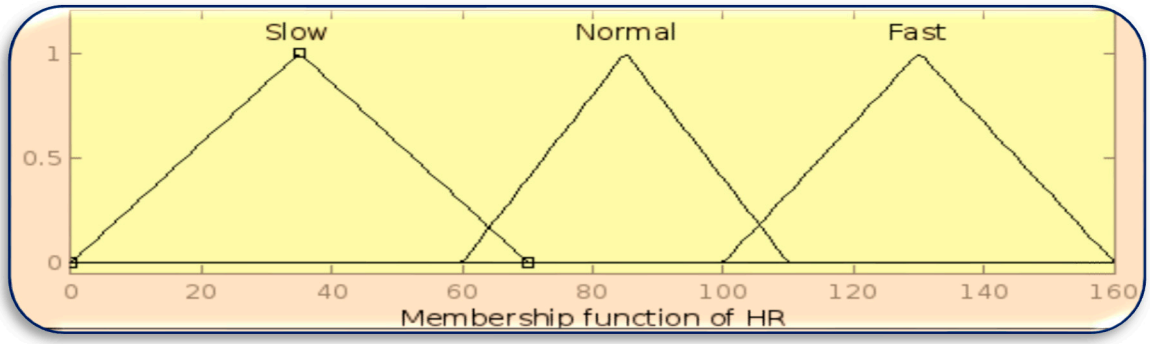

Figure 4. Membership function of HR.

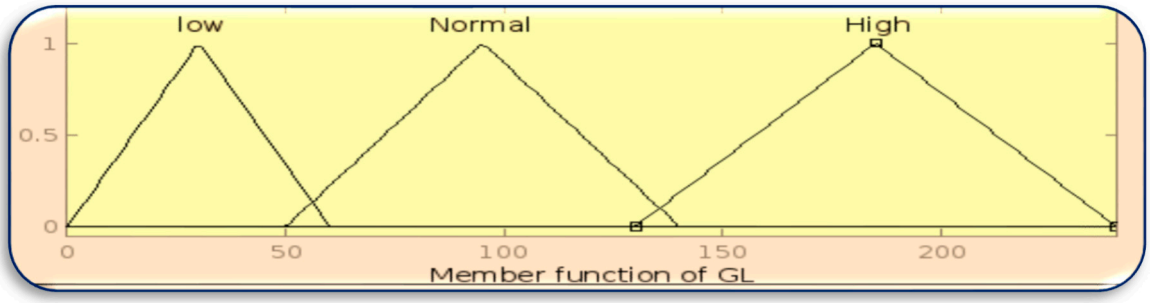

Figure 5. Membership function of GL.

In Phase 1, the number of rules in the fuzzy logic system depends on the number of sensors in the experiment and how many readings each sensor can sense, as follows.

The number of rules = (Number of reading BT) * (Number of reading HR) * (Number of reading GL).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

Here, the number of rules= 3*3*3 = 27 rules by using ***"If-Then" and "And" linguistic rules as logical operators*** to take minimum membership value. The rules are represented in Table 2. In the health score calculation, each normal health condition is counted as 30 points, medium 10 points and low 5 points. Figure 6 shows the design of the fuzzy logic system to generate the health score to assist in data categorisation, and Figure 7 shows the membership function of the health score.

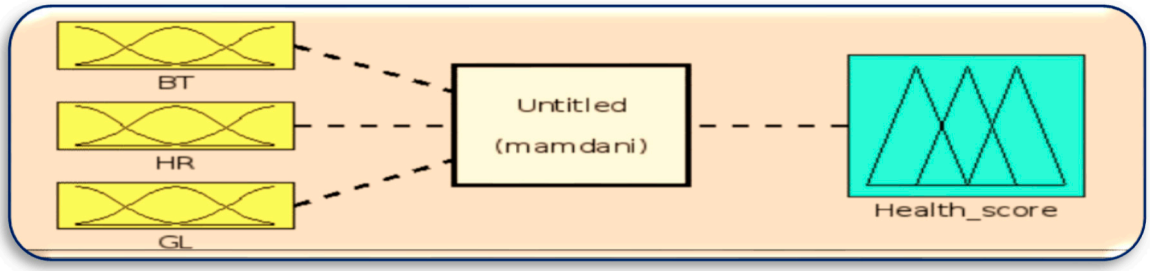

Figure 6. FLS for health score.

