Advances in Industrial Engineering, December 2024, 58(2): 439-456
DOI: 10.22059/aie.2024.379026.1899

RESEARCH PAPER

Performance Oriented Scheduling and Allocation Technique in Edge-Fog-Cloud Collaborative Environment

Fatemeh Ghayoor[1], Donya Rahmani[2]*, Amir Hossein Jafari Pozveh[3]

[1]*Ph.D. Candidate, Department of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran.*
[2]*Associate Professor, Department of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran.*
[3]*Ph.D., Department of electrical Engineering, Iran university of science and technology, Mobile Communication Company of Iran (MCI), Tehran, Iran.*

Received: 6 July 2024, Revised: 31 August 2024, Accepted: 1 September 2024

Abstract

Nowadays, smart devices are becoming more prevalent in industrialized countries generating requests that require computational processing. Recently, collaborative edge-fog-cloud computing networks have been developed to allocate users' requests to computing resources. However, scheduling these requests while accounting for user requirements and limited resources remains a challenge. This study proposes a structured planning at multiple levels in a collaborative edge-fog-cloud environment to allocate and schedule requests, aiming to reduce network latency. So, an Integer Programming (IP) formulation is developed to minimize network latency for users. Some network limitations are considered in the model, such as network logic for directing requests to computational resources, meeting deadline and nodes capacity constraints. Additionally, constraints related to processing allowable workload volume are integrated into the model. This strategy changes the workload distribution among the edge, fog, and cloud layers to approximately 24%, 27%, and 47%, respectively, creating a more balanced workload distribution and reducing workload traffic. Other results indicate that simply increasing the computational capacity of the fog nodes does not always improve network performance. This suggests the need for a more analytical approach, considering additional factors simultaneously in the underlying network. These outcomes underscore the efficiency and practical significance of the proposed model in a collaborative edge-fog-cloud computing landscape. The findings can help cloud service enterprises in providing efficient services for addressing the request scheduling and allocation challenges in edge-fog-cloud networks.

Keywords:
Edge-Fog-Cloud Network, Request Scheduling, Offloading, Allocation, Deadline.

Introduction

In the past decade, cloud computing has emerged as a popular approach for request processing, encompassing different areas from web applications to artificial intelligence [1]. In traditional cloud computing environments, cloud servers are usually located far from the end-user. In these environments, user requests are sent over the network to cloud servers for processing. Cloud

* Corresponding author: (Donya Rahmani)
Email: drahmani@kntu.ac.ir

computing resources are typically incorporate in data centers, where user requests are processed. Although centralized cloud servers can meet the needs of some latency-sensitive requests, they often fall short in meeting the requirements of latency-critical requests. Consequently, offloading request to the cloud can result in substantial latency [2]. On the other hand, as seen in Fig.1, global data is expected to grow to 181 zettabytes by 2025 [3]. So, the increasing computational workload in clouds leads to intolerable network latency. With the growth computational workloads by users, many cloud service providers have started offering edge and fog services [4].

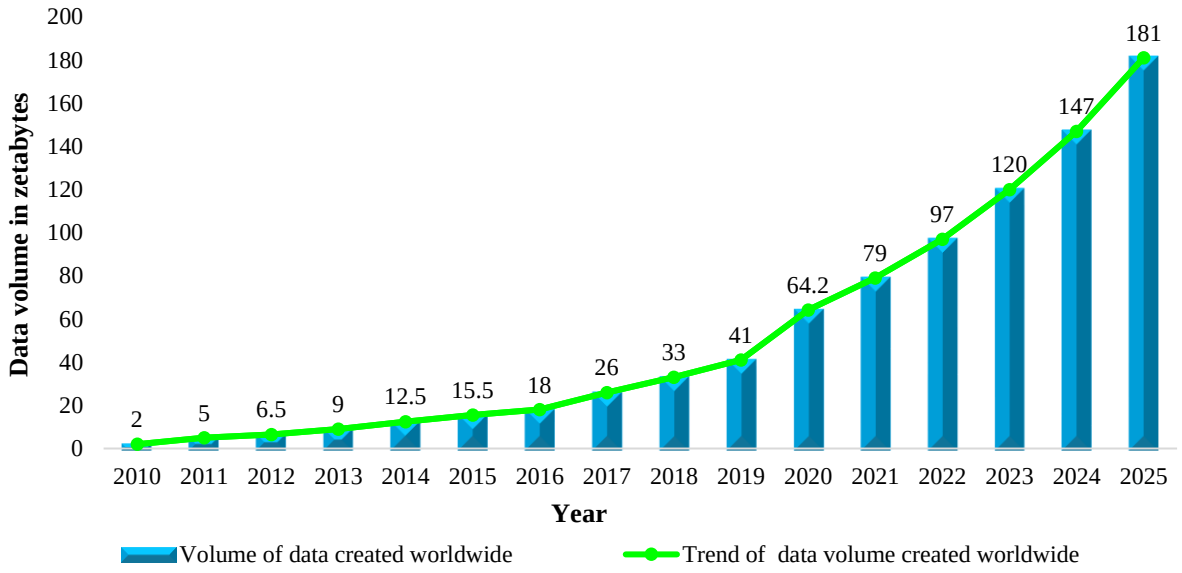

Fig. 1. Data volume growth

Therefore, edge and fog computing has emerged to represent like-cloud service, especially for running latency-sensitive workloads. However, their computational resources are limited and cannot quickly meet all high-volume computational needs [4]. So, these nodes alone cannot fully meet all user requests and Quality of Service (QoS) requirements. So, to achieve balanced computational requests in a computational network, collaboration between edge, fog nodes and the cloud servers is crucial. Fig.2 illustrates the general structure of edge-fog-cloud networks.

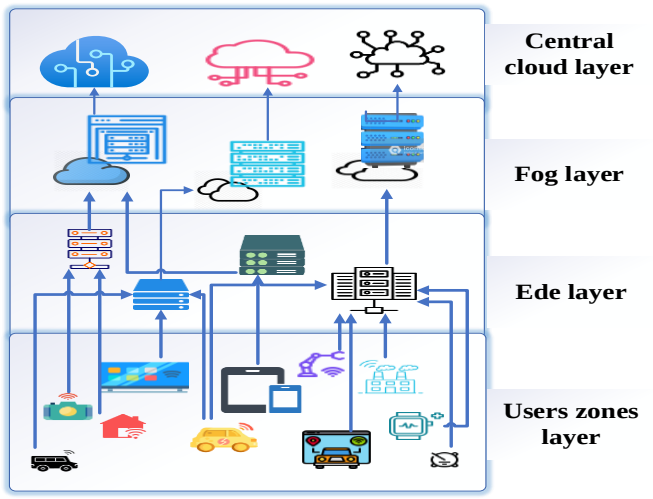

Fig. 2. General framework for edge-fog-cloud network framework

This hierarchical approach helps manage workloads effectively, ensuring requests are processed at the most suitable layer based on current network conditions and server capacities

[5]. Also, due to the challenges and requirements in Service Level Agreement (SLA) relating to edge-fog-cloud networks, efficient scheduling is necessary to balance the workloads across the network. Thus, a fundamental problem in edge-fog-cloud environments is how to schedule and allocate user requests among edge, fog and cloud servers to minimize latency for user requests [6].

Literature Review

Scheduling algorithms for request offloading in edge-fog-cloud computing play a crucial role in optimizing resource utilization, minimizing latency and enhancing overall system performance. It aims to balance workload across the network to prevent overloading of any specific node. So, in [7], authors proposed a novel algorithm for managing request queues and making offloading decisions in a computing network. It focused on optimizing the location where requests are offloaded based on queue scheduling strategies. In [8], an industrial cloud system supported by edge computing has been investigated. Authors, proposed a static scheduling strategy to aid dynamic scheduling, introducing queue theory to calculate job completion time. In [9], a scheduling and bandwidth planning with priority-based request scheduling algorithm was proposed to arrange request allocation, maximizing profit and handling requests. Initially, requests were processed by the nearest fog server's priority scheduling unit, with some reallocated to other fog servers if resources are insufficient. In [10], network dynamic scheduling was proposed to improve resource allocation for users. Authors classified requests into three categories based on their size and incorporated requests to minimize energy consumption and response time. In [11], the objective of resource scheduling was to select the best resource from a pool of available options. This study was evaluated using scheduling algorithms like Min-Min and Round Robin, based on metrics such as average waiting time, average response time, and makespan.

