IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 6 November 2024, accepted 14 November 2024, date of publication 20 November 2024,
date of current version 11 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3503058

RESEARCH ARTICLE

Resource Management Across Edge Server in Mobile Edge Computing

SAIFUR RAHMAN[1], MAZHAR HUSSAIN[2], SYED IBAD ALI SHAH[3], ZAIWAR ALI[2], MUHAMMAD AYAZ KHAN[4], GRZEGORZ NOWAKOWSKI[5], MAREK SIEJA[5], MUHAMMAD IRFAN[1], AND FAZAL MUHAMMAD[2]

[1]Electrical Engineering Department, College of Engineering, Najran University, Najran 61441, Saudi Arabia
[2]Faculty of Electrical Engineering, Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi 23640, Pakistan
[3]Faculty of Electrical Engineering, University of Engineering and Technology at Mardan, Mardan 23200, Pakistan
[4]Department of Engineering Management, University of Chester, CH1 4BJ Chester, U.K.
[5]Faculty of Electrical and Computer Engineering, Cracow University of Technology, 31-155 Cracow, Poland

Corresponding authors: Saifur Rahman (srrahman@nu.edu.sa) and Fazal Muhammad (fazal.muhammad@uetmardan.edu.pk)

The authors acknowledge the support from the Deanship of Graduate Studies and Scientific Research at Najran University. Kingdom of Saudi Arabia, for funding this work under the Easy Track Research funding program grant code number (NU/EFP/SERC/13/81).

ABSTRACT Mobile Edge Computing (MEC) has the potential to provide computational resources to edge devices. The Edge device (Ed) does not have enough computational resources, so it offloads its tasks to nearby Mobile Edge Servers (MES), while sometimes offloading tasks to MES causes MES to overload. The overload state of MES is when the data occupies all the CPU cores and memory of MES, so MES rejects requests for data offloading from other Eds. In this research, we have considered a multi-server, multi-device scenario, for which we have proposed an algorithm called CPU Memory Mobility Resource Allocation (CMMRA) that does not reject Ed from providing computational resources and encourages them to migrate the task to another MES in case the current MES is overloaded. The algorithm increases the service rate of a communication area, the rejection rate of the devices is also reduced, the total processing time of MES in this research is less as compared to others, and the migration in our proposed algorithm is greater than others. As we consider the mobility of devices and many devices are requesting computational resources, if the current MES is busy, it migrates the task to an underloaded MES for computation.

INDEX TERMS Mobile edge computing, resource allocation, mobile edge server, edge device.

I. INTRODUCTION

Mobile devices execute fewer tasks than traditional computers and are inherently limited by factors including minimal memory, computing power, and battery life. However, mobile applications, particularly those related to video gaming and graphics processing are becoming increasingly sophisticated every day [1], [2], [3]. Similarly, some applications such as video games, biomedical imaging data processing, speech recognition, and other applications need computing power that even the most advanced mobile phones cannot deliver [4], [5], [6].Offloading compute-intensive application processing to powerful remote servers is a solution to that problem. This idea is known as cloud computing, Large latencies, however, are caused by the distance between cloud servers and mobile devices. When using a mobile device, the transmission, execution, and reception times grow too long to provide a satisfactory user experience device [7].

A more realistic approach is to bring the network of cloud servers closer to the mobile device, to minimize the overall application execution delays.MEC is a new model that provides IT services and cloud computing to IoT devices within the radio access network, improving user experience by reducing network traffic and application delays [8].

As we know, a significant amount of data processing occurs at the network's edge in MEC [9],thus, moving the

The associate editor coordinating the review of this manuscript and approving it for publication was Irfan Ahmed.

computation away from the remote cloud data centers. Some edge devices are mobile devices, printers, sensors, desktops, PCs, tablets, and IoT devices that can be used for different purposes. These devices usually generate a large amount of data. MEC servers are located close to the edge devices and perform data processing, and the required data is transferred to the cloud for storage or processing. This helps in reducing the delay [10].The resources needed for processing the tasks in the edge servers can be physical resources like memory, CPU, or any network device or virtual resources like virtual machines (VM).

In a multi-device scenario, mobile devices request CPU cores and memory from the MES for computational tasks. When the MES allocates all its CPU cores to these tasks, it may become overloaded. In such cases, any new incoming requests for computational resources are redirected to other underloaded MESs to balance the load and prevent system overload [11]. Memory plays a vital role in assessing whether a MES is overloaded or underloaded, working alongside CPU core usage in this evaluation. It's crucial to manage these resources effectively to avoid pushing the MES into an overloaded condition. Ideally, device migrations to other MES should take place only when a server is completely overloaded, which helps maintain optimal resource distribution throughout the network. Additionally, migrations may occur if a device's position changes, requiring a handover. Both CPU cores and memory need to be closely monitored and factored into the evaluation of the MES's load status [12].

IoT workloads often require additional resources at MEC servers for processing, but limited hardware resources make resource contention an issue. To address this, we propose a distributed architecture where tasks are processed across MEC servers when memory and CPU cores are completely occupied, and further devices cannot be provided computational resources for processing, so, overloaded MES migrate devices request to nearby underloaded MES. This reduces time delay, increases service rate and will reduce the rejection rate of the Ed.

A. NOVELTY AND CONTRIBUTION

The novelty of this paper can be highlighted as follows:

- To consider the CPU cores and memory for determining the overloading state of MES.
- To calculate total computational time, the service rate, total number of migrations and total number of rejections in our proposed communication scenario.
- To proposed an algorithm which migrates the task to other MES in a local network when the MES become overloaded. The algorithm also rejects the Ed from providing computational resources when the MES are busy.
- To consider mobility of Ed in calculating the total number of migrations of the Ed in a local area network.

To the best of our knowledge, for considering the overloaded state of MES, no such model has been proposed previously for MEC resource allocation for Ed. We present our own algorithm that considers Ed requests and offloads to underloaded MES rather than waiting and uploading to overloaded MES.

The remainder of this paper is organized as follows. Section II presents a summary of the related work. section III describes the system model and the mathematical formulation of the proposed algorithm, section IV discuss different benchmark techniques, proposed algorithm and its limitations, section V provide the simulation results and discussion, and section VI concludes the paper.

II. RELATED WORK

The deployment of edge devices for IoT networks faces numerous challenges. The number of MEC servers required depends on network traffic and wireless heterogeneity in the area [13]. Resource management, data type comprehension, and determining whether to shift tasks to the cloud are the primary obstacles. It involves making prudent use of resources, understanding the data we are working with, and determining when it would be most effective to move some jobs to the cloud [14].Due to limited storage, computation capacity, battery power, mobility, energy management, and bandwidth allocation considerations, MEC presents several issues, such as resource management, data identification, and cloud offloading decision-making [10].