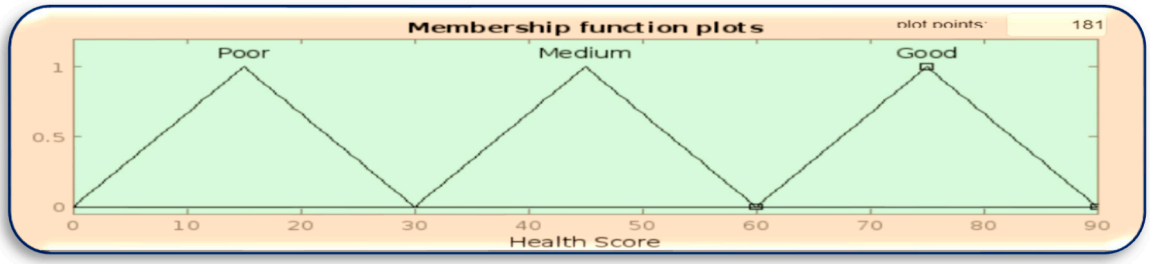

Figure 7. Membership function of health score.

Table 2. The fuzzy rules for data categorisation.

BT	HR	GL	Health score	Patient Health condition	Data priority
Low	Slow	Less	Poor	Critical	High
Low	Slow	Normal	Medium	Exposed to diseases	Medium
Low	Slow	High	Poor	Critical	High
Low	Average	Less	Medium	Exposed to diseases	Medium
Low	Average	Normal	Good	Healthy	Low
Low	Average	High	Medium	Exposed to diseases	Medium
Low	Fast	Less	Poor	Critical	High
Low	Fast	Normal	Medium	Exposed to diseases	Medium
Low	Fast	High	Poor	Critical	High
Normal	Slow	Less	Medium	Exposed to diseases	Medium
Normal	Slow	Normal	Good	Healthy	Low
Normal	Slow	High	Medium	Exposed to diseases	Medium
Normal	Average	Less	Good	Healthy	Low
Normal	Average	Normal	Good	Healthy	Low
Normal	Average	High	Good	Healthy	Low
Normal	Fast	Less	Medium	Exposed to diseases	Medium
Normal	Fast	Normal	Good	Healthy	Low
Normal	Fast	High	Medium	Exposed to diseases	Medium
High	Slow	Less	Poor	Critical	High
High	Slow	Normal	Medium	Exposed to diseases	Medium
High	Slow	High	Poor	Critical	High
High	Average	Less	Medium	Exposed to diseases	Medium
High	Average	Normal	Good	Healthy	Low

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

High	Average	High	Medium	Exposed to diseases	Medium
High	Fast	Less	Poor	Critical	High
High	Fast	Normal	Medium	Exposed to diseases	Medium
High	Fast	High	Poor	Critical	High

Phase 2

In the second phase, Mist Broker (MB) focuses on the Mist's computational capacity (see Equation 1 below) and data priority to allocate resources for healthcare services. MB selects one of the three resources: Mist, Fog and Cloud, to provide services for the healthcare system as follows.

First, MB directly transfers high-priority data to the Cloud and allows healthcare providers to access these critical data. In addition, MB helps patients with urgent conditions in real-time processing to make quick decisions as the MB is very close to the sensing devices.

Second, MB sends the medium priority data to the Mist nodes of available capacity. The data is transferred to the next available Mist node if a Mist node is overloaded. If all Mist nodes have insufficient computing space for the medium priority, the data is transferred to the Fog broker.

Third, MB sends the low-priority data to the Mist nodes of available capacity. The data is transferred to the next available Mist node if a Mist node is overloaded. However, if all Mist nodes lack adequate computing space (i.e., low and medium computing capacity) for low-priority data, it is then directed to the Fog broker for further processing, as indicated in Table 3.

Moving to the subsequent layer, the Fog broker is equipped with a load balancer responsible for distributing the medium and low-priority data received from the Mist broker among the available Fog nodes. This distribution is based on two factors: the remaining computing capacity of each Fog node and a clustering technique. The calculation of the remaining capacity of a Mist node, as described by **Equation 1**, plays a crucial role in determining whether the healthcare services should be computed within the Mist node itself or whether the healthcare data should be redirected to another Mist node or the Fog broker in the event of an overloaded Mist node:

$$\text{Remaining capacity} = C - \sum_{i=1}^{n} Pi \times Si \qquad \text{eq (1)}$$

$$\text{Remaining capacity percentage} = \frac{C - \sum_{i=1}^{n} Pi \times Si}{C} \times 100$$

The equation consists of four factors to determine the computing capacity of a Mist node:

- C is the capacity of a Mist node.
- Pi is the packet arrival rate for i as a data packet.
- Si is the size of the data packet i.
- n is the number of data packet

In Phase 2, the number of rules in the fuzzy logic system depends on the number of sensors in the experiment and how many readings each sensor can sense as follows:

The number of rules = (Number of data priority) * (Number of Computational capacities of Mist node)

Here, the number of rules= 3*3= 9 rules by using linguistic rules of ***"If-Then" and "And" as logical operators*** to take minimum membership value. The rules are represented in Table 3.

Table 3. The fuzzy rules for server allocation.

