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

System Model and Problem Formulation to Address Performance Issues in Edge Intelligence

Brinda Parekh [*1], Kiran Amin[2]

Submitted:07/03/2024 **Revised**: 21/04/2024 **Accepted**: 01/05/2024

Abstract: When data processing is implemented in close proximity to end devices with intelligence and ample capabilities, it not only improves real time processing but also increases the effectiveness of generated results and reduces a significant burden on the overall network. Various metrics, such as computational speed, reaction time, CPU demand, network demand, and delay sensitivity, play a crucial role in enabling edge devices to execute complex tasks within time constraints. This paper presents an approach by adopting fuzzy logic to transmit the incoming tasks from the edge devices to one of the edge-cloud servers, which is decided by the edge orchestrator, taking into account various application characteristics. The primary aim of the proposed approach is to enhance task offloading by reducing service time and boosting the efficiency of edge devices. A system model and problem formulation have been designed with the help of which QoS parameters are improved in an edge-cloud environment by taking into consideration the balancing workload among the resources in the network.

Keywords: *Edge-cloud computing, Edge intelligence, edge orchestrator, offload task, fuzzy logic*

1. Introduction

In recent times, billions of smart devices have already in use on the Internet of Things. Due to this, many industry sectors have gained new opportunities in terms of productivity, communication, ease of use, etc. This kind of usage has resulted in massive data generation from such devices. An increase in the data exchange rate from such devices and their processing resulted in time delays. So, nowadays, more focus is placed on improving communication between devices and processing centers. Edge computing addresses these concerns.

With advancements and breakthroughs in artificial intelligence, AI-based applications and services are rapidly developing. Many powerful methods for processing such massive amounts of data in AI technologies have been developed and implemented, which lead to better business decisions. Numerous applications necessitate real-time information, emphasizing the importance of proximity between devices and AI services to mitigate delays and latency. Utilizing the cloud as a centralized processing server results in increased bidirectional data exchange between enddevices and data centers [1].

The primary goal of edge intelligence is to process the data in close proximity to end devices by enabling the end devices with intelligence to make decisions and generate results. Consequently, by considering the privacy of data, the effectiveness and speed of data processing can be increased.

Instead of sending data to the cloud, as opposed to previous approaches, Edge Intelligence (EI) processes data locally, on the edge, closer to the data's place of origin. This means that EI can be described as a collection of interconnected systems and devices that gather data, cache it, analyse it, and perform analysis in proximity to the data's point of origin.

For edge devices to exhibit intelligence, they need components akin to other intelligent real-time systems,encompassing data gathering, referred to as edge caching, training involves learning to generate output from the gathered data, and inference to make decisions. While all these three components are provided with computing services through edge offloading. As illustrated in Figure 1, four distinct components of edge intelligence can be identified [2]:

1Department of Computer Engineering, Ganpat University, India.
ORCIDID:0009-0000-0983-1853
2Faculty of Engineering and Technology, Ganpat University, India.
ORCIDID:0000-0001-6136-4068
**Corresponding Author Email: brin.prkh@gmail.com*

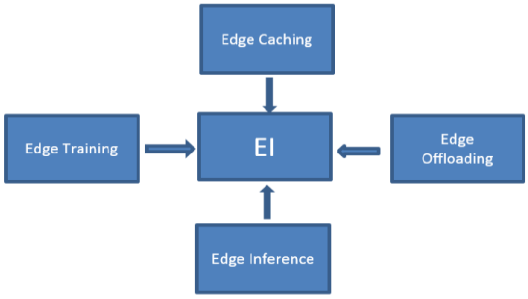

Fig 1. Components of EI

This research work aims to propose a model for offloading work in order to effectively use the resources in edge-cloud systems while handling the demands of latency-sensitive IoT applications. The following is a summarization of key objectives:

- Study various existing algorithms in the Edge environment for improving Quality of Service (QoS) parameters as well as considering fuzzy based approaches.Present a fuzzy logic system for the proposed system.
- Design a system model and problem formulation for the proposed architecture in an edge cloud environment in order to increase the quality-of-service parameters.

The order of the paper is as follows:

Background information on Edge Intelligence is represented in Section 2. Section 3 related work on existing algorithms considering QoS parameters and fuzzy based approaches. The fuzzy logic system for the proposed system is represented in Section 4 as well as the system model and problem formulation. Section 5 presents a conclusion.Background

Edge computing, a revolutionary technology that relocates data creation and processing to the edge of the network, has emerged in the post-cloud age. This allows edge devices to execute calculations for cloud and Internet of Things (IoT) services, processing data to and from, respectively [3]. Any computational and network resources located between data sources and cloud data centers are referred to as "edge" in this article by the author [3].