Researchers are using strategies from a field called game theory to help decide when to shift tasks from mobile devices to the cloud. This helps in managing tasks more efficiently [15].Researchers have found a new way to manage tasks in MEC. They are using something called game theory to decide when to move tasks from mobile devices to the cloud [16].The reinforcement learning technique has been used for offloading the data of the users for computation to surpass a given threshold [16].A new method has been suggested that considers delays in task graphs and chooses the best virtual machine. This method is used to lighten the load on IoT devices while maintaining the necessary service quality [17]. A strategy based on game theory has been suggested for transferring data in a MEC setting. In this scenario, the operator is responsible for managing the computing and wireless resources of the MEC servers [18].

Optimization of the system's utility has been regarded as the goal of the resource allocation and computation offloading challenge in MEC servers [18]. In [15] and [19] gaming models have also been used for the management of resources in the MEC servers. The work proposed in [20] explained a strategy for offloading the data by considering multiple parameters like computational complexity, radio resources, and security of data using the algorithm called the Advanced Encryption Standard (AES). A proposal was presented to use the Markov Decision Process as a decision-making technique to enhance the offloading time in MEC [21], [22].

A crucial area of cloud security dedicated to protecting cloud computing systems. As organizations increasingly rely

on cloud services for data storage and processing, concerns regarding data confidentiality and integrity arise, particularly when sensitive information is processed remotely. The review emphasizes the importance of data encryption as a key strategy for securing critical data, such as financial and medical information, which are particularly vulnerable to breaches. Despite the development of various methods aimed at enhancing data privacy during transmission, significant challenges remain in securing data that is stored in the cloud. This study comprehensively examines existing approaches to establishing data security in cloud environments, highlighting the need for continued research and innovation to address these ongoing vulnerabilities [23].

Research in Artificial Intelligence (AI) has enabled learning at the edge, which means making decisions based on the importance of data when allocating resources [24]. Different AI techniques, such as reinforcement learning [25], deep reinforcement learning, clustering [26] and federated learning, have also been used to optimize the resource allocation process [27].

CloudSec, a lightweight encryption method for securing medical images. Combining hashing, chaotic mapping, and genetic algorithms, CloudSec enhances data authentication and security by generating a unique key DNA image that disrupts pixel correlations. It offers fast encryption, a large key space, and strong resistance to attacks, making it suitable for real-time cloud computing applications and integration in IoHT frameworks for secure medical image transmission [28].

The Dynamic Energy-Efficient Offloading Algorithm (DEEO) for mobile devices in a Mobile Edge-Cloud Computing (MECC) environment, enhancing secure medical image transmission. DEEO enables devices to offload intensive tasks to nearby servers, reducing energy use, improving resource efficiency, and ensuring data security [29].

An analysis has been conducted on energy consumption-aware mobile edge computing, encompassing the current research on task framework computation offloading as well as the many approaches that have been explored in MEC [30].An energy-efficient algorithm has been proposed for the reduction in operation costs by excellent use of energy in Ed [31].Additionally, a type-classification and priority-based assignment method for energy-efficient compute offloading for smart devices has been developed [32].To minimize power consumption by offloading among all users while taking latency and power limits into account, a Sequential Convex Approximation Algorithm (SCA) has been suggested for energy-efficient conscious resource allocation in MEC [33].

The techniques for organizing and the management of location-based services in MEC have been proposed in [34].In MEC, virtual machines (VMs) were assigned according to user mobility, communication resources, and processing resources [35].

The linear regression technique [36] has been used to predict future CPU utilization. The lasso and ridge regression have been applied in multi-dimensional resource allocation for an auction-based application in a cloud computing environment [37]. The Root Mean Squared Error (RMSE) of the lasso and ridge regression was less when compared to linear regression. The utilization of CPU and storage was good when lasso and ridge regression were used in resource allocation [37].Several studies have used linear regression [36], multiple linear regression [37], and a hybrid approach based on ensemble empirical mode decomposition and Autoregressive Integrated Moving Average (ARIMA) to categorize the hosts as overloaded or underloaded [37]. VM migration was performed from overloaded hosts to underloaded hosts [36]. The CPU Utilization Prediction (LIRCUP) in Linear regression has reduced the power consumption and SLA violation [36].

In [11], an edge computing technique is presented to track the utilization of MES using a Dynamic Markov model for Resource Contention Prediction (DMRCP) model in Edge Cloud. When an offloaded task is anticipated to utilize all of its available CPU cores, the question determines whether it will overflow. This usage data was entered into a history matrix. The transition probability was then revised once again. If a given VM is identified by the model as creating overload on a particular server, the workload is moved to a different server.

Cloud computing makes it possible to provide a number of resources, including databases, storage, memory, CPU, and network bandwidth, in virtual machines (VMs). Allocating those VMs still presents a significant difficulty because it takes time. Dolphin Partner Optimisation (DPO) is therefore a security-enhancing algorithm that has been optimized [12]. Using energy and memory considerations, this approach selects virtual machines (VMs) and adds an extra layer of hypervisor protection. After that, the Dolphin Partner Optimization narrows down the options to get the top virtual machine (VM) in each group. A highly secure virtual machine is the outcome of applying simplified security methods to further improve security.

The previous studies only considered the utilization of CPU cores to determine the overloading state of MES. However, in our study, we consider both CPU cores and memory to accurately determine the overloading state of the MES as well as we considered the mobility of Ed. If the CPU and memory of an MES are fully utilized, any other devices requesting computational resources will be migrated to another nearby underloaded MES for providing computational resources. In addition, we have taken mobility into consideration in our work since it might cause migrations of devices between servers that depend on necessity and the right allocation of resources and maintenance of optimal performance.

III. SYSTEM MODEL

We consider a multi-user, multi-server scenario. The system model consists of several edge devices Ed = {1, 2, . . . ,n} and several mobile edge servers S = {1, 2, . . . ,n} in a local

TABLE 1. Symbols description.