Data priority	Computational capacity of Mist node	Source Allocation
High	High	Cloud
High	Medium	Cloud
High	Low	Cloud
Medium	High	Mist
Medium	Medium	Mist
Medium	Low	Fog
Low	High	Mist
Low	Medium	Fog

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

Low	Low	Fog

Figures 8, 9, 10, and 11 in MATLAB depict the design of the fuzzy logic system responsible for server allocation. This system takes two inputs, namely data priority and Mist capacity, and generates an output that determines the server allocation.

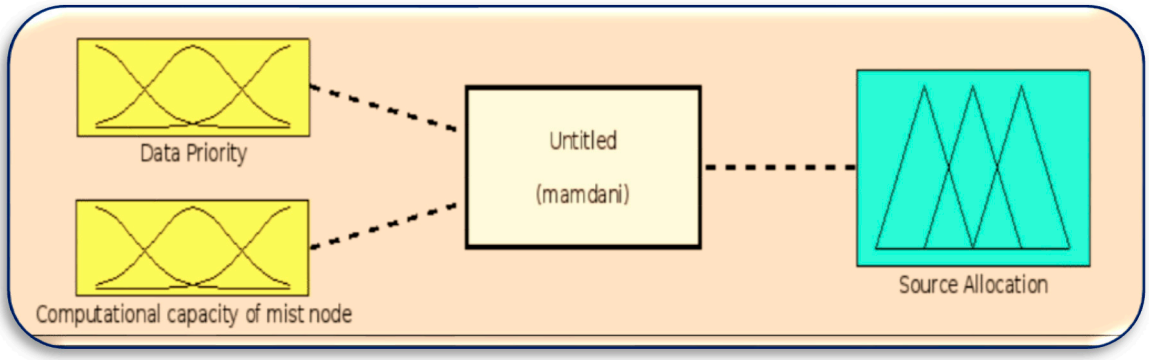

Figure 8. Fuzzy logic system for server allocation.

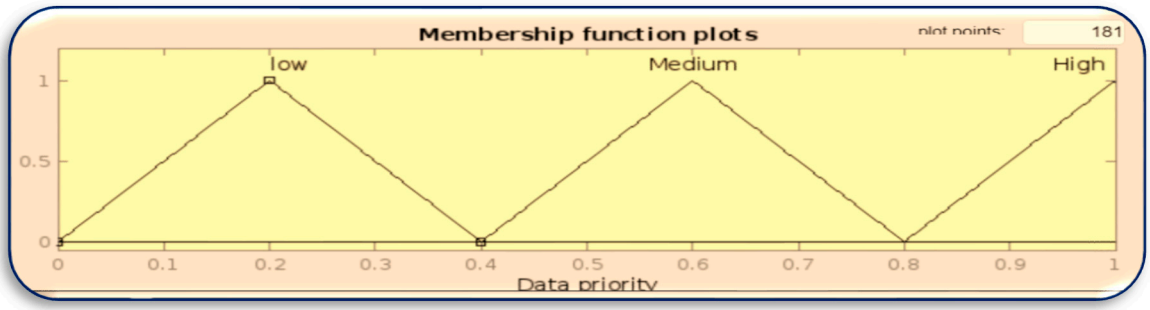

Figure 9. Membership function for data priority.

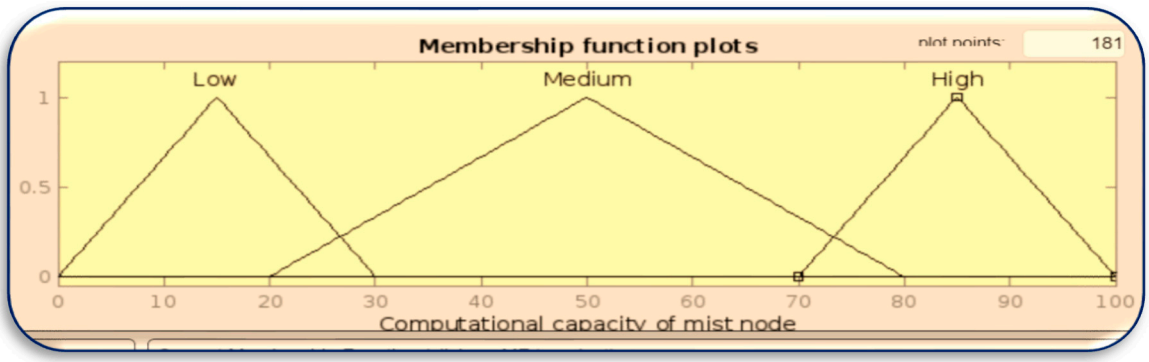

Figure 10. Membership function for Mist Capacity.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