It has been mentioned in the previous section that IoT applications produce enormous amounts of data, resulting in heavy loads on networks. Utilizing only cloud computing technology to enable these applications might not be effective enough. Furthermore, current intelligent programs frequently use a centralized cloud data center, where users upload their data [2]. This results in delay and latency for the user to upload and process data at the centralized data center, risking the privacy of the data.

By adapting on-demand cloud services and implementing the "vertical" offloading pattern shown in Figure 2(a), IoT systems can accept and execute complex tasks. But the latency issue exists as a problem to enhance the performance of the system. While employing the "horizontal" offloading pattern as shown in Figure 2(b), the above-mentioned issue can be handled by processing the data closer to edge devices having adequate hardware capabilities, as well as by sharing the workload amongst edge devices in small groups. [4].

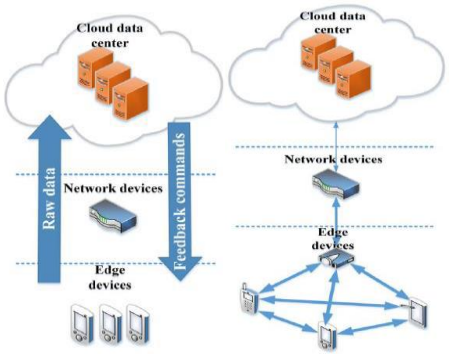

Fig.2(a) Vertical Offloading Pattern

Fig.2(b) Horizontal Off loading Pattern

To address performance challenges like latency, privacy, and security, data processing in the proposed architecture focuses on distributing the tasks among the neighboring edge devices.

Figure 3 illustrates how it is separated into four layers:

- Physical Layer: Comprises of various data sources.
- Edge Computing Services Layer with Intelligence: This layer offers services on edge devices, including data access, data preprocessing, data analysis, and service delivery. It is specifically designed to handle edge clusters and their roles in the proposed approach. A designated edge node, known as the coordinator, initiates the task offloading activity by communicating with other nodes through a broadcast Discovery off-loading request using the Edge Node Discovery service provided by the framework in Figure 4. These requests outline the essential functional specifications that each node must meet to participate in the computation, encompassing both hardware and software requirements. Edge nodes that meet these conditions may either approve or reject the off-loading request. In the first scenario, available nodes could be further selected based on additional (non-functional) factors such as proximity, battery life, potential security risks, etc. Subsequently, the Edge Selection and Configuration service configures the selected nodes, specifying the policies to be implemented

and enforced on each node. Once the cluster is identified, the coordinator oversees work scheduling and synchronization, while the processing phase runs on the nodes in parallel with Orchestration/Lifecycle Management.

- Network transport layer: This layer is the backbone of the Internet of Everything and includes wired and wireless networks, including LAN, WAN, mobile communication networks, and VPNs.
- Application layer: This layer is directly related to users and is in charge of processing requests sent to or received from systems.

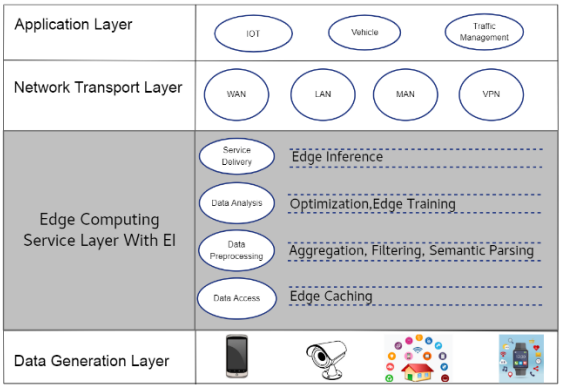

Fig 3. Proposed Architecture for Edge Intelligence [4]

Thus, by using the proposed architecture, an edge computing layer is integrated with edge intelligence to address platform-level application challenges, ultimately enhancing overall performance. The relevant work on current algorithms that take into account QoS and fuzzy-based techniques is shown in the next section.

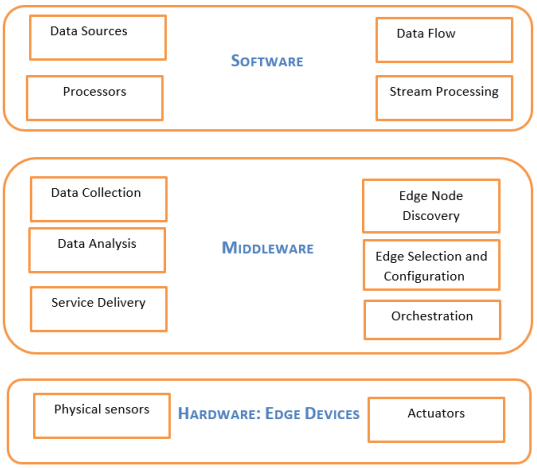

Fig 4. Edge Cluster Framework [4]