Symbol	Description
B	Bandwidth
β	Path loss component
C	CPU clock cycles per byte
di	Distance between edge device and server
f_s	CPU rate of MES
g_{dl}	Bit error rate for downlink
g_{ul}	Bit error rate for uplink
h_{dl}	Channel fading downlink coefficient
h_{ul}	Channel fading uplink coefficient
M	Total number of CPU cores of MES
m	CPU cores allocated to edge device
N	Total number of sub-carriers
n	sub-carrier allocated to edge device
N_o	Noise power
P_s	MES transmission Power
P_u	Edge device transmission power
r_{dl}	Max achieveable downlink rate
r_{ul}	Max achieveable uplink rate
W	Workload of MES
S_r	Service rate
N_{rs}	Num of Ed provided computational services
N_{rT}	Total number of edge device requests

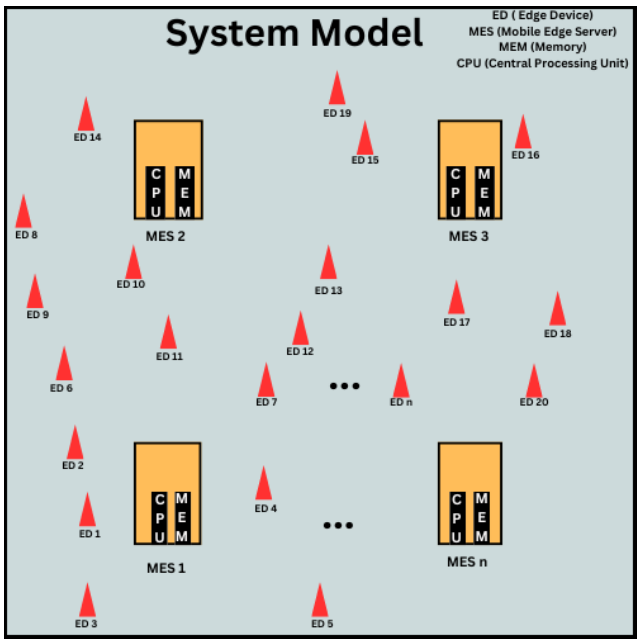

FIGURE 1. System model.

area network as shown figure 1. Initially, the servers are idle, but whenever a device sends a request for the computation of a task, one of the servers gets activated and will serve the request of the edge device. The server will serve the requests of edge devices until it becomes overloaded, and all its CPU cores and memory are utilized completely. In that case, it will not serve the requests of other devices for computation and will migrate the task to another server to perform the computation of the task assigned to it. If all the servers in a local area network are overloaded, the servers will reject the incoming request for computation services. The devices are dynamic as they are changing their positions.

The distance between Ed and MES is represented in the form of distance matrix.

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{bmatrix} \tag{1}$$

A. COMMUNICATION MODEL

The edge device will request the resources, and if its request is accepted, then the device will upload its data to the server. We will calculate the uploading time, and the server will process the task, or the amount of data uploaded to it. We will calculate the processing time, and then the data will be downloaded to the Ed. We will now calculate the downloading time. After this, we will calculate the total amount of data processed by all the servers in a local area network, and we will calculate the total time that the MES takes to process the data.

The Ed can upload a task to MES for execution. We consider that the network deployment utilizes the orthogonal frequency division multiple access (OFDMA), then we can assume that the Bandwidth B for transmission is divided into N number of sub-carrriers, the Ed will request for the available sub-carriers, if the sub-carriers are available then the device will be allowed to upload as well as to download a task. "n" is the number of subcarriers that are asssigned to an edge device. M is the total number of CPU cores of a MES, m is the number of CPU cores used in execution of task. If m = 0 it means the MES is busy, and all the CPU cores are assigned for the execution of tasks, we also consider the memory in a MES, if space or memory is available Ed data is stored in a MES, the data will be executed as soon as the CPU cores are available, the Ed is not rejected from providing computational services by the MES.

For transmission process, the maximum achievable uplink and downlink data rates for an additive white Gaussian noise (AWGN) can easily be derived as in [38].

$$\mathbf{r_{ul}} = n\frac{B}{N}\log_2\left(1 + \frac{P_u|h_{ul}|^2}{\Gamma(g_{ul})di^{\beta}N_o}\right) \tag{2}$$

$$\mathbf{r_{dl}} = n\frac{B}{N}\log_2\left(1 + \frac{P_s|h_{dl}|^2}{\Gamma(g_{dl})di^{\beta}N_o}\right) \tag{3}$$

We assume that there is the same noise behavior in the transmission of uplink and downlink. Here, B represents the bandwidth, N is the total number of sub-carriers, n denotes the sub-carriers allocated to the Ed for transmission. P_u and P_s refer to the transmission power of the Ed and the MES, respectively. h_{ul} and h_{dl} are the channel fading coefficients for uplink and downlink, β represents the path loss component,

while g_{ul} and g_{dl} are the required bit error rates for uplink and downlink, respectively. The $\Gamma(g_{ul}) = \frac{-2\log(5g_{ul})}{3}$ and $\Gamma(g_{dl}) = \frac{-2\log(5g_{dl})}{3}$ represent the SNR margin to meet the required bit error rate with a QAM (Quadrature Amplitude Modulation) array.

B. COMPUTATIONAL MODEL

The total time taken for processing the task on MES is:

Total time = uploading time + processing time + downloading time.

$$\mathbf{T} = T_{ul} + T_p + T_{dl} \tag{4}$$

The uploading time can be calculated from equation;

$$\mathbf{T_{ul}} = \frac{d}{r_{ul}} \tag{5}$$

where, T_{ul} is the uploading time, d is the data size in megabytes, r_{ul} is the maximum achievable uplink data rate for an additive white Gaussian noise (AWGN), the equation is taken from [38].The uploading scenario is shown in figure 2.

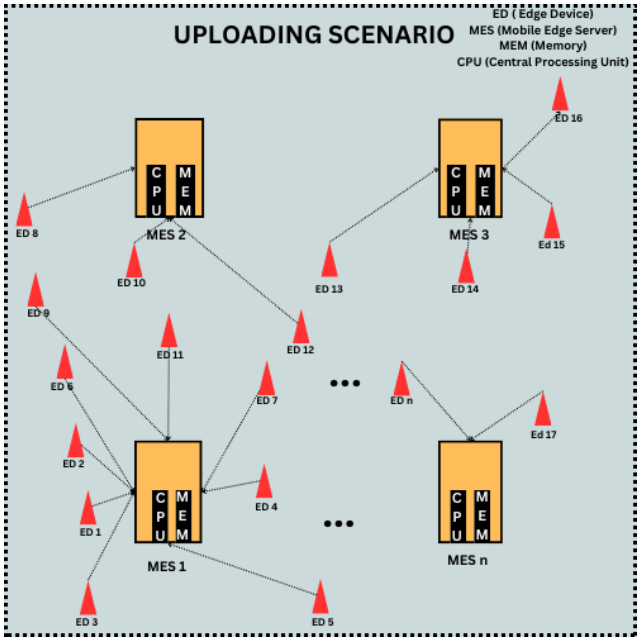

FIGURE 2. Uploading scenario.