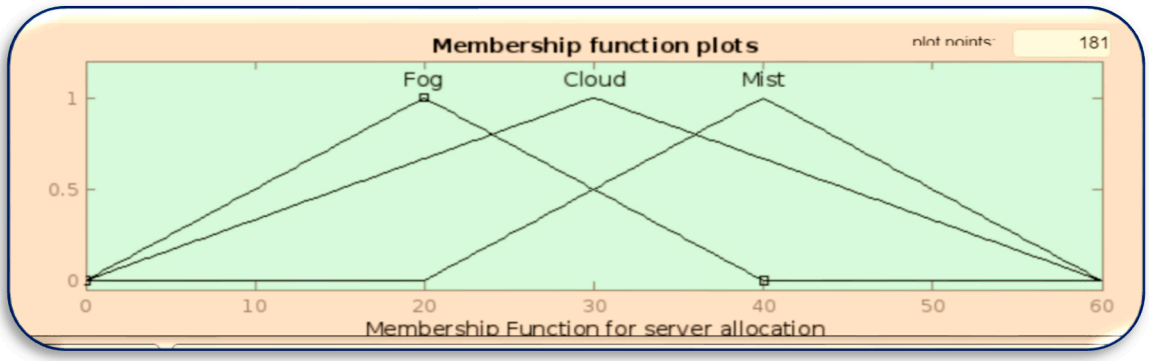

Figure 11. Membership function for server allocation.

Fog Broker

The Fog broker receives only two types of data: low-priority and medium-priority, which are passed on by the Mist broker. To address this distinction in data types, we have designed a clustering scheme within the Fog layer. The Fog nodes are divided into two clusters, with the Fog broker overseeing their operation. Figure 10 provides an overview of the workflow within the Fog layer. Cluster1 is responsible for computing the medium-priority data, while Cluster2 handles the low-priority data. This categorization aims to minimize execution time and the offloading process by assigning appropriate resources to each data type based on priority. To achieve this, the Fog nodes are divided into clusters based on their remaining computing capacity. The first cluster comprises the Fog nodes with higher remaining capacity, while the second cluster consists of nodes with lower remaining capacity. The decision-making process for medium-priority data prioritizes Fog nodes with higher remaining capacity, ensuring real-time processing without the need for task offloading or delays. The choice of remaining computing capacity as the main factor enables medium-priority data to make fast decisions for real-time processing. On the other hand, low-priority data is assigned to Fog nodes with lower remaining computing capacity since their real-time processing requirements are less critical compared to medium-priority data.

When the remaining energy or computing capacity of a selected Fog node falls below a threshold value of 25%, the data is rerouted from one Fog node to another or to the Cloud. The computation of a Fog node's remaining computing capacity is determined by Equation 1.

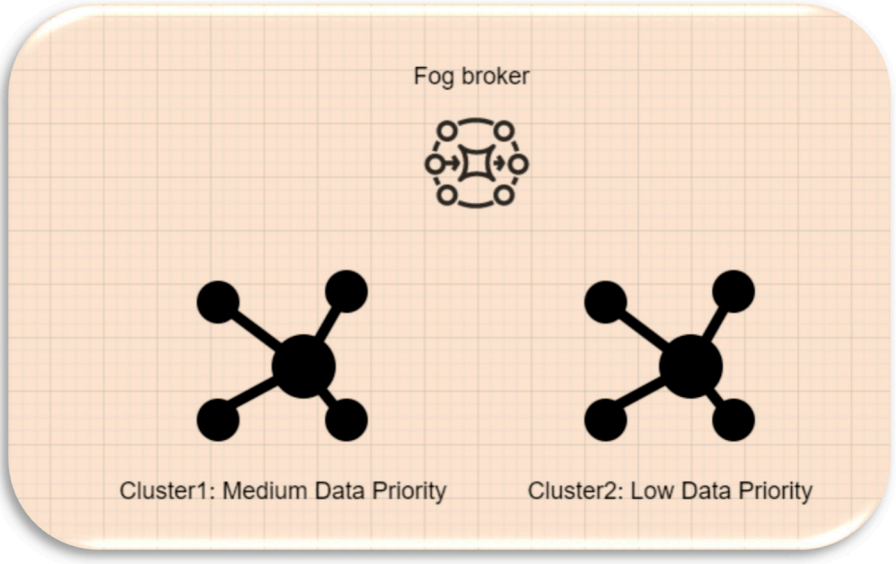

Figure 12. Fog nodes clusters.

5. Experimental setup

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

Eclipse and MATLAB are used to validate MFHS.

MATLAB generates the fuzzy outputs to determine the data priority and the resource allocation based on fuzzy rules.

Eclipse simulates the process environment (Edge, Mist, Fog, and Cloud) and calculates the power consumption, allocation time and processing time.

In the experiment, six sensors are used at the Edge layer, two for the body temperature, two for the heart rate and two for the glucose level. Two Mist nodes and one Mist broker are used at the Mist layer. Six Fog nodes and one Fog broker are used at the Fog layer, and three central Clouds are utilized at the Cloud layer. Table 4 summarizes the tools employed in the experiments, Table 5 illustrates the used nodes, and Figure 13 shows the process sequencing.