In edge-fog-cloud networks, deadline refers to the latest acceptable time by which a specific request must be completed. This concept is crucial because many requests, particularly in real-time and latency-sensitive environments, require to be processed within a strict time-frame to ensure proper functionality and service quality. Deadline management algorithms optimize request allocation to minimize latency and ensure timely completion. In [12], authors focused on leveraging distributed cloud resources. This study explored strategies for executing user requests in real-time while considering requests deadlines. In [13], researchers implemented a two-stage scheduling model designed to handle heterogeneous requests effectively. The model used deep reinforcement learning techniques in the scheduling process to minimize request completion time and meet stringent deadlines.

latency minimization refers to the strategic efforts and technological approaches aimed at reducing latency in request transmission and processing. Latency, in this context, is the time it takes request to travel from its source to its destination and back. Lower latency means that users experience quicker responses to their requests, which is especially important for real-time requests like online gaming, video conferencing, and augmented/virtual reality. In [2] authors, addressed the challenge of minimizing latency and energy consumption in a computing network by proposing a workload allocation method that guarantees minimal latency. In [14], researchers presented a dynamic approach using Integer Linear Programming to optimize request offloading from IoT devices to fog layer. By considering time constraints and resource availability, the method minimizes latency at fog nodes. In [15], the main goal was defined as minimizing computational resource consumption and communication latency. Initially, the problem has been restricted to edge servers, and an approximate algorithm has been used to solve it. Subsequently, both edge and cloud servers were considered and an enhanced algorithm was used to solve the problem. In [16], researchers introduced a scheme reducing latency by

classifying requests based on their resource requirements before distributing them among edge nodes by clustering techniques.

Solution approaches in the context of edge-fog-cloud networks, refer to the methodologies, techniques and frameworks used to address challenges and optimize various aspects of computing, networking and service delivery. In [17], researchers formulated request offloading as a stochastic optimization problem aimed at minimizing energy consumption and average queue length in mobile edge servers. In [18], authors introduced a flexible scheduling approach using zero-one programming to optimize request completion and energy consumption in job shop scheduling scenarios. In [19] researchers explored resource scheduling optimization for edge-cloud collaboration, introducing a multi-data center resource optimization method based on a graph data structure. In [20], the model with multi-objective of resource allocation method has been introduced. It formulated as a NP-hard problem by considering factors such as network sustainability, path contention, network delay, and cost-efficiency. In [21], stochastic nature of computation request arrivals, service times and dynamic computation resources have been addressed. Researchers model the edge-fog-cloud network as a two-stage tandem queue with multiple servers. Researchers in [22], enhanced an existing ILP problem by incorporating an ILP heuristic for multi-workflow allocation and utilized Particle Swarm Optimization (PSO) and Deep Reinforcement Learning (DRL) algorithms. Authors in [23], defined an optimization problem aimed at maximizing the worst utility across all services, taking into account constraints with various resource types. In [24] authors concentrated on a collaborative computing framework between nodes, with optimization of power control, scheduling and offloading decisions among mobile devices (MDs) and edge servers to minimize overall energy consumption, taking user mobility into account. The issue was formulated as a Mixed-Integer Programming (MIP) optimization problem to find the optimal solution. In [25], authors introduced a many-objective resource scheduling model aimed at optimizing edge cloud performance by considering factors such as request duration, cost, load balance, user satisfaction, and trust.

Summary of studied works has been shown in Table.1.

Based on Table.1 a comparative analysis of previous works in the field of edge-fog-cloud networks typically involves evaluating different approaches and methodologies used to tackle challenges. Some authors utilize Meta-Heuristic-Based methods used for their simplicity and speed in providing near-optimal solutions that suitable for real-time requests due to lower computational complexity [24,25,26,27]. However, these methods may not always guarantee optimal solutions and performance can vary depending on the problem's complexity. Some authors provide more precise solutions by formulating the problem as a mathematical model. Integer Linear Programming (ILP), Mixed-Integer Linear Programming (MILP), and other exact optimization techniques can give optimal or near-optimal solutions [17,20,22]. Some works presented Machine Learning-Based approaches that capable of learning from data, adapting to changes, and making predictions [22,13]. They can handle complex, dynamic environments by capturing patterns in the data. However, these methods require large datasets for training and may involve high computational costs. About using metrics, common metrics applied more, contain latency, energy consumption, cost, resource utilization, and balanced workload distribution. In addition, Network Constraints include deadline and QoS requirements, network logic and routing and nodes capacity [15,16,17].

Given the studies, this paper presents a user request scheduling and allocation problem in a collaborative edge-fog-cloud environment. The main objective is to develop and propose a structured multi-level scheduling and allocation strategy. The aim is to minimize network latency for users. Faster responses to requests enable users to complete more requests in less time, which is beneficial in time-sensitive requests such as healthcare monitoring or emergency response systems. We develop our model by evenly distributing and managing computational

workloads across the different layers considering constraints such as deadline meeting, network logic and routing and nodes capacity constraints. We achieve this by formulating an Integer Linear Programming (ILP) model.

Table.1. Review of the Literature

Author	Performance Index	Solution Technique	Time Estimation in scheduling			Main Idea
			Processing Time	Offloading Time	End Time	
Jin et al (2024) [26]	Latency	Whale Optimization Algorithm	✓	✓	-	Request offloading with intelligent network services
	Request partitioning					Improvement in response time
	Network traffic					Reduction in request execution time
Khaleel et al (2024) [27]	Latency	Moth Flame Optimization Algorithm	-	-	✓	Classifying requests into sensitive and non-sensitive categories
	Balanced distribution					
Ji et al (2023) [28]	Energy consumption	Particle Swarm Optimization Algorithm	✓	-	✓	Optimization of four parameters for each mobile device
	Network security					Enhancement of network security
Lin et al (2021) [29]	Request execution cost	Fuzzy Discrete Particle Swarm Optimization	✓	✓	✓	Cost-based scheduling in a non-deterministic computing environment
	Request deadline					
Abulizi et al (2022) [30]	Request partitioning	Simulation	-	-	-	Reducing network latency
	Latency					Using network control mechanisms to coordinate distributed environments
Ji et et al (2023) [31]	Latency	Simulation	✓	✓	-	Heterogeneous resources and request dependencies
						Algorithms combining minimum cut and maximum benefit
Zheng et al (2022) [5]	Workload balancing	Workload balancing	✓	✓	-	Proposing a deep learning-based approach for request scheduling aimed at reducing latency and resource consumption
	Incomplete request rate					
Li et al (2022) [32]	Request deadline	Lyapunov optimization	-	-	-	A scheduling algorithm for dependent request with limited deadline constraints under both online and offline scenarios
	Request execution cost	Simulation				
Current Work (2024)	Latency	Integer Linear Programming Model	✓	✓	✓	Proposing a new mathematical modeling approach for the requests scheduling and allocating in edge-fog-cloud environment
	Request deadline	Using GAMS software and CPLEX solver				Estimating end time of requests
						Balancing network using allowable workload volume constraints

Therefore, the main contributions of the paper can be followed as:

- Proposing a new Integer Linear Programming (ILP) model: The research proposes a novel approach using ILP to optimize the scheduling and allocation of user's requests within an edge-fog-cloud network. The proposed scheduling attempts to model many network limitations, including request offloading between layers, computational capacity of nodes and meeting deadlines.
- Estimating end time of requests: The study aims to estimate the end time for each request as it is progressed. This is particularly important in our dynamic environment where user workload fluctuate during time-steps.
- Optimal allocation of user requests: The proposed model tries to achieve efficient request allocation across different layers enhancing overall network performance and reducing latency by directing requests to the most appropriate computational resources.
- Workload balancing in the network: The presented programming aims to mitigate network congestion at the fog layer by providing a method to evenly distribute user workloads across the network. This approach helps optimize resource utilization by ensuring that no layer of the network is overwhelmed with excessive traffic.