The processing time can be calculated from equation..

$$\mathbf{T_p} = \frac{W_c}{mf_s} \tag{6}$$

where, T_p is the processing time, W_c is the workload ($W_c = d \times C$), C is CPU clock cycle per byte, d is the data size in megabytes, m is the number of CPU cores allocated for execution of task, f_s is CPU rate of MES, the equation is taken from [38]. The downloading time can be calculated from equation;

$$\mathbf{T_{dl}} = \frac{\lambda d}{r_{dl}} \tag{7}$$

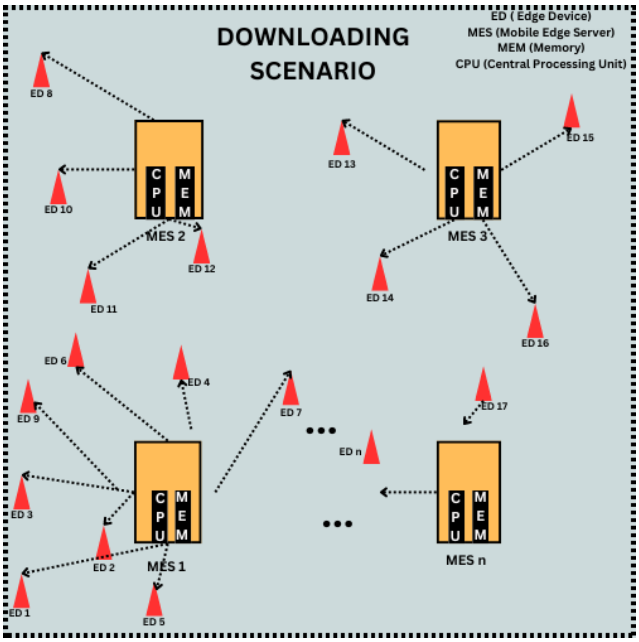

FIGURE 3. Downloading scenario.

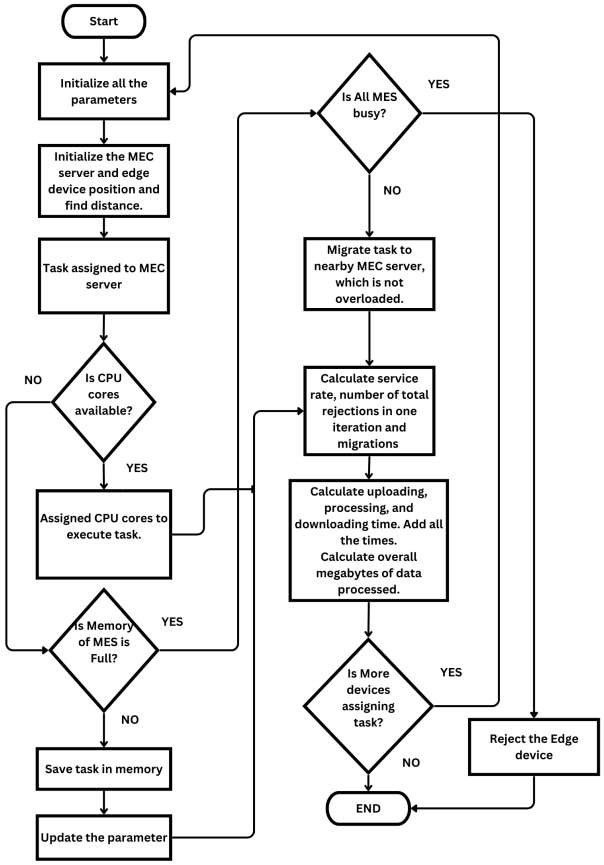

FIGURE 4. Step-wise representation of proposed algorithm.

where, T_{dl} is the downloading time, d is the data size in megabytes, *lambda* is any constant that has value 0.1, r_{dl} is

the maximum achievable downlink data rate, the equation is taken from [38]. The downloading scenario is shown in figure 3.

The service rate in a local area network can be calculated from equation:

$$\mathbf{S_r} = (\frac{N_{rs}}{N_{rT}}) \times 100 \tag{8}$$

where, S_r is the service rate, N_{rs} is the number of edge devices requests that are served by MES in a local area network, N_{rT} is the total number of edge devices requests that are received by MES in a local area network, this equation is taken from [39].

IV. ALGORITHM DESCRIPTION

In this section, we will discuss the proposed algorithm and its limitations, we will also discuss benchmark techniques that are implemented in this paper for comparison with the proposed algorithm.

A. PROPOSED ALGORITHM (CMMRA)

The CMMRA (CPU Memory Mobility Resource Allocation) algorithm is designed to manage computational resources effectively across MES while accommodating the mobility of Ed. As devices may move over time, the algorithm dynamically updates their locations by generating new random positions periodically. For each new position, it calculates the distance between the Ed and the MES. If the device falls within the MES's communication range, it is allowed to upload its data for computational processing; however, if the device is out of range, it is rejected for computational services due to the inability to maintain an efficient connection.

Since the algorithm continuously updates each device's position, it recalculates distances at every step to adapt to their new locations. If the device reappears within the MES range after moving, data can be downloaded back to it based on this updated distance. This dynamic tracking of distance and connectivity ensures that the devices can exchange data as their positions change.

Once a device is within communication range, the algorithm performs a series of resource checks on the MES to confirm whether the device's task can be processed immediately. First, it checks if communication sub-carriers are available to support data transfer. Following that, it verifies whether there are any free CPU cores to handle the device's request. If there are available CPU cores, the data is processed promptly. However, if all CPU cores are busy, the data is stored in the MES's memory until a CPU core becomes available. This two-tiered check ensures efficient resource allocation by utilizing memory as a backup when processing resources are temporarily limited.

If an MES is overloaded, meaning that both CPU cores are fully utilized and memory is also full, the algorithm activates its task migration mechanism, where it attempts to offload tasks to other MES within communication range.

Algorithm 1 CMMRA

Initialize number of servers, edge devices, communication range, edge devices positions, server position.
Calculate distance between devices and servers.
Random data, required space for saving data, required cores for processing.
$n \in [1, 512]$,$m \in [1, 32]$, $d \in [1000, 4000]$, $Ed = 100$, $s = 3$.
for Edge device $i = 1, 2, \ldots, Ed$ **do**
  for Server $j = 1, 2, \ldots, s$ **do**
    if the Ed is in range,the sub-carriers available **then**
      if CPU cores are available **then** Assign CPU cores.
      else if memory is available **then** Save in memory and wait for CPU cores to free.
      else. Migrate to other MES.
      end if
    else. Reject Ed from Computation.
    end if
  end for
end for
Find the new Ed Position, calculate distnace between Ed and Server.
for Edge device $i = 1, 2, \ldots, Ed$ **do**
  for Server $j = 1, 2, \ldots, s$ **do**
    if i is <= total Ed - migrations - rejections **then**
      Calculate Uplink rate from equation 2,Uplink time from equation 5, Proceesing time from equation 6.
      if new distance equals to old distance **then** Calculate downlink rate from equation 3 and downlink time at old distance from equation 7.
      else. Calculate downlink rate from equation 3 and downlink time at new distance from equation 7.
      end if
      use equation 4 to calculate all time and also calculate all data processed.
    else if i > total Ed - migrations - rejections **then**
      Calculate Uplink rate from equation 2,Uplink time from equation 5, Proceesing time from equation 6.
      if new distance equals to old distance **then** Calculate downlink rate from equation 3 and downlink time at old distance from equation 7.
      else. Calculate downlink rate from equation 3 and downlink time at old distance from equation 7.
      end if
      use equation 4 to calculate all time and also calculate all data processed.
    end if
  end for
end for
Calculate Service Rate from equation 8, Rejection Rate, Migrations.
Plot the results.