Table 4. System Configuration.

Parameters	Configuration
Processor	11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 1.69 GHz
Language	Java
Integrated Development Environment (IDE)	Eclipse
Development Kit	Java Development Kit (JDK) 17
Fuzzy rules integration	MATLAB

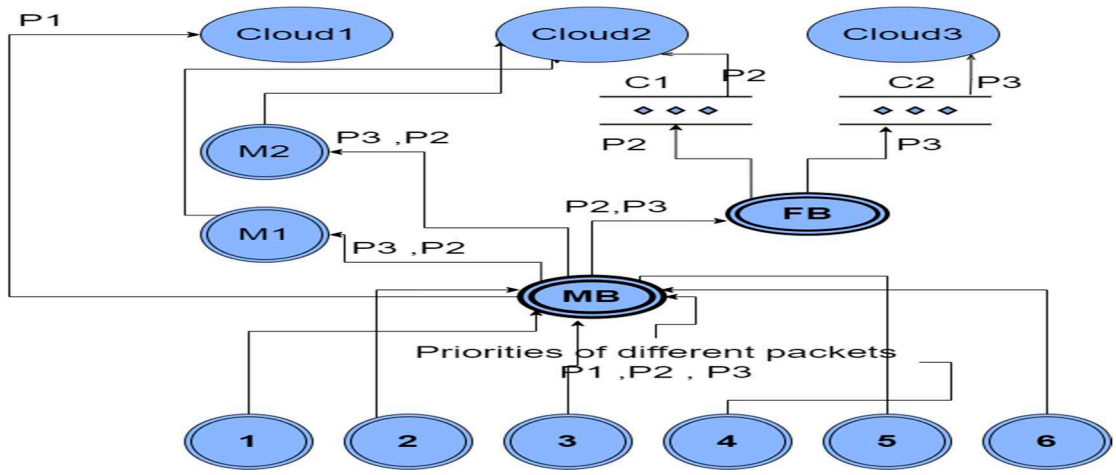

Figure 13. Work Sequence.

Table 5. Nodes Description.

Nodes	Description
1,2,3,4,5,6	BT , HR , GL nodes at the Edge layer
MB	Mist broker
FB	Fog broker
F1, F2,F3,F4,F5,F6	Nodes at the Fog layer
CL1 , CL2 , CL3	Nodes at the Cloud layer
C1 and C2	The two Fog clusters
M1 and M2	Mist nodes
CC	Computational Capacity

In the diagram, six Fog nodes are formed into two clusters based on the computational capacity (CC). Each cluster consists of several Fog nodes. Cluster one (C1) represents Fog nodes with a higher CC, and cluster two (C2) represents Fog nodes with a lower CC.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

For an illustration, let's assume in Table 6 that we have six Fog nodes, namely F1, F2, F3, F4, F5, and F6, with their respective CC. Based on our clustering principle, C1 will consist of F1, F4, and F6, while C2 will contain F2, F3 and F5. Accordingly, C1 is responsible for receiving the medium data priority from the Fog broker, and C2 is responsible for receiving the low data priority from the Fog broker.

The Fog clusters continue to process data until the CC of any Fog node goes below a threshold value of 25% of its capacity; when that happens the data will be offloaded to the Cloud to avoid damaging the node.

Table 6. Fog nodes computational capacity.

Node Name	Computational Capacity
F1	360
F2	60
F3	120
F4	280
F5	40
F6	160

6. Evaluation

The evaluation of this experiment is based on two factors: the total energy consumption and the processing time.

The total energy consumption (Etotal) consists of three different types of energy: the transmission of the packets (Etr), the classification of the packets (Ec) into high, medium and low priority and the resource allocation of the categorized packets (Ea) as follows:

$$Etotal = Etr + Ec + Ea \qquad \text{eq (2)}$$

Etr is calculated based on the size of the packets as in equation (3):

$$Etr = \sum_{i=1}^{n} Sn.Eob \qquad \text{eq (3)}$$

where Eob is the energy cost of transmitting one byte for a single hop, n is the total of transmitted packets, and Sn is the size of the n packet. In this experiment, the energy cost of transmitting one byte of packets for a single hop is assumed to be 0.5mj.

In a similar manner, the total processing time (Ttotal) is determined by considering the same factors as the total energy cost. It can be calculated using the following equation:

$$Ttotal = Ttr + Tc + Ta \qquad \text{eq (4)}$$

where Ttr represents the packet transmission time, Tc denotes the packet classification time, and Ta represents the packet allocation time.

7. Results

The energy costs of the number of VMs used in the Optimized Fuzzy Scheduling Approach (FOFSA) algorithm are shown in Table 7 [15].

Table 7. Energy cost for VMs.