The following sections are structured as follows: next section describes the presented request scheduling and allocation problem and then explains its formulation. Next, we present a demonstrate case with real-world data. Then, we provide a detailed examination of the experimental results and numerical analysis. Finally, the conclusion addresses the implications of our findings and offers suggestions for future research directions.

Problem Statement

In this study, an integrated computing framework that includes users of intelligent devices, an edge computing layer, a fog computing layer and a cloud computing layer is offered. The user layer is full of requests from automated smart devices generating data. For simplification, users are divided into some geographical locations and any location is called user zone. In this approach, requests from users in each zone are collected at the edge layer and are accumulated in these nodes. At the beginning of each time-step, these cumulative requests are then ready for allocation and processing at the edge, fog or cloud layer. The edge computing layer is positioned above the user layer. This layer consists of edge nodes, any of them efficiently placed close to each user zone to ensure fast data processing and offloading. The goal of the problem is to minimize requests latency in processing requests and offloading them to other layers. By latency minimization, users spend less time waiting for requests to be processed, which can significantly enhance productivity, particularly in industrial and enterprise environments where delays can slow down operations. When requests come in edge node, the system analyzes what each request needs, checks the capacity of the edge nodes and decides whether to process the accumulative requests locally or offload to the fog layer. After the edge network layer, the fog layer is located at a greater distance from the users' zones. However, it has stronger computational capacity compared to the edge nodes and is used to enhance network efficiency due to the limited capacity of edge nodes. Both cloud computing and fog computing provide request processing for end users. However, fog computing is closer to the user zone leading to lower latency and has a wider geographical distribution. High above is the cloud computing layer, filled with powerful remote servers. It is crucial to determine when and where requests must be executed across edge, fog and cloud resources. The programming ensures that the offloading and processing of the requests be processed within a specific time frame, that is, a predetermined deadline.

Each user request consumes a portion of node processing capacity, reducing the available resources for other requests. Therefore, efficient resource allocation is crucial for processing requests within an acceptable timeframe. A detailed Integer Linear Programming model is presented to optimize this process, distributing requests across layers based on request needs, nodes capabilities, and network conditions. By minimizing latency, the system ensures timely request completion, enhancing overall performance and user satisfaction.

The hierarchical structure of the studied network is described as follows:

- **User Layer:** This layer includes industrial devices such as Automated Guided Vehicles (AGVs), industrial robots, tablets, IoT devices, smart appliances, and sensors. Users are categorized into several zones and each user can only exchange data with edge nodes.
- **Edge Computing Layer:** This layer consists of edge nodes positioned between the user zones and fog layers, typically within the intranet for sufficient bandwidth. Edge nodes, located near user zones, process or offload requests to the fog layer. Processing on edge nodes is called local processing.
- **Fog Computing Layer:** Positioned closer to the edge nodes, fog nodes decide whether to process requests regionally or offload them to the cloud, considering network latency, deadlines, and capacity.
- **Cloud Computing Layer:** Comprising remote servers with higher computational speed,

cloud servers handle requests offloaded from fog layer, known as central processing.

To estimate the end time of requests, it is essential to take into account both processing and offloading time. Edge layer latency contains processing time on the edge nodes. Since the edge nodes are located close to the user zones, the latency associated with offloading requests from the user zone to the edge can be disregarded. Fog layer latency considers processing time and any offloading time required to offload request from edge to the fog nodes. Cloud layer latency accounts for processing time in the cloud data centers and offloading from edge to fog nodes and then to cloud data center.

Because we have set a short-term goal (minimizing latency) and we want to determine the request scheduling and allocation to achieve this goal in a very tiny time-frame (120 seconds), our planning is considered as operational planning. Therefore, key aspects of represented operational planning include:

1. **Resource Allocation:** Determining what computational resources are needed in any time-step.
2. **Scheduling:** Meeting deadlines for requests and estimating the end time of requests.
3. **Process Management:** Designing and optimizing processes to achieve the best node for processing any request.
4. **Performance Monitoring:** Setting a key performance indicator (KPI) (latency minimization) for user to measure performance network.
5. **Load balancing:** Distributing requests workload volume evenly in order to decreasing layers traffic.

Minimizing latency in edge-fog-cloud networks is crucial for quick responses in requests like autonomous vehicles and smart cities. Latency, the delay between a request generation and response, must be reduced by optimizing where and how requests are processed, ensuring the fastest possible response time leading to more user satisfaction.

An illustration of the studied problem is shown in Fig.3.

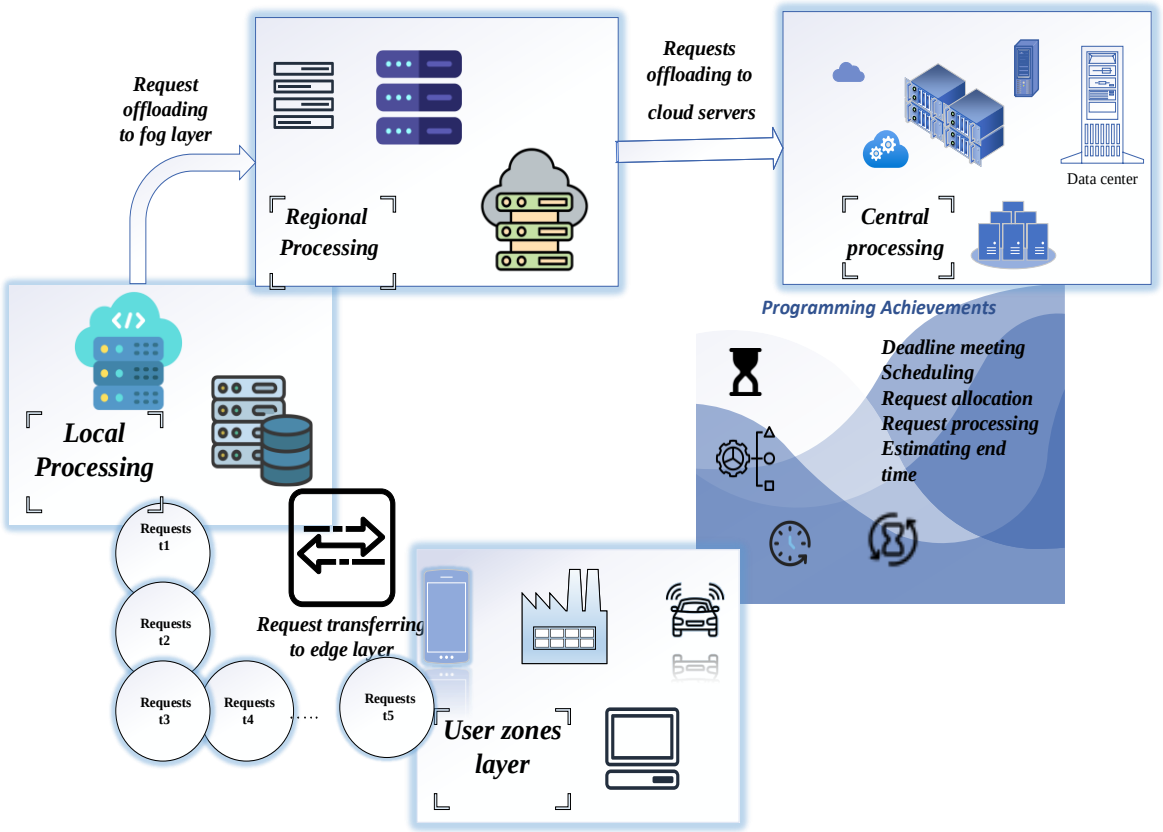

Fig. 3. General outline of the edge-fog-cloud network under study

Problem Formulation

Main model

In this section, we first describe the Integer Linear Programming for edge-fog-cloud network and then introduce demonstrative case for evaluating the model performance. The symbols are presented in Table. A in appendix.