2. Related Work

Computation offloading is frequently applied in the field of cloud computing, as stated in [5]. The main aim of offloading is to process the data from mobile devices, which having less capacity in terms of resources, to the cloud nodes in order to increase overall performance and efficiency. Mobile devices use WLAN to offload the task to the Edge/cloud nodes via network edges, which are uniformly distributed over the network. The incoming task is computed by other edge nodes in the network if any of the edge nodesare unable to process it in order to efficiently run IoT applications. This process involves offloading heavy computational tasks to more powerful nodes within the network

For task offloading in an edge-cloud system many parameters play a crucial role, specifically two main groups which consider it [24]:

- Infrastructure characteristics: include resource allocation for specific tasks, the edge server's utilization level, etc.
- Application characteristics: concentrateon computation demand, data transfer rate, and task completion deadlines.

A number of studies [6-9] attempted to balance maximizing total revenue and resource utilization while minimizing service latency, energy consumption, and obligatory costs.

In order to task offloading number of challenges are faced in edge-cloud networks such as reducing latency while enhancing resource utilization. A review and discussion of some studies on the same are mentioned below.

Based on application characteristics: (Computation and communication demands), Wang et al. [10] consider application characteristics for homogenous resources to improve application performance by reducing resource utilization and load balancing. The authors of [11] propose an advanced technique to minimize service latency and decrease power consumption. The approach focuses on mitigating communication and computational delays through the strategic migration of the virtual machine (VM) to an underloaded server. However, it's noteworthy that their method overlooks application delay constraints and the potential for offloading to the cloud. An approach distributing the incoming task requests between the fog and cloud was presented by Deng and collaborators [12]. Their approach proposes to minimize only network latency and power consumption. The primary goal of the strategy presented in [13] was to reduce completion time, considering both processing time and transmission time, but not encompass resource heterogeneity. Fan et al. [14] worked to minimize service latency and introduced an approach to inspecting the effect of overloading of the VM in terms of processing time. However, the approach fails to consider resource heterogeneity. Many studies have taken place based on latency sensitivity for task offloading Mahmud and co-authors [15] put forth a latency-aware policy designed to fulfill specified deadlines for task offloading by

improving application characteristics as well as resource utilization for edge devices. However, it's crucial to note that their work does not tackle the challenge of resource heterogeneity. In [16], the researchers developed a priority-based service placement policy with improved application characteristics but failed to take into consideration heterogeneous edge devices for IoT applications. Considering latency sensitive applications Sonmez et al. [17] put forth an approach for offloading the incoming task request by employing fuzzy logic. However, it is noteworthy that this approach does not take into consideration resource heterogeneity.

Also, much research has been done which focus on resource utilization. Nan et al. [18] focus on reducing the cost and increasing the utilization of the resources at the edge by proposing an algorithm to offload the task, though it does not have a good impact on latency sensitive applications. Xu and co-authors [19] introduced a model to increase resource utilization, reduce service latency, and associated costs, but the model does not delve into the aspect of resource heterogeneity. Li and Wang [20] present a placement approach with the objective of decreasing energy consumption at edge nodes and maximizing resource utilization. However, it's worth noting that their work does not improve computation, communication, or delay-sensitivity.

Considering another QoS parameter as resource heterogeneity, considering heterogenous virtual machines and scheduling heavy tasks to powerful VMs, an algorithm was implemented by Scoca and coauthors [21] while ignoring resource utilization, which impacted the service time performance. Roy et al. [22] propose a task allocation strategy taking into account resource heterogeneity by reducing execution latency and balancing the load across edge nodes while impacting communication time While in [23], tasks are offloaded to the edge servers based on their task requirements, such as CPU, bandwidth, etc., to enhance application service time.

After a literature review, we found a few research gaps, which prompted us to propose an approach for offloading incoming taskrequests from edge devices in edge-cloud environment to improve QoS parameters in healthcare by taking into consideration the balancing of workload among the resources in the network. The next section focuses on the fuzzy logic system for the proposed approach.

4. Fuzzy Based Approach for the Proposed System

In situations where developing precise mathematical models is challenging, fuzzy logic proves to be a valuable tool. Compared to other decision-making algorithms, the use of fuzzy logic is more significant as it is having lower computational complexity [24]. Particularities of edge and cloud computing are taken into account while making judgments using fuzzy logic. The offloading ratio based on fuzzy logic was formulated in [24], considering factors like link delay and signal-to-noise ratio. A mobile edge orchestrator (MEO) can process the data by using network information and associating it with requirements obtained from applications [31].