Energy Consumption	Number of VMs
4 KWH	50
10 KWH	100
18 KWH	150
20 KWH	200
23 KWH	250
28 KWH	300

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

In our experimental setup, we make the assumption that a single virtual machine (VM) is capable of processing an average of 10,000 tasks. Consequently, Table 8 provides a breakdown of the energy expenses associated with varying numbers of tasks.

Table 8. Energy cost for Tasks.

Energy Consumption	Number of Tasks
4 KWH	500000
10 KWH	1000000
18 KWH	1500000
20 KWH	2000000
23 KWH	2500000
28 KWH	3000000

In the subsequent calculation, we will consider the number of packets to be equivalent to the number of tasks, with each task corresponding to one packet. For instance, the energy cost associated with 500,000 packets is 4 KWH, as demonstrated in Table 8.

To streamline the computation, the total number of packets is normalised into five sets: 20, 40, 60, 80, and 100 packets. This division aims to simplify the calculation process. Table 9 presents the energy costs incurred by running these newly created sets using the FOFSA algorithm.

Table 9. Energy costs for packets.

Energy Consumption	Number of Packets
0.56 KJ	20
1.44 KJ	40
2.592 KJ	60
2.88 KJ	80
3.312 KJ	100

The computation of energy costs for the Adaptive Task Allocation Technique (ATAT) and Osmosis Load Balancing (OLB) algorithm in [15] follows a similar methodology.

Figure 14 presents a comparison of the energy costs between our proposed MFHS approach and the FOFSA, OLB, and ATAT algorithms discussed in [15], with respect to the number of packets. The chart clearly illustrates that our MFHS approach achieves superior energy cost results compared to the existing methods, particularly as the number of packets increases. This indicates that the MFHS algorithm outperforms its counterparts in handling extensive data analysis, which is essential for IoT networks.

Top of Form

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0310.v1

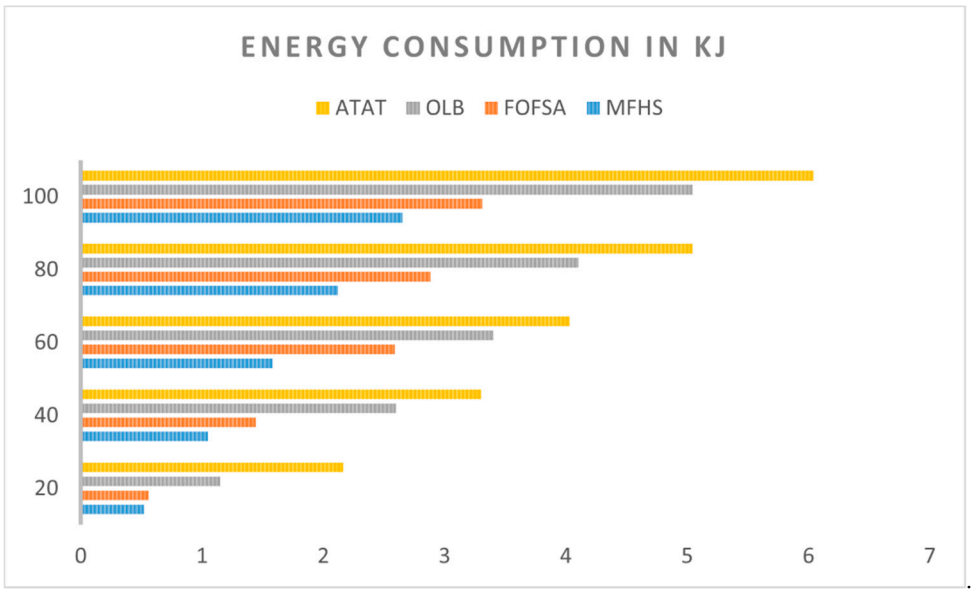

Figure 14. A comparative analysis of energy consumption.

In Figure 15, we present a comparison of the processing time between our proposed MFHS algorithm and the FOFSA method, which has been demonstrated to outperform both ATAT and OLB algorithms [15], with respect to the number of packets. The chart clearly indicates that the processing time of the MFHS algorithm outperforms the existing approach, particularly as the number of packets increases. This suggests that the MFHS algorithm excels in handling big data analyses, which are crucial for IoT networks. This comparison is made without sacrificing generality, highlighting the superior performance of MFHS in terms of processing time.