Objective Function

The main goal of the problem is to minimize end time of accumulative requests processed locally, regionally, or centrally leading to latency minimization. Therefore, the objective function is formulated like follows:

$$Z = Min\left(Max\ \left(Et_z^t, Et_{z,e}^t, Et_{z,c}^t\right)\right) \qquad \forall z, e, c, t \qquad (1)$$

$$s_{z,e,c}^{t} = Max\ \left(Et_z^t, Et_{z,e}^t, Et_{z,c}^t\right) \qquad \forall z, e, c, t \qquad (2)$$

$$s_{z,e,c}^{t} \geq Et_z^t \qquad \forall z, e, c, t \qquad (3)$$

$$s_{z,e,c}^{t} \geq Et_{z,e}^t \qquad \forall z, e, c, t \qquad (4)$$

$$s_{z,e,c}^{t} \geq Et_{z,c}^t \qquad \forall z, e, c, t \qquad (5)$$

As seen the objective function in Eq.1 is nonlinear. Thus, for linearization Eq.2 to Eq.5 are added to the model. These constraints illustrate that if the variable $s_{z,e,c}^{t}$, is larger than each of the end times for edge, fog and cloud processing in any time-step, maximizing it leads to end time minimization for all requests.

Offloading and Processing Constraints

There are certain rules in the network that show the permission to offload requests or process them in different layers. These rules relating to the general logic of the network and request routing, and if they are not followed, the logic of the model will be wrong. These rules are as follows:

$$x_z^t + x_z^{e,t} + x_z^{c,t} = 1 \qquad \forall z, e, c, t \qquad (6)$$

Eq. (6) states that each cumulative request must be processed either on the edge, fog nodes or on the cloud servers.

$$\sum_{e} y_{z,e,c}^{t} \leq 1 \qquad \forall z, c, t \qquad (7)$$

Eq. (7) stipulates that if the offloading occurs from edge z to the cloud c, this cumulative request can only be offloaded from one fog node e and cannot be transferred from other fog nodes.

$$\sum_{c} y_{z,e,c}^{t} \leq 1 \qquad \forall z, e, t \qquad (8)$$

In the same way, Eq. (8) shows that if the transfer of cumulative request from fog e to the cloud c occurs, this request can only be offloaded to one cloud server and cannot be offloaded to other clouds.

$$\sum_{e} y_{z,e}^{t} \leq 1 \qquad \forall z, t \qquad (9)$$

Eq. (9) stipulates that cumulative request from edge z can only be offloaded to one fog node e.

$$1 - M(1 - \sum_{c} y_{z,e,c}^{t}) \leq y_{z,e}^{t} \leq 1 + M(1 - \sum_{c} y_{z,e,c}^{t}) \qquad \forall z, e, t \qquad (10)$$

Eq. (10) indicates that if a cumulative request from fog e is offloaded to the cloud c, it first must be transferred to an edge node.

$$1 - M\left(1 - \sum_{e} y_{z,e,c}^{t}\right) \leq x_{z}^{c,t} \leq M\left(1 - \sum_{e} y_{z,e,c}^{t}\right) + 1 \qquad \forall z, c, e, t \qquad (11)$$

$$-M(\sum_{e} y_{z,e,c}^{t}) \leq x_{z}^{c,t} \leq M(\sum_{e} y_{z,e,c}^{t}) \qquad \forall z, c, e, t \qquad (12)$$

Eq. (11) shows that if cumulative request is processed in the cloud, it must be offloaded to the cloud by a fog node. Additionally, Eq. (12) indicates that if a request is not transferred to the cloud via any fog node, it can not be processed in the cloud.

$$-M(1 - \mathrm{x}_{\mathrm{z}}^{\mathrm{t}}) \leq \sum_{e} \mathrm{y}_{\mathrm{z},e}^{\mathrm{t}} \leq M(1 - \mathrm{x}_{\mathrm{z}}^{\mathrm{t}}) \qquad \forall \mathrm{z}, \mathrm{t} \qquad (13)$$

Eq. (13) states that if local processing (on edge nodes) is performed, offloading to the fog nodes is not allowed.

$$x_{z}^{e,t} \leq y_{z,e}^{t} \qquad \forall z, c, e, t \qquad (14)$$

Eq. (14) specifies that if the request is offloaded from edge layer to the fog nodes, the request can either be processed at the fog node or transferred to the cloud.

$$-M\,(1 - x_{z}^{e,t}) \leq \sum_{c} y_{z,e,c}^{t} \leq M\,(1 - x_{z}^{e,t}) \qquad \forall z, e, t \qquad (15)$$

Eq. (15) discusses that if processing is processed at any fog server, transfer to the cloud is not possible.

$$-M\,(1 - x_{z}^{t}) \leq \sum_{e} y_{z,e}^{t} \leq M\,(1 - x_{z}^{t}) \qquad \forall z, e, t \qquad (16)$$

Eq. (16) indicates that if local processing occurs, request offloading to any fog server is not possible.

$$1 - M\,(1 - x_{z}^{t}) \leq \sum_{e} y_{z,e}^{t} \leq 1 + M\,(1 - x_{z}^{t}) \qquad \forall z, e, t \qquad (17)$$

Eq. (17) states that if there is no local processing, the accumulative request must be offloaded to one of the fog nodes to process or offload to the cloud from there.

End Time Constraints

By these constraints, the end time of each request is estimated. This estimation process is crucial for effective workload management and resource allocation within the network.

$$et_{z}^{t} = {(t\,x_{z}^{t} + \tau\,w_{z}^{t}\,x_{z}^{t})}/{c_{z}} \qquad \forall z, t \qquad (18)$$

Eq. (18) indicates the end time of the cumulative request processed locally. It is assumed that as soon as the request is generated, it can be processed locally and then some time is also needed for processing on the edge layer.

$$et^{t}_{z,e} = x^{e,t}_{z}((t + \tau w^{t}_{z})/c_{e} + (d^{e}_{z}/\mu + \delta w^{t}_{z} x^{e,t}_{z})/b_{z,e}) \qquad \forall z, e, t \qquad (19)$$

Eq. (19) describes the end time of the cumulative requests processed at the fog node. This time includes cumulative request generation time, processing time and request offloading time from edge to fog nodes.

$$et^{t}_{z,e,c} = x^{c,t}_{z}((t + \tau w^{t}_{z})/c_{c} + (d^{e}_{z}/\mu + \delta w^{t}_{z} x^{e,t}_{z})/b_{z,e}) + (d^{c}_{e}/\mu + \delta w^{t}_{z}) y^{t}_{z,e,m}/b_{e,c} \qquad \forall z, c, e, t \qquad (20)$$

Eq. (20) outlines the end time of the cumulative requests processed in the cloud servers. This time includes request generation time, processing time in the cloud data center, offloading time from the edge to the fog node and sequentially from the fog node to the cloud server.

Node Capacity Constraints

Node capacity constraints refer to the limitations on the computational resources that each node in a network can provide. These constraints can be as follows:

$$w^{t}_{z} x^{t}_{z} \leq c_{z}/CF \qquad \forall z, t \qquad (21)$$

$$\sum_{z} w^{t}_{z} x^{e,t}_{z} \leq c_{e}/CF \qquad \forall e, , t \qquad (22)$$

Eq. (21) and Eq. (22) indicate that the total processed cumulative requests in the edge and fog nodes at each time-step must be less than their computational capacity. c_z and c_e are computational power for processing in edge and fog nodes respectively, however since their unit is based on MIPs, a correction factor e.g., CF is needed to convert it based on MB.