By redistributing tasks to nearby MES with available resources, the CMMRA algorithm minimizes the risk of overloading any single MES, thereby improving overall system efficiency. This resource sharing capability allows tasks to be transferred to alternate servers, preventing device rejection due to the unavailability of resources on one server. However, if all MES within range are also overloaded and have neither free CPU cores nor memory, the device's task is ultimately rejected as there are no available resources to meet its requirements.The whole algorithm is step by step discussed in Algorithm 1 and in Figure 4.

1) LIMITATIONS

In the proposed CMMRA algorithm, improvements are shown in detecting and managing the state of overloading in MES. There are several limitations identified in the implementation of the algorithm; correction can serve to improve efficiency further in resource allocation in MEC.

First, the existing algorithm, in fact, does not really take into account the Ed specific computational time required by it to finish its task. In quite a number of cases, when the involved processing time taken by a particular Ed exceeds the time slot allowed, the corresponding Ed hogs the computation resources for longer without being released, which could lead to a probable bottleneck and less efficient resource utilization by the device. The time that is left will then be checked for each Ed and, in the event of exceeding the specified timeslot, should be forwarded to an underloaded MES node or declined if not possible due to available resources. Such an optimization can also benefit management by cutting service interruptions and device rejection because of overload.Another thing would be to have a time slot management mechanism in the algorithm, which would significantly improve the efficiency of the whole process. This would involve assigning a time slot for computation to every device beforehand. If any device completes its tasks within the end of the time slot, then the remaining time can be dynamically reallocated to another waiting device in the queue. This would minimize idle time slots and serve waiting Ed very effectively so that use of MES resources is optimized and queue times reduced for devices that instant computational support.

B. BENCHMARK TECHNIQUES

1) DMRCP

An algorithm, DMRCP: Dynamic Markov Model for Resource Contention Prediction [11], designed for edge computing environments in the context of predicting and managing CPU usage on MES, tracks offloaded tasks on the servers and identifies instances of overload when all CPU cores are occupied. Such instances are logged in a history matrix that is used to update a transition probability matrix for predicting the server's future state. The model moves the task to a different server if it finds a risk of server overload in order to keep resources balanced. In DMRCP, the key indication of server load is the usage of CPU cores, while the mobility of edge devices (Ed) is also taken into account. With every need for computational resources by a device, it generates a random location and requests the MES. The model calculates the distance between the MES and the device. If this device is in the communication range, the CPU cores can be allocated for the computation. Otherwise, the device is migrated to the underloaded MES that falls within the communication range. If all the MES resources are occupied so that each MES CPU core is used, then the request of the device for the computational services is rejected. Device mobility is monitored all the time; the computational services being provided are cancelled if a device goes out of communication range. Through these mechanisms, DMRCP tries to predict overload situations in advance and can manage them. It distributes the load effectively and makes optimum use of resources.

2) DPO

Cloud computing provides a pool of resources, such as virtual machines, which are equipped with CPU, storage, memory, network bandwidth, and databases. The allocation process of such VMs into the system in an efficient as well as secure manner has proved to be quite demanding in terms of time requirements. To overcome this, DPO, an advanced algorithm for security optimization, is proposed in [12]. This method selects VMs based on factors such as energy, memory, and even CPU usage. In doing so, it provides the additional layer of hypervisor security.In the DPO approach, when an Ed demands computational resources, the algorithm calculates the distance between the edge device and the MES. If the device is in communication range, DPO allocates the required number of CPU cores and memory according to the device's demand and the available resources of the MES. The algorithm re-evaluates the VM selection if an MES is prone to overloading due to high CPU or memory demand. It differentiates itself from others, which do not consider Eds' mobility but focus instead on preventing overloads for MES by also accounting for CPU and memory use. In case the MES is fully loaded, having exhausted all its CPU cores as well as memory, either a request from the Ed is directed to a underloaded MES or the request is turned away. It further implements streamlined security protocols whereby chosen VMs are extremely secure and reliable. All of these mechanisms optimize the usage of resources, distribute load, and maintain security on edge and cloud computing environments.

3) ARCES

The Automated Resource Controller for Energy-aware Server (ARCES) [40] is an approach to the management of computing and communication processes within virtualized computing platforms of the MEC framework. ARCES has the purpose of optimizing energy efficiency without degrading quality of service (QoS). It predicts demand of servers,

handles the process of VM scaling, distributes workload, processes a rate allocation of transmission, and makes an adjustment in transmission drivers by scheduling location-aware traffic.When an Ed requires computational resources, ARCES calculates the distance from the Ed to the MES and determines if the device is in the communication range. If the device is within the communication range, ARCES assigns its CPU cores to the MES based on the processing required by the device and the available CPU capacity of the MES. Unlike other algorithms, ARCES does not consider memory as a factor in determining overloaded states, although it considers CPU core utilization when detecting MES overload. Therefore, the MES is overloaded in case the CPU core capacity is fully utilized and thus ARCES either scales available VMs or it redirects requests to another suitable server if possible. Location-aware traffic scheduling further improves resource allocation as it changes the transmission drivers based on network conditions to efficiently manage data transfer. In contrast, ARCES adapts to the MEC environment through CPU core and energy-aware VM scaling, missing Ed mobility, and memory utilization.

TABLE 2. Parameters for simulations.

Symbol	Value	Symbol	Value
B	0.5 MHz	$Memory$	32768 MB
C	12000	M	32
f_s	10^9	N	512
g_{dl}	10^{-5}	N_o	5×10^{-5}
g_{ul}	10^{-5}	P_s	0.1
h_{dl}	5	P_u	0.01
h_{ul}	5	β	2

V. SIMULATIONS AND RESULTS

We use MATLAB 2021a on core i5 CPU @1.3GHz for simulations. We have considered 100 devices and 3 servers, and random data for each device is generated. The communication range of the area is 500 m. Total sub-carriers in the communication area are N = 512. Subcarriers required for edge device may range from n = 1 to 512. CPU cores required for processing are m = 1 to 32. The loop is run 5 times to get the service rate, migrations and rejection rate results.