4.1 Fuzzy-Based Approaches Considering Task Offloading and Load Balancing

The edge devices are resource constrained in terms of battery life and/or could not have enough capacity to process the requested computation. In a fuzzy-based approach, the activity is initiated by a specific edge node, which initiates communication with other neighbouring nodes to offload the request. This fuzzy-based MEO decides a suitable edge node from the information available of the incoming request. This selected edge device could be a local edge server, a neighboring edge server, or a cloud server [31].

Ramaswamy et al. [26] and Mao et al. [27] propose a load balancing approach that focuses on communication and computational delays. This strategy centered on utilization entails making decisions regarding task offloading based on the server's usage level, choosing the machine with the lowest load for the offloading process. The primary goal is to efficiently use edge resources and achieve load balancing. However, it should be noted that this approach does not take into account task communication demand and sensitivity to application delays. In the research work of Flroes et al. [28], the approach decides to offload the incoming task to the edge devices or to the cloud, considering it to have lower service latency. But it fails to consider the utilization of the edge cloud devices, which eventually leads to overloading the virtual machine (VM) and increasing latency. Similarly, Snomes et al. [17] introduced a task offloading method through a fuzzy logic system, considering only homogenous resources, but in the real world, edge-cloud systems are comprised of heterogenous resources. resources. Nguyen et al. [24] also proposed an algorithm to enhance various QoS parameters by determining a suitable server to handle the incoming application task but failed to consider heterogeneous resources.

From the previous literature review, it is summarized as to decide the suitable server to execute the incoming task by not taking into consideration the load balancing of the edge resources and also working with only homogenous resources were identified as some major challenges in these existing approaches.

More literature reviews have been done for fuzzy logic-based techniques, taking into account these research gaps

in the related approaches where:

- *Objective*: Task Offloading Decision,
- *Input Parameters*: Basic fuzzy input (WAN bandwidth, task length, average VM utilization on the edge server, delay sensitivity of task, MAN delay, local edge VM utilization, remote edge VM utilization.)
- *Target Server*: mobile device, local edge, neighbor edge, cloud

In comparison to existing fuzzy-based MEO approaches, Sonmez et al. [17] did not furnish precise details. Nevertheless, their methodology showcases notable efficiency concerning resource utilization and response time, even in the face of the difficulty posed by unevenly distributed workloads among edge servers. It is essential to note that their approach does not take into consideration the task migration strategy. There is no specific information available for Hossain et al. [29]. Nevertheless, their method outperforms by efficiently managing a larger computational workload within the MEC system and reducing reliance on the remote cloud. However, it encounters challenges associated with unevenly distributed workloads among edge servers and does not incorporate a task migration strategy. Nguyen et al. [30] employ a worst-fit algorithm to address the bottlenecks in multi-tier edge computing architectures. This solution focuses on crucial factors such as WLAN delay, MAN delay, as well as local and neighboring VM utilization [31]. However, challenges arise from unbalanced loads among edge servers, and the approach does not take into account a task migration strategy [31]. Tran TrongKhanh[31] addresses a simple optimal problem with a focus on load balancing. During instances of system overload, the suggested method not only reduces the proportion of failed tasks but also guarantees the shortest processing time, proving particularly advantageous for healthcare and augmented reality (AR) applications. However, it's crucial to highlight that this method does not account for a task migration strategy.

Load balancing is essential to ensure that each cloud node receives an equal share of the extra dynamic local workload in order to achieve good user satisfaction and resource utilization ratios. It also ensures the efficient and fair allocation of each computing resource. In addition to minimizing energy consumption, load balancing aims to increase customer happiness, maximize resource usage, considerably improve cloud system performance, reduce reaction time, and lessen job rejections.

4.2 Process of Fuzzy Logic System

The proposed framework divides the system into three layers or tiers: the software tier, the edge computing layer, which includes the edge orchestrator, various edge servers, etc., and IoT devices (user devices) [5]. Every edge computing node at the periphery functions as a micro datacenter with a virtualized environment [5]. Positioned in proximity to the Wi-Fi access point or the connected IoT devices of the base station, each edge node is supervised by the Edge Orchestrator, who manages its computing resources and application services.

When IoT devices opt for remote processing in edge-cloud environments, the execution of offloading tasks becomes feasible. IoT applications can employ their associated edge nodes to transmit their offloadable tasks to the Edge-Cloud system for execution [24]. Guided by decisions from the Edge Orchestrator (EO), the connected edge has the capability to independently process IoT tasks or collaborate with other edge nodes or the cloud as deemed appropriate [5].