Deadline Constraints

Deadline constraints refer to the specific time limits within each request must be completed. Adhering to deadline constraints ensures that requests are completed within the required time-frame, preventing latency that could affect overall system performance. The deadline constraints are as below:

$$et^{t}_{z} \leq tx^{t}_{z} + d_{z} \qquad \forall z, t \qquad (23)$$

$$et^{t}_{z,e} \leq tx^{e,t}_{z} + d_{e} \qquad \forall z, e, t \qquad (24)$$

$$et^{t}_{z,e,c} \leq tx^{e,t}_{z} + d_{c} \qquad \forall z, e, c, t \qquad (25)$$

Eq. (23) to Eq. (25) indicate that if a request must be processed locally, regionally or centrally, their end time must be before the predetermined deadline. These deadlines have shown using d_z, d_e, d_c parameters. Also, t in above constraints is time-step in scheduling.

$$et^{t}_{z,e,c}, et^{t}_{z,e}, et^{t}_{z}, s^{t}_{z,e,c} \geq 0 \qquad \forall z, c, e, t \qquad (26), (27), (28), (29)$$

$$y^{t}_{z,e}, y^{t}_{z,e,m}, x^{t}_{z}, x^{e,t}_{z}, x^{c,t}_{z} \in \{0,1\} \qquad \forall z, c, e, t \qquad (30)\ (31)\ (32)\ (33)\ (34)$$

Finally, Eq. (26) to Eq. (29) are continuous variables and Eq. (30) to Eq. (34) are binary

variables in the proposed model.\

Demonstrative Case

This section presents a numerical example using real-world data to evaluate the proposed model. Assumptions are as follows:

- Users with smart devices are divided into several zones, with cumulative requests collected on edge nodes at each time-step.
- Requests must be processed locally, regionally, or centrally.
- Requests for processing in any layer, have pre-defined deadlines.
- A 120-second planning horizon with stable time-steps is considered.
- All requests are processed within one second.
- The cloud has infinite capacity.

We summarize the parameters based on the real inputs provided like Table.2:

Table.2. Model input parameters

Parameter	Quantity	Unit
User zones/edge node	16	Zone/node
Fog nodes	3	Node
Cloud servers	2	Server
Bandwidth between fog and cloud	1500	Mbps
Bandwidth between edge and fog	1000	Mbps
Cumulative request workload	U (45, 65)	MB
Computational speed in edge nodes	U (10,000, 14,000)	MIPs
Computational speed in fog nodes	U (50,000, 55,000)	MIPs
Computational speed in cloud servers	U (90,000, 100,000)	MIPs

This setup enables rigorous evaluation and optimization of resource allocation and scheduling across edge-fog-cloud network. The model programming has been done in GAMS software, and the solver utilized is CPLEX. All experiments are conducted on a system with 16 GB of RAM and an Intel Core i7-6650 processor with a frequency of 2.2 GHz.

Numerical Results

In this section, the numerical results obtained from solving the model are presented in detail. The results are discussed and analyzed comprehensively to provide insights about the model's performance and implications.

Fig.4 is related to the cumulative requests from various users' zones generated at the 102nd second of the scheduling. It shows the total offloading and processing time of the requests generated and processed by different fog nodes. As seen, due to the computational speed in the fog nodes, the time required to complete the users' cumulative requests is less than one second highlighting the model's efficiency. Consistently lower latency in the network, contributes to higher service quality, leading to greater user satisfaction.

Fig.5 compares the percentage of cumulative requests processed across different layers, highlighting the system’s performance. Due to the limited computational capacity, only 1.2% of requests are processed locally at the edge layer. A small percentage is handled by the cloud, while 98.3% are processed on fog nodes, ensuring quick processing and minimizing latency by avoiding excessive offloading to the cloud. This comparison helps in understanding the overall performance and balancing of the system, and it underscores the importance of each layer in achieving network optimal operations.

Table.3 illustrates the impact of workload volume on network performance by showing the

objective function's value across two workload ranges. As seen, doubling the workload increases the objective function by 18%, reaching 284,975. This rise indicates that higher workloads strain fog node capacity, shifting more processing to the cloud by up to 24%. While the cloud handles the increased load, it introduces additional latency, adversely affecting the objective function.

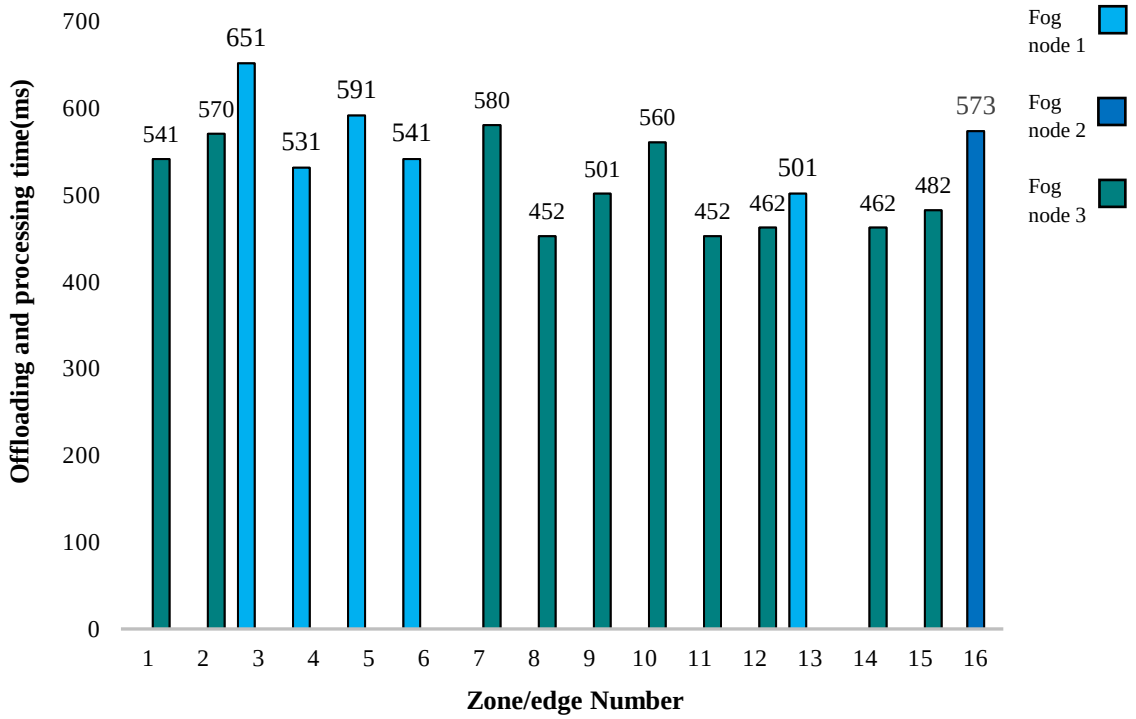

Fig. 4. Summation of offloading and processing time in fog nodes

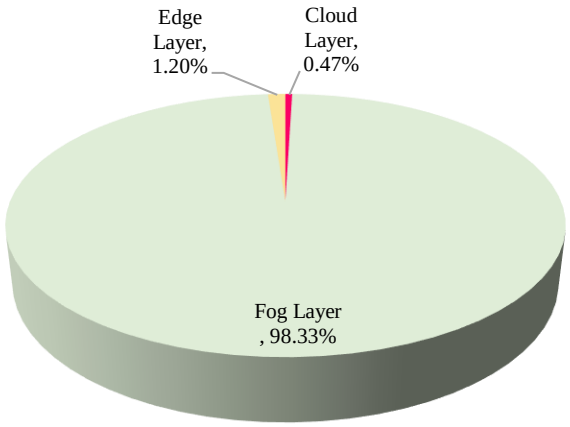

Fig. 5. The percentage of the processed requests in different layers

Table.3. Percentage of processed requests in different layers

Completed requests percentage			Objective function	Workload volume	Row
Central	Regional	Local			
0.47%	98.33%	1.2%	240,944	(45,65)	1
24%	69%	6%	284,975	(90,130)	2

Fig.5 shows that most workload processing occurs at the fog nodes, leading to network traffic imbalance. To address this, an alternative scheme can be implemented by introducing constraints that balance workload distribution across local, regional, and central processing based on each zone's workload volume.