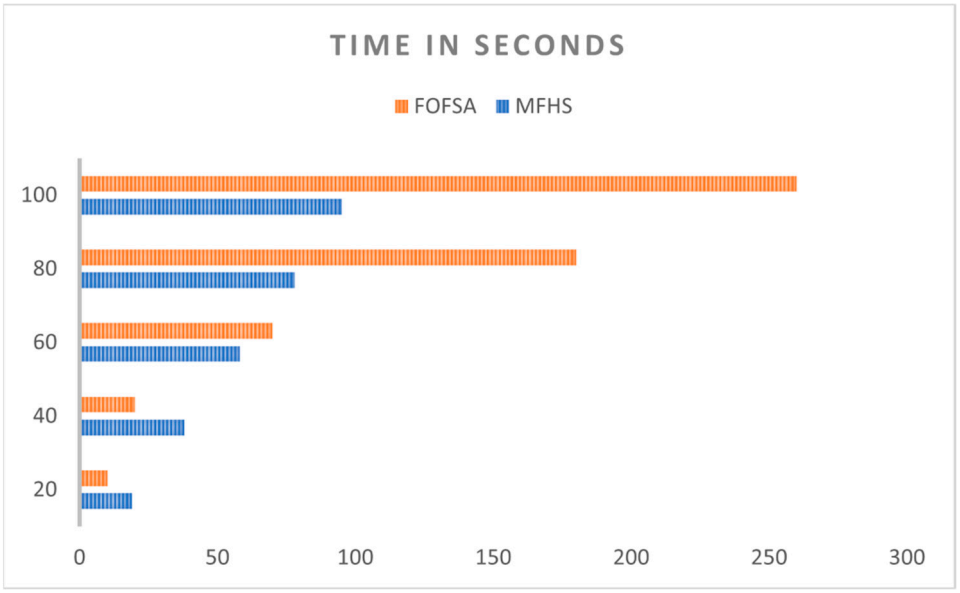

Figure 15. A comparative analysis of processing time.

8. Discussion

The source allocation process is initiated at the early stages within the Mist layer, facilitated by the Mist broker serving as the central distributor. This approach streamlines the offloading steps and reduces offloading time, as all data is pre-allocated at the Mist broker. To achieve this, two Fuzzy logic systems are employed for resource allocation, as mentioned previously:

- **MFHS1** is responsible for data classification.
- **MFHS2** focuses on resource allocation, based on the computing capacity of the Mist layer.

In addition, the Fog broker incorporates a clustering mechanism that takes into account the computing capacity of the Fog nodes. This approach aims to minimize both the processing time and energy cost involved in the system. The implementation of this approach provides promising results in terms of energy cost and processing time for the Mist and Fog brokers, as well as the Mist, Fog, and Cloud nodes. The above results demonstrate superior performance compared to other algorithms such as the Feedback-based Optimized Fuzzy Scheduling Approach (FOFSA), Adaptive Task Allocation Technique (ATAT) and Osmosis Load Balancing algorithm (OLB).

9. Conclusion

The proposed approach in this study focuses on the key aspects of offloading and load balancing, recognizing their potential to enhance the performance of any computing network. To achieve this, fuzzy logic systems were employed to aid in processing, offloading, and workload balancing within IoT networks. The proposed MFHS systems initiate operations at the Mist level, enabling early decision-making and aiming to improve both energy efficiency and processing time. The evaluation of MFHS involved the use of Eclipse and MATLAB, and the results demonstrated successful reductions in processing time and power consumption. Notably, the MFHS approach outperformed the Feedback-based Optimized Fuzzy Scheduling Approach (FOFSA) algorithm, as well as the Adaptive Task Allocation Technique (ATAT) and Osmosis Load Balancing algorithm (OLB), in terms of energy efficiency and processing time.

Author Contributions: Conceptualization, Ziyad Almudayni and Ben Soh; Methodology, Ziyad Almudayni and Ben Soh; Software, Ziyad Almudayni; Validation, Ziyad Almudayni; Formal analysis, Ziyad Almudayni; Investigation, Ziyad Almudayni, Ben Soh and Alice Li; Writing – original draft, Ziyad Almudayni; Writing – review & editing, Ben Soh and Alice Li; Supervision, Ben Soh and Alice Li.