The suggested architecture is simply one potential realization among various architectures outlined in the existing literature, including those in [20, 32, 33]. The added edge computing layer with intelligence is the primary distinction in the proposed architecture. This layer is in charge of managing and allocating edge node offloading jobs. In this layer, EO communicates with other architectural elements to learn about the availability and usage of system resources, the number of IoT devices, the tasks performed by their applications, and the locations of IoT tasks. EO decides and selects the best server to allocate the work based on information available from edge devices, which includes the number of connected devices, task length, task duration, etc. [24]

The Edge orchestrator employs a fuzzy logic system in the proposed architecture at the edge computing layer with intelligence to select and assign jobs from Edge devices to one of the edge servers while taking into account various QoS factors [5].To enhance a fuzzy inference system's ability to identify a target server for an incoming application task, we propose introducing new input variables and decisions. Four basic stages of a fuzzy logic system (FLS) include the fuzzification step, fuzzy rules, inference engine, and defuzzification, as shown in Figure 5.

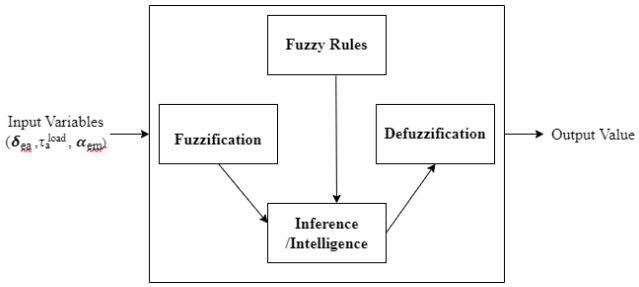

Fig 5. Fuzzy Logic System for the Proposed Approach

The basic steps of FLS are as mentioned below:

- **Fuzzy Input Variables:** The input variables designated for the fuzzy algorithm are known as fuzzy input variables.Task length, CPU cycle

count, and service load time are the necessary input variables for the proposed system. Each of these variables is represented using a linguistic variable: High, Medium, and Low.

F1 = $\{\boldsymbol{\delta}_{\text{ea}}, \tau_a^{load}, \boldsymbol{\alpha}_{\text{em}}\}$

where $\{\boldsymbol{\delta}_{\text{ea}}, \tau_a^{load}, \boldsymbol{\alpha}_{\text{em}}\}$ are task length, number of CPU cycles, service load time respectively.

- **Membership function [24]**: The triangular version of the membership function, which is used to quantify the linguistic word for each fuzzy variable, is the one we employ the most frequently in the proposed approach.
- **Fuzzification step [24]:** During the fuzzification step, the crisp value undergoes transformation into a fuzzy value through the utilization of these membership functions.
- **Fuzzy Rules [24]:** A set of fuzzy rules that are similar to human thinking constitutes a fuzzy rule base. A fuzzy rule is referred to as IF-AND-THEN rule containing a condition and a conclusion. In computational experiments, the authors systematically vary the fuzzy rule set to identify the significantly superior one through empirical discovery. The best combination of rules, determined through these experiments, is then utilized to establish the fuzzy rules [30]. For the proposed approach, three linguistic variables with four membership functions are used, which results in n = 3^4 = 81 fuzzy rules.
- **Defuzzification [24]**: For the proposed system,centroid defuzzifier is used toconvert fuzzy rule output to a particular value.

4.3 System Model and Problem Formulation

The major goal of the proposed framework is to increase the QoS parameters by applying problem formulation method below.

The main focus of the presented approach is to facilitate task offloading with lower service time and increase the efficiency of edge devices.

The author in [34] describes various evaluation metrics and their significance in the healthcare domain:

- Latency/Response Time: The term "latency" denotes delays that commonly occur when one system component awaits the completion of a task by another component. Essentially, it represents the duration the processor requires to process the request.In the healthcare setting, delays are not acceptable because they could result in accidents in life-threatening circumstances.
- Propagation latency [35]: The data delivered to the edge server has a propagation delay, which is in turn affected by the physical path and the corresponding link congestion that exists between the user edge device and the destination edge server.
- Processing latency [35]**:**In addition to traffic, processing delays typically involve a queuing delay that is based on the volume of user service requests. Alternatively put, the total amount of time a request is in the network

We model our proposed system as:

The suggested system model is defined using notations where ‘E’ signifies the set of edge devices (EDs) and ‘e’ denotes individual devices. ‘A’ represents the set of applications (APPs), each falling into a known collection, with ‘a’ representing diverse application types[36]. ‘T’ represents the network time slot, and ‘Tn’ denotes the current time slot at time n. ‘Tn−1’ signifies the previous network time slot, while ‘Tn+1’ corresponds to the subsequent time slot[36] . The tasks submitted by device 'e' in each network slot are denoted by ‘Ae’.