Advanced Model: Considering Allowable Processing Workload Volume Constraints

In the "main model" section, programming excluded allowable workload volume constraints. However, this scheme adds a strategy requiring allowable processing workload to meet specific levels, necessitating the following constraints:

$$w_z^t \leq A + M(1 - x_z^t) \qquad \forall z, t \qquad (35)$$

Eq. (35) states that if the workload volume is less than a specified value "A", then the request must be processed locally.

$$a - M(1 - x_z^{e,t}) < w_z^t \leq b + M(1 - x_z^{e,t}) \qquad \forall z, e, t \qquad (36)$$

Eq. (36) specifies that if the workload volume falls between two specified values A and B, the cumulative request must be processed regionally.

$$B - M(1 - x_z^{c,t}) \leq w_z^t \qquad \forall z, t \qquad (37)$$

Lastly, Eq. (37) indicates that if the workload volume exceeds a specified value B, the request must be processed centrally. Also, Eq. (1) to Eq. (34) must be considered in the advanced model.

The numerical results, considering the addition of the above constraints to the model, are as follows. As observed, in Fig.6, the 25% processing at the local layer indicates that a significant portion of the requests is handled close to the data source, minimizing latency. The 28% processing at the regional layer shows that a moderate share of the workload is handled at an intermediary level. Finally, the 48% processing at the central layer suggests that the most requests are efficiently managed at the data center. These results illustrate more balanced workload distribution among the network layers in comparison with main model.

This balanced distribution is vital for system performance, ensuring overloading in each layer. By avoiding over-reliance on a single layer, the model handles varying workloads effectively while meeting network constraints, resulting in a more robust and adaptable system.

Table.4. Objective function value in different schemes

Model	Objective function value
Main model	240,944
Developed Model	409,203

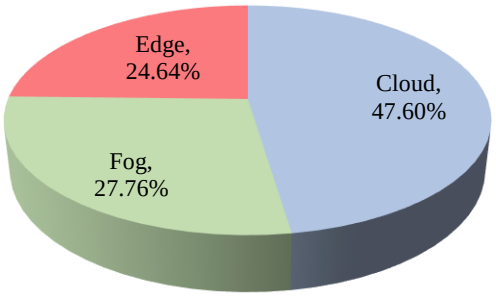

Fig. 6. Processed requests in layers under the developed model

According to Table.4 when there is no workload volume constraint (Main Model), the objective function value equals 240,944 seconds, and when the workload constraint is applied in the model (Advanced Model), the objective function increases by 70% to 409,203 seconds. Therefore, it is necessary, to have a balance between the objective function and the workload volume constraints in the planning strategy.

Fig.7 & Fig.8 show the offloading and processing times on the fog nodes 1 and 3 over 120 seconds, highlighting performance differences. 161 in the vertical axis indicates that requests from edge node 16 were offloaded and processed by fog node 1. As observed, required time in fog node 1's is notably longer than fog node 3's due to factors like computing speed, workload, and distance. Fog node 3's with higher computing speed allows to handle requests more quickly.

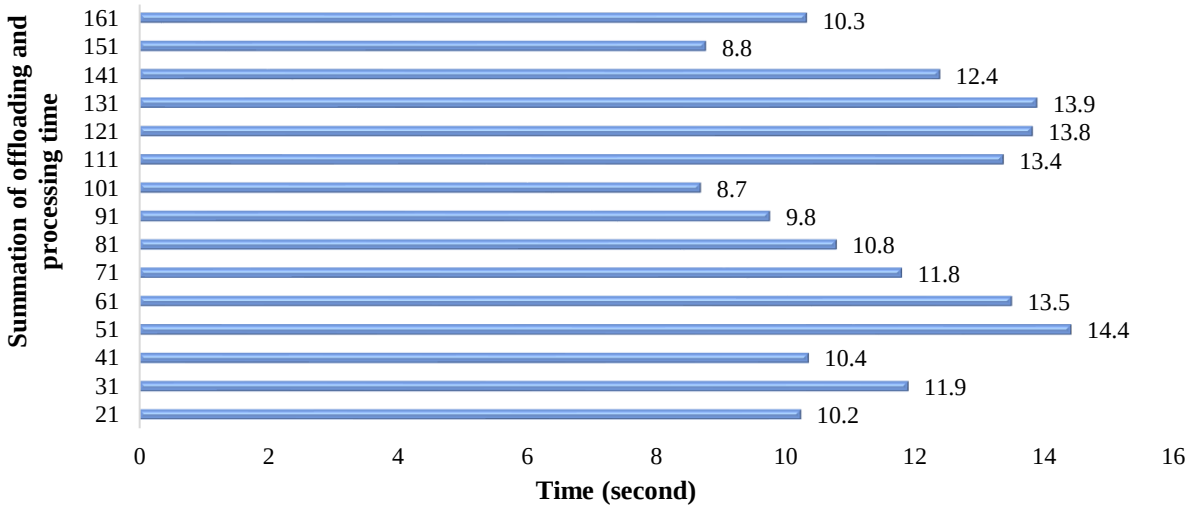

Fig. 7. Summation of offloading and processing time in fog node 1

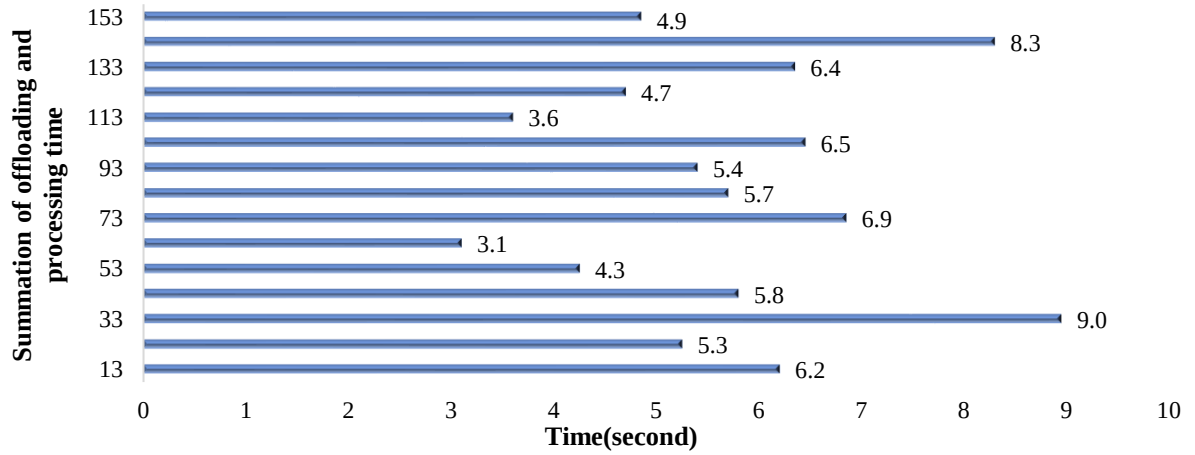

Fig. 8. Summation of offloading and processing time in fog node 3

To better understanding the performance of the fog nodes, Table.5 shows the completed requests percentage on different fog nodes. Additionally, server node 2 does not process any requests within the planned time horizon, which is due to its lower computational speed compared rather than two other nodes. As shown, fog nod 3 processes approximately twice as many requests as node 1, handling about 67% of the total processed requests in the fog layer. Therefore, node 3 is capable of processing a greater number of requests in a shorter amount of time. This implies that node 3 has higher efficiency and computational power compared to the other servers, making it more effective in handling heavy workloads within a limited time-frame.