References

1. Alghofaili, Y., and Rassam, M.A.: 'A Dynamic Trust-Related Attack Detection Model for IoT Devices and Services Based on the Deep Long Short-Term Memory Technique', Sensors, 2023, 23, (8), pp. 3814
2. R. K. Barik, S. S. Patra, P. Kumari, S. N. Mohanty, and A. A. Hamad, "A new energy aware task consolidation scheme for geospatial big data application in Mist computing environment," in *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2021, pp. 48–52.
3. F. Hmissi and S. Ouni, "An MQTT Brokers Distribution Based on Mist Computing for Real-Time IoT Communications," 2021.
4. H. Eldin Refaat, "MLITS: Multi-Level tasks scheduling model for IoT Service Provisioning," *Information Bulletin in Computers and Information*, vol. 2, no. 1, pp. 1–9, 2020.
5. E. Rubio-Drosdov, D. D. Sánchez, F. Almenárez, and A. Marín, "A framework for efficient and scalable service offloading in the Mist," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 460–463.
6. F. Hensh, M. Gupta, and M. J. Nene, "Mist-Edge-Cloud (MEC) Computing: An Integrated Computing Architecture," in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2021, pp. 1035–1040.
7. H. Shahid *et al.*, "Machine Learning-based Mist Computing Enabled Internet of Battlefield Things," *ACM Transactions on Internet Technology (TOIT)*, vol. 21, no. 4, pp. 1–26, 2021.
8. A. S. Hosen, P. K. Sharma, and G. H. Cho, "MSRM-IoT: A reliable resource management for Cloud, Fog and Mist assisted IoT networks," *IEEE Internet of Things Journal*, 2021.
9. M. Ejaz, T. Kumar, M. Ylianttila, and E. Harjula, "Performance and efficiency optimization of multi-layer IoT Edge architecture," in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–5.
10. S. S. Tripathy, R. K. Barik, and D. S. Roy, "Secure-M2FBalancer: A Secure Mist to Fog Computing-Based Distributed Load Balancing Framework for Smart City Application," in *Advances in Communication, Devices and Networking*, Springer, 2022, pp. 277–285.
11. R. K. Barik, C. Misra, R. K. Lenka, H. Dubey, and K. Mankodiya, "Hybrid Mist-Cloud systems for large scale geospatial big data analytics and processing: opportunities and challenges," *Arabian Journal of Geosciences*, vol. 12, no. 2, pp. 1–15, 2019.
12. G. L. Stavrinides and H. D. Karatza, "Security and Cost Aware Scheduling of Real-Time IoT Workflows in a Mist Computing Environment," in *2021 8th International Conference on Future Internet of Things and Cloud (FiCloud)*, 2021, pp. 34–41.

13. Ali, H.S., Rout, R.R., Parimi, P., and Das, S.K.: 'Real-Time Task Scheduling in Fog-Cloud Computing Framework for IoT Applications: A Fuzzy Logic based Approach', in Editor (Ed.)^(Eds.): 'Book Real-Time Task Scheduling in Fog-Cloud Computing Framework for IoT Applications: A Fuzzy Logic based Approach' (IEEE, 2021, edn.), pp. 556-564
14. Das, A.K., Kalam, S., Sahar, N., and Sinha, D.: 'UCFL: User Categorization using fuzzy logic towards PUF based two-phase authentication of Foassisted IoT devices', Computers & Security, 2020, 97, pp. 101938
15. Arunkumar Reddy, D., and Venkata Krishna, P.: 'Feedback-based fuzzy resource management in IoT using Fog computing', Evolutionary Intelligence, 2021, 14, (2), pp. 669-681
16. Cuka, M., Elmazi, D., Bylykbashi, K., Spaho, E., Ikeda, M., and Barolli, L.: 'Implementation and performance evaluation of two fuzzy-based systems for selection of IoT devices in opportunistic networks', Journal of Ambient Intelligence and Humanized Computing, 2019, 10, (2), pp. 519-529
17. Haripriya, A., and Kulothungan, K.: 'Secure-MQTT: an efficient fuzzy logic-based approach to detect DoS attack in MQTT protocol for internet of things', EURASIP Journal on Wireless Communications and Networking, 2019, 2019, (1), pp. 1-15
18. Khalil, A., Mbarek, N., and Togni, O.: 'Fuzzy Logic based security trust evaluation for IoT environments', in Editor (Ed.)^(Eds.): 'Book Fuzzy Logic based security trust evaluation for IoT environments' (IEEE, 2019, edn.), pp. 1-8
19. Sankar, S., and Srinivasan, P.: 'Fuzzy logic based energy aware routing protocol for Internet of Things', International Journal of Intelligent Systems and Applications, 2018, 11, (10), pp. 11
20. Ramkumar, K., Ananthi, N., Brabin, D.D., Goswami, P., Baskar, M., Bhatia, K.K., and Kumar, H.: 'Efficient routing mechanism for neighbour selection using fuzzy logic in wireless sensor network', Computers & Electrical Engineering, 2021, 94, pp. 107365
21. Radhika, S., and Rangarajan, P.: 'Fuzzy Based Sleep Scheduling Algorithm with Machine Learning Techniques to Enhance Energy Efficiency in Wireless Sensor Networks', Wireless Personal Communications, 2021, 118, (4), pp. 3025-3044
22. Deabes, W., Bouazza, K.E., and Algthami, W.: 'Smart Fuzzy Petri Net-Based Temperature Control Framework for Reducing Building Energy Consumption', Sensors, 2023, 23, (13), pp. 5985
23. Hu, Y., Meng, J., Li, G., Zhao, D., Feng, G., Zuo, G., Liu, Y., Zhang, J., and Shi, C.: 'Fuzzy Adaptive Passive Control Strategy Design for Upper-Limb End-Effector Rehabilitation Robot', Sensors, 2023, 23, (8), pp. 4042