The parameters discussed in table 2 are taken based on realistic network scenarios.The total frequency range that the edge server can use to send data to mobile devices is known as the bandwidth, or B. To provide fair data rates with little interference, a bandwidth of 0.5 MHz can be used. Because of the bandwidth restriction, such a scenario is helpful for replicating real-time resource management systems that are sensitive to system throughput and latency.The path loss exponent (β) gives the rate at which the signal attenuates with distance. In an urban or suburban environment, interference is present but not as extreme as in a dense metropolitan environment; thus, it is appropriate to set $\beta = 2$. The value is used to model the signal attenuation realistically; this affects the efficiency of resource allocation in edge networks.The MES's CPU rate C indicates how many cycles it takes to process a byte for each calculation. It indicates how many CPU cycles are needed to process each byte. The important role of low latency and high throughput necessitates a comparable influence on the system's processing speed and overall responsiveness in jobs close to the network edge, making this characteristic applicable in MEC settings.The parameter f_s is the CPU rate of the MES, measured in cycles per second, or Hertz. A value of 10^9 would mean that the server can carry out a billion processing cycles per second. With this high CPU rate, the MES could easily manage and process tremendous amounts of data workloads without delays to respond accordingly to the users' demands. In a MEC environment, such responsiveness is critical because the applications are sensitive to delays and require rapid, real-time data processing for the optimization of resources and maintaining low latency.

The uplink and downlink communication BERs are denoted by the parameters g_{dl} and g_{ul} respectively. Values of 10^{-5} suggest that there may be data transport errors. In order to simulate the dependable communication channels required for steady and effective data transfers between Ed and the MES, BER low values are chosen. In order to guarantee data integrity, lower retransmission cost, and improve system speed and responsiveness, edge computing in mobile networks should minimize bit error rates.In MEC, fading coefficients of the downlink and uplink channels are denoted by h_{dl} and h_{ul}, representing the strength and quality of signals that are exchanged between Ed and MES through downlink and uplink channels, respectively. Channel fading coefficients are very important in MEC since they reflect the real effects of signal degradation in a channel due to reasons such as distance, interference, and obstacles between devices and servers. The value of the channel fading coefficients is taken as 5 for moderate fading conditions because it has been observed that most of the real MEC environments are moderate.A crucial part of this procedure is the value N, which stands for the number of subcarriers and is set to 512. Instead of directly competing for the same frequency, each sub-carrier functions as a distinct "lane" in the bandwidth, enabling numerous users to send data simultaneously. With 512 sub-carriers, the bandwidth is sufficiently finely divided to enable resource allocation based on OFDMA. Noise power density (N_o) is assumed to model the background noise. A 5×10^{-5} value creates a real noise floor affecting the quality of data and becomes one of the bottlenecks in resource optimization when channels are not perfect. The P_s and P_u represent the powers of MES and Ed respectively, which is used to denotes that both the MES and Ed operated under some power constraints.In this research paper, we have introduced the concept of overloading of both CPU cores and memory. In a realistic scenario, memory should have some value. So, for simulation purposes, we have to consider 32,678 MB of memory for one server. $M = 32$ represents the total number of CPU cores of single MES.

A. TIME

In Figure 5, the computational time graph has been plotted. The time taken has been taken on the y-axis, and the total data processed by the MES is taken on the x-axis. As shown in figure 5, our proposed algorithm, CMMRA, has taken less time and processed more data as compared to DMRCP, DPO, and ARCES. The time plotted is the total time that all the MES take to process the data from all the Eds. By considering the CPU cores, memory, and mobility of devices, our proposed algorithm presents the best realistic scenario and reduces the time taken to process the data.

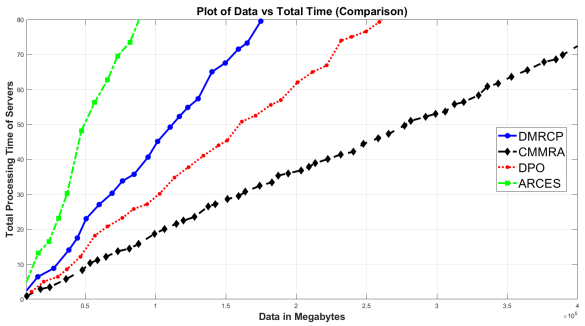

FIGURE 5. Computational time of mobile edge servers.

B. SERVICE RATE

In Figure 6, the service rate of the communication area has been plotted. The results have been produced by taking different iterations in a loop. As shown in figure 6, by increasing the number of MES in a communication area our proposed algorithm CMMRA has a better service rate than all the proposed schemes. It shows that our proposed algorithm will provide more computational services to Eds.

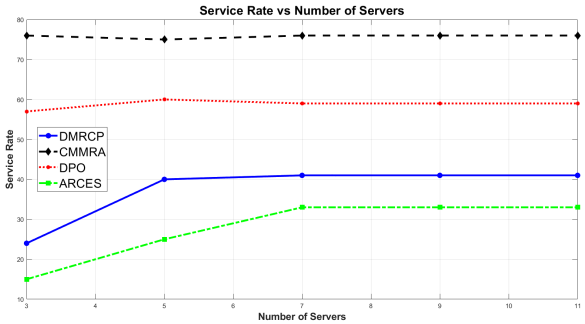

FIGURE 6. Service rate of communication area.

C. REJECTIONS

In Figure 7, the rejection rate in a communication area has been plotted. As discussed in the previous result, the service rate is maximum because computational services are provided to maximum Eds. So, the maximum number of devices are getting computational services, and fewer rejections will be given to edge devices when all MES are busy. By considering CPU cores and memory both for the overloading state of MES, it reduces rejections and increases the service rate.

FIGURE 7. Rejections of computational services to edge devices.

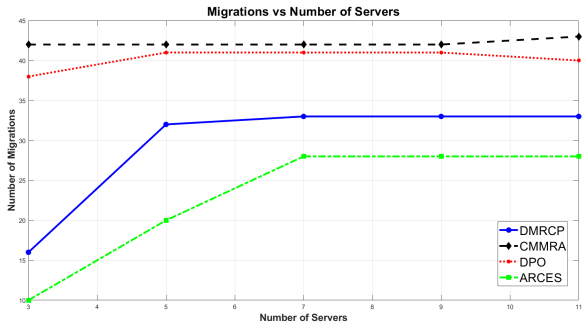

FIGURE 8. Migrations of edge devices.

D. MIGRATIONS

In Figure 8, the total number of VM migrations of devices is plotted. In our proposed algorithm, CMMRA, as shown in 7, we have the maximum number of VM migrations, as we have considered the mobility of Eds. Another factor that is causing more migrations is that many devices are sending requests for computational services, and when the MES is overloaded, it automatically migrates the required data to be processed on an underloaded MES in a communication range.

VI. CONCLUSION

In this paper, we proposed an algorithm CPU Memory Mobility Resource Allocation (CMMRA). We consider CPU cores, memory, mobility of devices and multi sever multi devices scenario in our algorithm. The algorithm did not prevent the Eds from providing computational resources and encourage to migrate the devices to other underloaded MES for computation. This reduces computational time, increases service rate, and decreases rejection rate in a local area network. Our proposed algorithm performs better than other algorithms in the same scenario.