Table.5. Processed requests in edge nodes

Percentage	Number of Processed Requests	Fog Node
32.8%	175	1
0%	0	2
67.2%	358	3

Considering that none of the cumulative requests are processed by fog node 2, a sensitivity analysis is conducted on the computational speed parameter of the fog node 2. Fig.9 relates to the behavior of the objective function versus to changes in the computational capacity of the fog node 2. As seen, increasing the fog computational speed does not always lead to better performance of the problem's objective function. For instance, in the range of (45,000,55,000) MIPs, increasing the computational speed of the node 2 does not result in any change to the objective function. This indicates that there is a threshold beyond which further increases in computational speed have no additional impact on the performance. In other words, the objective function only begins to decrease linearly when the speed of node 2 exceeds that of the largest fog node (which is fog node 3, with a computational speed of 55,000). Therefore, it is crucial to carefully examine the impact of increasing the computational speed of a node on the entire network. Blindly increasing it, without this consideration may not always lead to an improvement in the network performance but also result in unnecessary resource expenditure. This emphasizes the importance of comprehensive planning and analysis before making changes in network components to ensure optimal network performance.

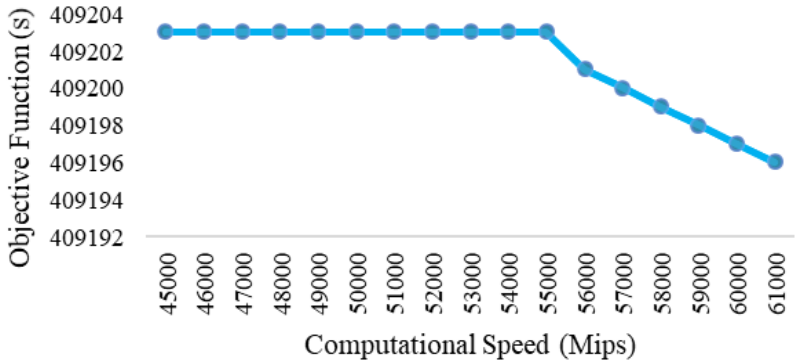

Fig.9. Changes in the objective function versus computational speed

Based on the detailed analysis of the model's performance, the following **managerial insights** can be derived:

Leveraging Edge/Fog Computing: Edge and fog nodes efficiently handle requests due to their proximity to users and high computational speed. Managers should enhance these capabilities to minimize latency and improve service delivery.

Optimized Workload Distribution: Overloading fog nodes is a risk. Implementing workload volume limitations across local, regional, and central layers can balance the processing load. Managers should ensure even workload distribution to prevent bottlenecks.

Monitoring and Scaling Resources: Workload increases significantly impact network performance. Managers must monitor trends and scale computational resources dynamically to handle demand effectively without over-relying on cloud processing.

Strategic Allocation of High-Performance Nodes: Deploying powerful nodes in high-demand areas can boost efficiency. Managers should strategically place these nodes where request volumes are highest to maximize network performance.

Cost-Benefit Analysis for Capacity Expansion: Increasing computational capacity doesn't always improve performance beyond a certain point. Managers should conduct cost-benefit analyses before investing in additional resources, prioritizing upgrades that offer tangible benefits.

Continuous Performance Monitoring: Regularly monitoring offloading and processing times helps to identify performance disparities, guiding resource allocation decisions.

Balancing Workload Constraints: Implementing workload limitations improves distribution but may increase latency. Managers should balance these elements with network performance goals.

Scenario Planning and Future Demand: Given processing variability, managers should plan for future demand fluctuations, establishing flexible resource allocation and capacity management strategies.

Also, the following managerial tips, inferenced from some works [33&34] are as follows:

Monitoring Performance Metrics: Managers should regularly track key performance indicators (KPIs) such as request latency, resource consumption, and overall system efficiency. This monitoring will help identify areas for improvement and ensure that the resource management model is functioning as intended.

Emphasizing Scalability: Managers should design their resource management strategies with scalability. As the number of terminal devices and requests increases, they should ensure that the system can handle the additional load without significant degradation in performance.

Adapting to Dynamic Changes: Managers should develop strategies to quickly adapt to changes in the environment, such as fluctuations in request workloads or network conditions. Implementing real-time monitoring and adaptive algorithms can help maintain optimal resource allocation even during unexpected scenarios.

Balancing Trade-offs: Managers should be aware of the trade-offs involved in optimizing request execution latency and resource consumption. They should evaluate the implications of these trade-offs on costs and latency.

Monitoring Resource Utilization: Managers should continuously monitor the utilization of edge, fog and cloud resources. This monitoring helps identify bottlenecks and underutilized resources, allowing for adjustments in service placement and orchestration strategies.

Therefore, the managerial insights deducted from the results of this article are consistent with those found in other articles. These insights emphasize critical aspects of the network such as resource allocation, latency management, cost optimization, and scalability.

Conclusion

With the increasing number of smart devices used by users, request offloading and processing is becoming a complex issue. Traditional cloud infrastructures may fail to meet requests latency requirements properly. Therefore, edge and fog nodes contribute to lower latency in network. Quicker responses to user allow them to accomplish more requests in a shorter period, which is advantageous in time-critical situations like healthcare monitoring or emergency response systems. Moreover, edge and fog nodes may not be able to respond to all user requests due to limited computational capacity. Therefore, this study focuses on a requests scheduling and allocation model in a collaborative edge-fog-cloud network to minimize network latency. An ILP formulation is developed to plan multi-layer network and then, an example with real inputs is examined and solved the model in GAMS software. The proposed programming ensures that request offloading and processing be completed in predefined deadlines. In main model the most proportion of the requests (98%) will be processed on the fog layer disrupting the balance of the network. So, in order to achieve the more balanced network, some constraints related to allowable processing workload volume are applied. In this developed model, processing on edge, fog and cloud will be modified to approximately 24%, 27%, and 47%, respectively, however; the objective function will be increased up to 70%. Therefore, judgement to have a balanced network with acceptable latency is crucial. The presented numerical results demonstrate the effectiveness and credibility of the mathematical modeling in the collaborative edge-fog-cloud network. By employing the proposed model, cloud services enterprises can optimize resource utilization, meet deadline, minimize latency and deliver reliable services to users. For future research, request splitting can also be considered. In other words, part of the user's request workload can be processed simultaneously across the three network layers within any time-step.