In the proposed architecture, the edge computing services layer possesses intelligence. It receives incoming tasks from various edge devices. Different edge servers connected to the network are assigned these tasks uniformly. For the proposed system, the set of edge servers is represented as ‘M’, individual edge servers are represented as ‘m’ with overall bandwidth denoted ‘$\boldsymbol{\beta}_{\text{m}}$’ and ‘$\beta_m^{idle}$’ as available bandwidth, ‘$\boldsymbol{\alpha}_{\text{m}}$’as overall computing resources and ‘α_m^{idle}’ represented as available computing resources [36]. Using the fuzzy logic approach, the proposed system decides the selected edge server, which computes the incoming tasks from the end devices and generates the output by improving the QoS parameters.

4.4 User Requested QoS Parameters

The proposed scheme considers various QoS factors and focuses on achieving user-specified objectives. These objectives primarily revolve around user-requested QoS parameters such as response time, data delay, computing delay, cost, and execution time. For achieving the above objectives, the proposed system, the incoming task requests from edge devices are assigned to edge servers, which are in proximity to the end devices with less workload.

4.4.1 Transmission Delay

Transmission of an incoming task request from anedge device to an edge server, considering the connection establishment between both the ends. Transmission delay considers both data and link delay represented as ‘τ_{em}^{link}’ between edge device ‘e’ and edge server ‘m’. It is the duty of EO to update and maintain this delay. The data delay can be considered as the amount of data, ‘$\boldsymbol{\delta}$ea’, transmitted and processed in the task once it is allocated to the edge server with allocated bandwidth ‘$\boldsymbol{\beta}$em’ is

calculated as follows in Equation (1):

$$\tau_{em}^{dd} = \boldsymbol{\delta}_{ea}/\boldsymbol{\beta}_{em} \quad (1)$$

Only link delay is considered in the data transfer on the way back, while data delay is ignored as it typically transmits the results. The total transmission delay for a task is expressed as in Equation (2):

$$\tau_{em}^{tot} = 2 * \tau_{em}^{link} + (\boldsymbol{\delta}_{ea} / \boldsymbol{\beta}_{em}) \quad (2)$$

4.4.2 Computing Delay

The duration needed to complete a task is determined by the computational capacity ('$\boldsymbol{\theta}_{ea}$')required for one unit of data from device 'e' in the present time slot. The overall computational capacity during task processing is represented as in Equation (3):

$$C = \boldsymbol{\theta}_{ea} * \boldsymbol{\delta}_{ea} \quad (3)$$

The computing delay is expressed as in Equation (4):

$$\tau_{em}^{comp} = \tau_{a}^{load} + [(\boldsymbol{\theta}_{ea} * \boldsymbol{\delta}_{ea})/\boldsymbol{\alpha}_{em}] \quad (4)$$

The minimum system cost, MinCost, is defined as the minimum sum of the total transmission delay and computing delay in Equation (5):

$$\text{MinCost} = \min(\tau_{em}^{tot} + \tau_{em}^{comp}) \quad (5)$$

Considering the following constraints:

$0 \leq \boldsymbol{\beta}_{m} \leq \beta_{m}^{idle}$(the bandwidth of the assigned task should be less than the unused bandwidth capacity of the server.)

$0 \leq \boldsymbol{\alpha}_{em} \leq \alpha_{m}^{idle}$ (the computational capacity of the assigned task on the edge server should be less than the rest of the computational capacity of the server.)

$(\tau_{em}^{tot} + \tau_{em}^{comp}) \leq \tau_{ea}^{md}$ (the overall delay for each task should be less than the highest permissible time limit.)

By using the above-formulated parameters and system model to address the performance issues, we plan to design algorithms using a fuzzy-based approach. for selecting the target edge server to offload the incoming task requests while taking into consideration workload balancing, reliability and scalability. For implementation and generating the results of the various metrics such as service latency, average processing delay, average VM utilization, etc., EdgeCloudSim simulator will be used.

5. Conclusion

Processing the data at the edge with intelligence rather than in the cloud is the aim of this research work. In the healthcare scenario, for achieving this aim, the proposed architecture has a service layer with edge intelligence to process the incoming task requests from the edge devices on the edge servers by enhancing the performance considering challenges like high latency, bandwidth, privacy, etc. Various existing algorithms have also been studied to design algorithms for the proposed framework using fuzzy logic systems at the edge computing level with intelligence. This paper presents an approach where a fuzzy edge orchestrator in an edge cloud system makes a decision whether to offload the incoming task to the local edge, neighbouring edge, or cloud server. By incorporating fuzzy logic and taking into account application characteristics such as CPU demand, network demand, and delay sensitivity, a system model and problem formulation have been developed. This approach significantly enhances Quality of Service (QoS) parameters.