In future work, we can concentrate on three main areas to enhance resource management in MEC: predictive load balancing, task prioritization, and memory optimization. First, predictive load balancing will leverage machine learning to anticipate incoming demands, enabling the

relocation of tasks before a MES reaches its capacity. Second, by prioritizing tasks according to their significance and resource requirements, we can ensure that urgent tasks receive the necessary CPU power and memory promptly. Lastly, memory optimization strategies, such as caching and data normalization, will help minimize unnecessary memory consumption, freeing up more resources for processing active tasks. These enhancements can result in improved performance and efficiency in managing resources across MES.

FUNDING STATEMENT

The APC was funded by the Faculty of Electrical and Computer Engineering, Cracow University of Technology, and the Ministry of Science and Higher Education, Republic of Poland (grant no. E-1/2024).

REFERENCES

[1] P. Bhattacharya, S. Tanwar, R. Shah, and A. Ladha, "Mobile edge computing-enabled blockchain framework—A survey," in *Proc. ICRIC*, 2020, pp. 797–809.
[2] D. Dechouniotis, N. Athanasopoulos, A. Leivadeas, N. Mitton, R. Jungers, and S. Papavassiliou, "Edge computing resource allocation for dynamic networks: The DRUID-NET vision and perspective," *Sensors*, vol. 20, no. 8, p. 2191, Apr. 2020.
[3] T. Dreibholz, S. Mazumdar, F. Zahid, A. Taherkordi, and E. G. Gran, "Mobile edge as part of the multi-cloud ecosystem: A performance study," in *Proc. 27th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, 2019, pp. 59–66.
[4] C. Sun, H. Li, X. Li, J. Wen, Q. Xiong, and W. Zhou, "Convergence of recommender systems and edge computing: A comprehensive survey," *IEEE Access*, vol. 8, pp. 47118–47132, 2020.
[5] W. Sun, J. Liu, and H. Zhang, "When smart wearables meet intelligent vehicles: Challenges and future directions," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 58–65, Jun. 2017.
[6] F. Zhou, R. Q. Hu, Z. Li, and Y. Wang, "Mobile edge computing in unmanned aerial vehicle networks," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 140–146, Feb. 2020.
[7] M. Mehrabi, D. You, V. Latzko, H. Salah, M. Reisslein, and F. H. P. Fitzek, "Device-enhanced MEC: Multi-access edge computing (MEC) aided by end device computation and caching: A survey," *IEEE Access*, vol. 7, pp. 166079–166108, 2019.
[8] I. Sittón-Candanedo, R. S. Alonso, Ó. García, A. B. Gil, and S. Rodríguez-González, "A review on edge computing in smart energy by means of a systematic mapping study," *Electronics*, vol. 9, no. 1, p. 48, Dec. 2019. [Online]. Available: https://www.mdpi.com/2079-9292/9/1/48
[9] S. Svorobej, P. T. Endo, M. Bendechache, C. Filelis-Papadopoulos, K. M. Giannoutakis, G. A. Gravvanis, D. Tzovaras, J. Byrne, and T. Lynn, "Simulating fog and edge computing scenarios: An overview and research challenges," *Future Internet*, vol. 11, no. 3, p. 55, Feb. 2019. [Online]. Available: https://www.mdpi.com/1999-5903/11/3/55
[10] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
[11] K. Surya and V. M. A. Rajam, "Novel approaches for resource management across edge servers," *Int. J. NetWorked Distrib. Comput.*, vol. 11, no. 1, pp. 20–30, Jun. 2023.
[12] D. Dhanya and D. Arivudainambi, "Dolphin partner optimization based secure and qualified virtual machine for resource allocation with streamline security analysis," *Peer-to-Peer Netw. Appl.*, vol. 12, pp. 1194–1213, Sep. 2019.
[13] Z. Zhao, G. Min, W. Gao, Y. Wu, H. Duan, and Q. Ni, "Deploying edge computing nodes for large-scale IoT: A diversity aware approach," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3606–3614, Oct. 2018.
[14] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An overview on edge computing research," *IEEE Access*, vol. 8, pp. 85714–85728, 2020.
[15] N. Shan, Y. Li, and X. Cui, "A multilevel optimization framework for computation offloading in mobile edge computing," *Math. Problems Eng.*, vol. 2020, pp. 1–17, Jun. 2020.
[16] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.
[17] X. Chen, Q. Shi, L. Yang, and J. Xu, "ThriftyEdge: Resource-efficient edge computing for intelligent IoT applications," *IEEE Netw.*, vol. 32, no. 1, pp. 61–65, Jan. 2018.
[18] F. Jia, H. Zhang, H. Ji, and X. Li, "Distributed resource allocation and computation offloading scheme for cognitive mobile edge computing networks with NOMA," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Aug. 2018, pp. 553–557.
[19] K. Poularakis et al., "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc. IEEE INFOCOM*, Paris, France, 2019.
[20] I. A. Elgendy, W. Zhang, Y.-C. Tian, and K. Li, "Resource allocation and computation offloading with data security for mobile edge computing," *Future Gener. Comput. Syst.*, vol. 100, pp. 531–541, Nov. 2019.
[21] G. Yang, L. Hou, X. He, D. He, S. Chan, and M. Guizani, "Offloading time optimization via Markov decision process in mobile-edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2483–2493, Feb. 2021.
[22] K. R. Alasmari, R. C. Green, and M. Alam, "Mobile edge offloading using Markov decision processes," in *Proc. 2nd Int. Conf. Edge Comput.*, Seattle, WA, USA. Cham, Switzerland: Springer, Jun. 2018, pp. 80–90.
[23] A. Mondal and P. S. Chatterjee, "A systematic literature survey on data security techniques in a cloud environment," in *Proc. OITS Int. Conf. Inf. Technol. (OCIT)*, Dec. 2022, pp. 451–456.
[24] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Toward an intelligent edge: Wireless communication meets machine learning," *IEEE Commun. Mag.*, vol. 58, no. 1, pp. 19–25, Jan. 2020.
[25] X. Liu, Z. Qin, and Y. Gao, "Resource allocation for edge computing in IoT networks via reinforcement learning," in *Proc. ICC - IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[26] X. Liu, J. Yu, J. Wang, and Y. Gao, "Resource allocation with edge computing in IoT networks via machine learning," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3415–3426, Apr. 2020.
[27] N. Shan, X. Cui, and Z. Gao, "'DRL+FL': An intelligent resource allocation model based on deep reinforcement learning for mobile edge computing," *Comput. Commun.*, vol. 160, pp. 14–24, Jul. 2020.
[28] A. Mondal and P. S. Chatterjee, "CloudSec: A lightweight and agile approach to secure medical image transmission in the cloud computing environment," *Social Netw. Comput. Sci.*, vol. 5, no. 2, p. 237, Jan. 2024.
[29] A. Mondal, P. S. Chatterjee, and N. K. Ray, "An optimal novel approach for dynamic energy-efficient task offloading in mobile edge-cloud computing networks," *Social Netw. Comput. Sci.*, vol. 5, no. 5, p. 655, Jun. 2024.
[30] C. Jiang, T. Fan, H. Gao, W. Shi, L. Liu, C. Cérin, and J. Wan, "Energy aware edge computing: A survey," *Comput. Commun.*, vol. 151, pp. 556–580, Feb. 2020.
[31] T. Bahreini, H. Badri, and D. Grosu, "Energy-aware capacity provisioning and resource allocation in edge computing systems," in *Proc. 3rd Int. Conf. Edge Comput.*, San Diego, CA, USA. Cham, Switzerland: Springer, Jun. 2019, pp. 31–45.
[32] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.
[33] A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Wireless Commun. Lett.*, vol. 6, no. 3, pp. 398–401, Jun. 2017.
[34] D. Spatharakis, I. Dimolitsas, D. Dechouniotis, G. Papathanail, I. Fotoglou, P. Papadimitriou, and S. Papavassiliou, "A scalable edge computing architecture enabling smart offloading for location based services," *Pervas. Mobile Comput.*, vol. 67, Sep. 2020, Art. no. 101217.
[35] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic resource allocation exploiting mobility prediction in mobile edge computing," in *Proc. IEEE 27th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Sep. 2016, pp. 1–6.
[36] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers," in *Proc. 39th Euromicro Conf. Softw. Eng. Adv. Appl.*, Sep. 2013, pp. 357–364.
[37] H. Shingne, S. Sountharrajan, M. Karthiga, and C. Rajan, "Lasso and ridge regression for optimized resource allocation in cloud computing," *J. Adv. Res. Dynam. Contr. Syst.*, vol. 12, pp. 1740–1747, 2020.
[38] Z. Ali, L. Jiao, T. Baker, G. Abbas, Z. H. Abbas, and S. Khaf, "A deep learning approach for energy efficient computational offloading in mobile edge computing," *IEEE Access*, vol. 7, pp. 149623–149633, 2019.
[39] Z. Ali, S. Khaf, Z. H. Abbas, G. Abbas, F. Muhammad, and S. Kim, "A deep learning approach for mobility-aware and energy-efficient resource allocation in MEC," *IEEE Access*, vol. 8, pp. 179530–179546, 2020.