References

[1] B.Li, P.Hou, H.Wu and F.Hou, Optimal edge server deployment and allocation strategy in 5G ultra-dense networking environments. Pervasive and Mobile Computing; 72:101312.2021.
[2] A.Ali-Eldin, B.Wang, P.Shenoy. The hidden cost of the edge: a performance comparison of edge and cloud latencies. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-12, 2021.
[3] Ulrike, Hack. (2021), What's the real story behind the explosive growth of data?, www.red-gate.com
[4] N.Kherraf, S.Sanaa, M.Chadi and Ali Ghrayeb. "Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds." IEEE Transactions on Network and Service Management 16, no. 4, 1435-1449, 2019.
[5] Zheng T, Wan J, Zhang J, Jiang C. Deep reinforcement learning-based workload scheduling for edge computing. Journal of Cloud Computing. 2022;11(1):3.
[6] L.Liu, Z.Chang, X.Guo, T.Ristaniemi. Multi-objective optimization for computation offloading in mobile-edge computing. In2017 IEEE symposium on computers and communications (ISCC), pp. 832-837). IEEE, 2017.
[7] H.Birhanie, MO.Adem. Optimized request offloading strategy in IoT edge computing network. Journal of King Saud University-Computer and Information Sciences;36(2):101942, 2024.
[8] Y.Ma, Y.Han, J.Wang, Q.Zhao, A constrained static scheduling strategy in edge computing for industrial cloud systems. International Journal of Information Technologies and Systems Approach (IJITSA), 14(1):33-61, 2021.
[9] Apinaya Prethi KN, Sangeetha M, Nithya S. Optimized scheduling with prioritization to enhance network sustainability in edge-cloud environment. Journal of Intelligent & Fuzzy Systems. 2023;44(3):4323-34.
[10] Prabhu R, Rajesh S. An Advanced Dynamic Scheduling for Achieving Optimal Resource Allocation. Computer Systems Science & Engineering. 2023;44(1).
[11] Rahul S, Bhardwaj V. Optimization of Resource Scheduling and Allocation Algorithms. In2022 Second International Conference on Interdisciplinary Cyber Physical Systems (ICPS) 2022 May 9 (pp. 141-145). IEEE.
[12] Szalay, P.Mátray, L.Toka. Real-time request scheduling in a FaaS cloud. 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), pp. 497-507. IEEE, 2021.
[13] X.Zhou, W.Liang, K.Yan, W.Li, I.Kevin, K.Wang, J.Ma, Q.Jin. Edge-enabled two-stage scheduling based on deep reinforcement learning for internet of everything. IEEE Internet of Things Journal,10(4):3295-304, 2022.
[14] S.Bebortta, SS.Tripathy, UM.Modibbo, I.Ali. An optimal fog-cloud offloading framework for big data optimization in heterogeneous IoT networks. Decision Analytics Journal,8:100295, 2023.
[15] Y.Mao, X.Shang, Y.Yang. Joint resource management and flow scheduling for SFC deployment in hybrid edge-and-cloud network. InIEEE INFOCOM 2022-IEEE Conference on Computer Communications. 170-179, IEEE,2022.
[16] I.Ullah, HY.Youn,. Request classification and scheduling based on K-means clustering for edge computing. Wireless Personal Communications, 113(4):2611-24, 2020.
[17] CF.Liu, M.Bennis,HV.Poor HV, Latency and reliability-aware request offloading and resource allocation for mobile edge computing, 2017 IEEE Globecom Workshops (GC Wkshps) pp. 1-7. IEEE. 2017
[18] Q.Gao, GU.Fu, L.Li , J.Guo. A framework of cloud-edge collaborated digital twin for flexible job shop scheduling with conflict-free routing. Robotics and Computer-Integrated Manufacturing,86:102672, 2024.
[19] Wen S, Jia F, Lijun W, Hui L, Yu W, Linhui M. Research on resource scheduling optimization technology for power cloud edge collaboration. InSixth International Conference on Intelligent Computing, Communication, and Devices (ICCD 2023). 2023. (Vol. 12703, pp. 223-229). SPIE.
[20] Apinaya Prethi KN, Sangeetha M. A multi-objective optimization of resource management and minimum batch VM migration for prioritized request allocation in fog-edge-cloud computing. Journal of Intelligent & Fuzzy Systems. 2022;43(5):5985-95.
[21] Yang C, Xu H, Fan S, Cheng X, Liu M, Wang X. Efficient resource allocation policy for cloud edge end framework by reinforcement learning. In2022 IEEE 8th International Conference on Computer and Communications (ICCC) 2022 (pp. 1363-1367). IEEE.
[22] Buschmann P, Shorim MH, Helm M, Bröring A, Carle G. Request allocation in industrial edge networks with particle swarm optimization and deep reinforcement learning. In Proceedings of the 12th International Conference on the Internet of Things 2022 Nov 7 (pp. 239-247).
[23] Bi R, Peng T, Ren J, Fang X, Tan G. Joint service placement and computation scheduling in edge clouds. In2022 IEEE International Conference on Web Services (ICWS) 2022 Jul 10 (pp. 47-56). IEEE.
[24] Hua W, Liu P, Huang L. Energy-efficient resource allocation for heterogeneous edge-cloud computing. IEEE Internet of Things Journal. 2023 Jul 7.
[25] Zhang J, Ning Z, Ali RH, Waqas M, Tu S, Ahmad I. A many-objective ensemble optimization algorithm for the edge cloud resource scheduling problem. IEEE Transactions on Mobile Computing. 2023 Jan 9;23(2):1330-

46.
[26] Jin H, Ma S, Ding Y, Deng X, Yao Z, Zhao J. Edge cloud cooperation environment driven multi-level edge computing method for delay optimization. In Fourth International Conference on Machine Learning and Computer Application (ICMLCA 2023) 2024 May 22 (Vol. 13176, pp. 513-518). SPIE.
[27] Khaleel MI. A dynamic weight–assignment load balancing approach for workflow scheduling in edge-cloud computing using ameliorated moth flame and rock hyrax optimization algorithms. Future Generation Computer Systems. 2024 Jun 1;155: 465-85.
[28] Ji T, Wan X, Guan X, Zhu A, Ye F. Towards optimal application offloading in heterogeneous edge-cloud computing. IEEE Transactions on Computers. 2023 Jun 28.
[29] Lin B, Lin C, Chen X, Xiong NN, Shen Q. A Fuzzy Scheduling Strategy for Intelligent Workflow Decision Making in Uncertain Edge-Cloud Environments. 2021;1–15. Available from: http://arxiv.org/abs/2107.01405
[30] Abulizi J, Hu Q, Wang W. Delay Optimization of Power Internet of Things based on Edge-Cloud Collaboration. In: 2022 IEEE 22nd International Conference on Communication Technology (ICCT). 2022. p. 905–10.
[31] Ji T, Wan X, Guan X, Zhu A, Ye F. Towards optimal application offloading in heterogeneous edge-cloud computing. IEEE Transactions on Computers. 2023 Jun 28.
[32] Li Y, Dai W, Gan X, Jin H, Fu L, Ma H, Wang X. Cooperative service placement and scheduling in edge clouds: A deadline-driven approach. IEEE Transactions on Mobile Computing. 2021 Feb 23;21(10):3519-35.
[33] Aslanpour MS, Gill SS, Toosi AN. Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. Internet of Things [Internet]. 2020;12: 100273.
[34] Taleb T, Samdanis K, Mada B, Flinck H, Dutta S, Sabella D. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. IEEE Commun Surv Tutorials. 2017;19(3):1657–81.

Appendix A

Symbol	Description
t	Set of planning time-steps
z	Set of user's zones/ edges
C	Set of cloud servers
e	Set of fog nodes
M	Big M
τ	Correction factor for converting megabytes to operations quantity (MI/MB)
μ	Speed of electromagnetic waves (km/second)
δ	Correction factor for converting bytes to bits (bite/byte)
c_z	Computational power for edge processing (MIPS)
c_e	Computational power for fog processing (MIPS)
CF	Correction factor for converting number of instructions unit to MB unit
d_z , d_e , d_c	Deadline time for request processed locally, regionally and centrally
$b_{z,e}$	Bandwidth between edge/fog (Mbps)
$b_{e,c}$	Bandwidth between fog/cloud (Mbps)
$\mathrm{d}_z^{\mathrm{e}}$	Distance between edge/fog (km)
$\mathrm{d}_{\mathrm{e}}^{\mathrm{c}}$	Distance between fog/cloud (km)
w_z^{t}	Accumulative workload from zone z in time-step t (MB)
a	Upper bound for local processing (MB)
b	Upper bound for regional processing (MB)
$Et_{z,c}^{t}$	End time for accumulative requests on cloud node c (Second)
$Et_{z,e}^{t}$	End time for accumulative requests on fog node e (Second)
Et_z^t	End time for accumulative requests on edge z (Second)
$s_{z,e,c}^{\mathrm{t}}$	Maximum time between end times (Second)
$\mathrm{x}_z^{\mathrm{t}}$	If request processed locally, it takes 1, otherwise 0.
$\mathrm{x}_z^{\mathrm{e,t}}$	If request processed regionally, it takes 1, otherwise 0
$\mathrm{x}_z^{\mathrm{c,t}}$	If request processed centrally, it takes 1, otherwise 0
$\mathrm{y}_{z,e,c}^{\mathrm{t}}$	If request is offloaded to cloud c from fog e, it takes 1, otherwise 0.
$\mathrm{y}_{z,e}^{\mathrm{t}}$	If request is offloaded to fog e from edge e, it takes 1, otherwise 0.