As future work, we intend to design and implement the algorithms for the presented approach to validate the effectiveness of the proposed architecture as well as generating the simulation results.

Author contributions

Brinda Parekh initiated the research topic and actively participated in the design and implementation of the modelling system. **Kiran Amin** provided helpful insights and suggestions on various aspects of writing the paper. Both the authors read and approved the final version of the article.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Parekh, Brinda, and Kiran Amin. "Edge Intelligence: A Robust Reinforcement of Edge Computing and Artificial Intelligence." In Innovations in Information and Communication Technologies (IICT-2020) Proceedings of International Conference on ICRIHE-2020, Delhi, India: IICT-2020, pp. 461-468. Springer International Publishing, 2021.

[2] Xu, Dianlei, Tong Li, Yong Li, Xiang Su, SasuTarkoma, Tao Jiang, Jon Crowcroft, and Pan Hui. "Edge intelligence: Architectures, challenges, and applications." arXiv preprint arXiv:2003.12172 (2020).

[3] Shi, Weisong, Jie Cao, Quan Zhang, Youhuizi Li, and LanyuXu. "Edge computing: Vision and challenges." IEEE internet of things journal 3, no. 5 (2016): 637-646.

[4] Parekh, Brinda, and Kiran Amin. "A Proposed Architecture For Resolving Performance Issues In Edge Intelligence." In 2021 International Conference on Communication information and Computing Technology (ICCICT), pp. 1-5.

IEEE, 2021.

[5] Almutairi, Jaber, and Mohammad Aldossary. "A novel approach for IoT tasks offloading in edge-cloud environments." Journal of Cloud Computing 10, no. 1 (2021): 1-19.

[6] Lyu, Xinchen, HuiTian, Li Jiang, Alexey Vinel, SabitaMaharjan, Stein Gjessing, and Yan Zhang. "Selective offloading in mobile edge computing for the green internet of things." IEEE network 32, no. 1 (2018): 54-60.

[7] Dinh, ThinhQuang, Jianhua Tang, QuangDuy La, and Tony QS Quek. "Offloading in mobile edge computing: Task allocation and computational frequency scaling." IEEE Transactions on Communications 65, no. 8 (2017): 3571-3584.

[8] Flores, Huber, Xiang Su, VassilisKostakos, Aaron Yi Ding, PetteriNurmi, SasuTarkoma, Pan Hui, and Yong Li. "Large-scale offloading in the Internet of Things." In 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 479-484. IEEE, 2017.

[9] Samie, Farzad, VasileiosTsoutsouras, Lars Bauer, Sotirios Xydis, DimitriosSoudris, and Jörg Henkel. "Computation offloading and resource allocation for low-power IoT edge devices." In 2016 IEEE 3rd world forum on internet of things (WF-IoT), pp. 7-12. IEEE, 2016.

[10] Wang, Shiqiang, MurtazaZafer, and Kin K. Leung. "Online placement of multi-component applications in edge computing environments." IEEE Access 5 (2017): 2514-2533.

[11] Rodrigues, Tiago Gama, KatsuyaSuto, Hiroki Nishiyama, and Nei Kato. "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control." IEEE Transactions on Computers 66, no. 5 (2016): 810-819.

[12] Deng, Ruilong, Rongxing Lu, Chengzhe Lai, Tom H. Luan, and Hao Liang. "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption." IEEE internet of things journal 3, no. 6 (2016): 1171-1181.

[13] Zeng, Deze, Lin Gu, Song Guo, Zixue Cheng, and Shui Yu. "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system." IEEE Transactions on Computers 65, no. 12 (2016): 3702-3712.

[14] Fan, Qiang, and Nirwan Ansari. "Application aware workload allocation for edge computing-based IoT." IEEE Internet of Things Journal 5, no. 3 (2018): 2146-2153.

[15] Mahmud, Redowan, KotagiriRamamohanarao, and RajkumarBuyya. "Latency-aware application module management for fog computing environments." ACM Transactions on Internet Technology (TOIT) 19, no. 1 (2018): 1-21.

[16] Hassan, Hiwa Omer, SadoonAzizi, and Mohammad Shojafar. "Priority, network and energy-aware placement of IoT-based application services in fog-cloud environments." IET communications 14, no. 13 (2020): 2117-2129.

[17] Sonmez, Cagatay, AtayOzgovde, and CemErsoy. "Fuzzy workload orchestration for edge computing." IEEE Transactions on Network and Service Management 16, no. 2 (2019): 769-782.