[40] T. Dlamini and Á. F. Gambín, "Adaptive resource management for a virtualized computing platform within edge computing," in *Proc. 16th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, Jun. 2019, pp. 1–9.

SAIFUR RAHMAN received the Ph.D. degree in electrical and electronic engineering. He has more than 12 years of academic experience in teaching and research. He is currently an Associate Professor with the College of Engineering, Najran University, Saudi Arabia. He has authored more than 100 research articles in reputed journals, books, and conference proceedings. His main research interests include artificial intelligence, the Internet of Things (IoT), big data analytics, smart cities, and smart healthcare.

MAZHAR HUSSAIN received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Mardan, Pakistan, in 2024. He is currently a Laboratory Engineer with the Faculty of Electrical Engineering, Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi, Pakistan. His research interest includes resource management in mobile edge computing. His other interest focused on programming the microprocessor and digital signal processing boards.

SYED IBAD ALI SHAH received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Mardan, Pakistan, in 2024, where he is currently pursuing the master's degree in electrical engineering. His research interest includes resource management in mobile edge computing. His other interests focused on wireless communication and signal and processing.

ZAIWAR ALI received the B.S. degree in electronics engineering from the COMSATS Institute of Information Technology, Abbottabad Campus, Pakistan, in 2012, the M.S. degree in electronics engineering from the Ghulam Ishaq Khan Institute of Engineering Sciences and Technology (GIK Institute), Topi, Pakistan, in 2015, and the Ph.D. degree in electronics engineering from the Telecommunications and Networking Research Laboratory, GIK Institute, in 2021. From 2013 to 2015, he was a Graduate Assistant with the GIK Institute. From 2016 to 2018, he was a Lecturer with the Faculty of Electrical Engineering, GIK Institute, where he is currently an Assistant Professor. His research interests include multi-access edge computing, cloud computing, stochastic processes, the IoT, machine learning, and wireless sensor networks. He was a recipient of the Highest Level of Merit Scholarship.

MUHAMMAD AYAZ KHAN received the B.Sc. degree in electrical engineering from CECOS University, Peshawar, in 2014, the M.Sc. degree in energy engineering from the University of Bologna, Italy, in 2018, and the second M.Sc. degree in engineering management from the University of Chester, U.K., in 2023. He is an accomplished engineer with a diverse educational background. Throughout his career, he has successfully led and managed various electrical projects. His research interests include renewable energy, high-to-medium voltage distribution, smart grids, project initiation, scoping, budgeting, and 6G networks.

GRZEGORZ NOWAKOWSKI is currently an Assistant Professor with the Department of Automatics and Informatics, Cracow University of Technology, Poland, specializing in computer science. He focuses on fuzzy database queries and their application in big data analysis. His research interests include information systems, computational intelligence, databases, big data, cloud computing, and soft computing methods.

MAREK SIEJA is currently with the Faculty of Electrical and Computer Engineering, Cracow University of Technology, Poland. His research interests include computer engineering, networks and communication, and electrical measurements.

MUHAMMAD IRFAN received the Ph.D. degree in electrical and electronic engineering from Universiti Teknologi PETRONAS, Malaysia, in 2016. He is currently an Associate Professor with the Electrical Engineering Department, Najran University, Saudi Arabia. He has over 10 ten years of experience in industry, academia, and research. He has one patent granted by U.S. Patent Office and one patent granted by Malaysian Patent Office. He secured several research grants. He has authored several research articles in reputed journals, books, and conference proceedings (Google Scholar Citations 5000 and H-index of 30). His main research interests include artificial intelligence, intelligent fault diagnosis, intelligent fault tolerant control, intelligent robotics, smart grids, smart cities, and smart healthcare systems. He received several research awards, including the Best Researcher Award from Najran University.

FAZAL MUHAMMAD received the B.Sc. and M.S. degrees in electrical engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2004 and 2007, respectively, and the Ph.D. degree from the Ghulam Ishaq Khan Institute of Engineering Science and Technology, Topi, Pakistan, in 2017. He is currently an Assistant Professor with the Department of Electrical Engineering, University of Engineering and Technology, Mardan. His research interests include modeling and analysis of heterogeneous cellular networks using tools from stochastic geometry, point process theory, and spatial statistics, interference channel modeling, cognitive radio networks, mmWave, optical networks, and antenna modeling and design for 5G applications.