[18] Nan, Yucen, Wei Li, Wei Bao, Flavia C. Delicato, Paulo F. Pires, and Albert Y. Zomaya. "Cost-effective processing for delay-sensitive applications in cloud of things systems." In 2016 IEEE 15th international symposium on network computing and applications (NCA), pp. 162-169. IEEE, 2016.

[19] Xu, Jinlai, BalajiPalanisamy, Heiko Ludwig, and Qingyang Wang. "Zenith: Utility-aware resource allocation for edge computing." In 2017 IEEE international conference on edge computing (EDGE), pp. 47-54. IEEE, 2017.

[20] Li, Yuanzhe, and Shangguang Wang. "An energy-aware edge server placement algorithm in mobile edge computing." In 2018 IEEE International conference on edge computing (EDGE), pp. 66-73. IEEE, 2018.

[21] Scoca, Vincenzo, Atakan Aral, IvonaBrandic, Rocco De Nicola, and Rafael BrundoUriarte. "Scheduling latency-sensitive applications in edge computing." (2018): 158-168.

[22] Roy, DeepsubhraGuha, Debashis De, Anwesha Mukherjee, and RajkumarBuyya. "Application-aware cloudlet selection for computation offloading in multi-cloudlet environment." The Journal of Supercomputing 73 (2017): 1672-1690.

[23] Taneja, Mohit, 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), pp. 1222-1228. IEEE, 2017.

[24] Nguyen, VanDung, Tran TrongKhanh, Tri DT Nguyen, ChoongSeon Hong, and Eui-Nam Huh. "Flexible computation offloading in a fuzzy-based mobile edge orchestrator for IoT applications." Journal of Cloud Computing 9, no. 1 (2020): 1-18.

[25] Duan, Qiang, Shangguang Wang, and Nirwan Ansari. "Convergence of networking and cloud/edge computing: Status, challenges, and opportunities." IEEE Network 34, no. 6 (2020): 148-155.

[26] Ramaswamy, Lakshmish, ArunIyengar, and Jianxia Chen. "Cooperative data placement and replication in edge cache networks." In 2006 International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 1-9. IEEE, 2006.

[27] Mao, Li, Yin Li, GaofengPeng, XiyaoXu, and Weiwei Lin. "A multi-resource task scheduling algorithm for energy-performance trade-offs in green clouds." Sustainable Computing: Informatics and Systems 19 (2018): 233-241.

[28] Flores, Huber, Xiang Su, VassilisKostakos, Aaron Yi Ding, PetteriNurmi, SasuTarkoma, Pan Hui, and Yong Li. "Large-scale offloading in the Internet of Things." In 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 479-484. IEEE, 2017.

[29] Hossain, MdDelowar, Tangina Sultana, VanDung Nguyen, Waqasur Rahman, Tri DT Nguyen, Luan NT Huynh, and Eui-Nam Huh. "Fuzzy based collaborative task offloading scheme in the densely deployed small-cell networks with multi-access edge computing." Applied Sciences 10, no. 9 (2020): 3115.

[30] Nguyen, VanDung, Tran TrongKhanh, Tri DT Nguyen, ChoongSeon Hong, and Eui-Nam Huh. "Flexible computation offloading in a fuzzy-based mobile edge orchestrator for IoT applications." Journal of Cloud Computing 9, no. 1 (2020): 1-18.

[31] Khanh, Tran Trong, VanDung Nguyen, and Eui-Nam Huh. "Fuzzy-based mobile edge orchestrators in heterogeneous IoT environments: An online workload balancing approach." Wireless Communications and Mobile Computing 2021 (2021): 1-19.

[32] Vaquero, Luis M., and Luis Rodero-Merino. "Finding your way in the fog: Towards a comprehensive definition of fog computing." ACM SIGCOMM computer communication Review 44, no. 5 (2014): 27-32.

[33] Bonomi, Flavio, Rodolfo Milito, Jiang Zhu, and SateeshAddepalli. "Fog computing and its role in the internet of things." In Proceedings of the first edition of the MCC workshop on Mobile cloud computing, pp. 13-16. 2012.

[34] Khattak, Hasan Ali, Hafsa Arshad, Ghufran Ahmed, SohailJabbar, AbdullahiMohamud Sharif, and Shehzad Khalid. "Utilization and load balancing in fog servers for health applications." EURASIP Journal on Wireless Communications and Networking 2019, no. 1 (2019): 1-12.

[35] Mendel, Jerry M. "Fuzzy logic systems for engineering: a tutorial." Proceedings of the IEEE 83, no. 3 (1995): 345-377.

[36] Qin, Zhenquan, Zanping Cheng, Chuan Lin, Zhaoyi Lu, and Lei Wang. "Optimal workload allocation for edge computing network using application prediction." Wireless Communications and Mobile Computing 2021, no. 1 (2021): 5520455.Ghosh,