IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 18 December 2023, accepted 31 December 2023, date of publication 5 January 2024,
date of current version 18 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3350556

RESEARCH ARTICLE

EdgeMatch: A Smart Approach for Scheduling IoT-Edge Tasks With Multiple Criteria Using Game Theory

ANJAN BANDYOPADHYAY[1], VAGISHA MISHRA[1], SUJATA SWAIN[1], KALYAN CHATTERJEE[2], SWETA DEY[3], SAURAV MALLIK[4], (Member, IEEE), AMAL AL-RASHEED[5], MOHAMED ABBAS[6], AND BEN OTHMAN SOUFIENE[7]

[1]School of Computer Engineering, Kalinga Institute of Industrial Technology Deemed to be University, Bhubaneswar, Odisha 751024, India
[2]Department of Computer Science and Engineering, Nalla Malla Reddy Engineering College, Hyderabad 500088, India
[3]Indian Institute of Technology Ropar, Rupnagar 140001, India
[4]Harvard T. H. Chan School of Public Health, Boston, MA 02115, USA
[5]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
[6]Department of Electrical Engineering, College of Engineering, King Khalid University, Abha 61421, Saudi Arabia
[7]Prince Laboratory Research, ISITcom, University of Sousse, Hammam Sousse, Sousse 4011, Tunisia

Corresponding author: Ben Othman Soufiene (soufiene.benothman@isim.rnu.tn)

This research was financially supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R235), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University (KKU) for funding this research through the Research Group Program Under the Grant Number:(R.G.P.2/572/44).

ABSTRACT For an extended period, a technological architecture known as cloud IoT links IoT devices to servers located in cloud data centers. Real-time data analytic are made possible by this, enabling better, data-driven decision making, optimization, and risk reduction. Since cloud systems are often located at a considerable distance from IoT devices, the rise of time-sensitive IoT applications has driven the requirement to extend cloud architecture for timely delivery of critical services. Balancing the allocation of IoT services to appropriate edge nodes while guaranteeing low latency and efficient resource utilization remains a challenging task. Since edge nodes have lower resource capabilities than the cloud. The primary drawback of current methods in this situation is that they only tackle the scheduling issue from one side. Task scheduling plays a pivotal role in various domains, including cloud computing, operating systems, and parallel processing, enabling effective management of computational resources. In this research, we provide a multiple-factor autonomous IoT-Edge scheduling method based on game theory to solve this issue. Our strategy involves two distinct scenarios. In the first scenario, we introduced an algorithm containing choices for the IoT and edge nodes, allowing them to evaluate each other using factors such as delay and resource usage. The second scenario involves both a centralized and a distributed scheduling approach, leveraging the matching concept and considering each other. In addition, we also introduced a preference-based stable mechanism (PBSM) algorithm for resource allocation. In terms of the execution time for IoT services and the effectiveness of resource consolidation for edge nodes, the technique we use achieves better results compared with the two commonly used Min-Min and Max-Min scheduling algorithms.

INDEX TERMS Edge computing, IoT, fog computing, resource allocation, game theory, matching algorithm, centralized matching, distributed matching.

I. INTRODUCTION

Internet of Things (IoT) is becoming a popular thing in our daily lives. Everyone used the practical implementation of the IoT devices to get more advantages [4]. IoT devices create a

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

link between wearable devices and smart meters in our smart cities. As a result, there has been a globally unparalleled rise in the deployment and use of IoT devices [5]. For instance, Cisco predicts that by 2030, there will be 30 billion IoT devices [6]. Due to its extensive and high-capacity processing and storage capabilities, cloud computing [7], [8], [9] has remained the favored choice for IoT producers and suppliers to fulfill their storage and computational requirements [10], [11], [12]. However, the geographical distance between cloud datacenters [13], [14] and IoT devices, often located in remote areas, can lead to increased latency and delays in high-performance systems. Consider a scenario where a wearable device worn by a patient transmits brain data for rapid analysis. In this case, any delay in transmitting the information processing call to the designated server could escalate the potential health risks for the patient [15].

This challenge becomes particularly pronounced in a cloud-based environment, where analysis is required to be performed in regional cloud data centers. In light of the fact that applications for intelligent transportation systems and other delay-critical services sometimes need answers in milliseconds, this poses a severe difficulty. Fog computing [16], [17] is an emerging method that addresses latency problems by enabling data analysis at the edge [18], [19] in close proximity to IoT device sources [20], [21]. It is important to note that IoT devices have limited computational network, and storage capabilities [22], [23]. For example, these devices can consume substantial network bandwidth when exchanging data. This might result in bottleneck difficulties and slow down the entire network. By giving IoT devices access to quick data analyses and decision-making tools at the network's edge, fog nodes help to mitigate this issue [24], [25]. The resource constraints that fog nodes commonly experience are the main cause of the system's multiple major problems, which the fog computing system possesses despite its advantages. To increase both the number of tasks that can be done and the Quality of Service (QoS) [26] that is provided for every job, these resources must be properly employed and handled.

When fog computing is deployed, data is examined within an IoT gateway. Data is examined on the sensor or device itself via edge computing. In short, data is not transported anywhere when edge computing is used. This lowers expenses and enables real-time data analysis, improving performance [27], [28]. The most important aspect of edge computing is the effective distribution of scarce resources, and the effectiveness of this process directly impacts how well the edge computing paradigm as a whole performs(see Figure 1 depicting the architecture of edge computing). Due to its enormous success, edge computing is becoming more and more popular, and the research community is paying close attention to it. The crucial factor, namely the latency between the two sides, must be taken into account in order to properly profit from edge computing technology. In order to ensure quick and effective data and message transfer, IoT devices must connect to the edge nodes that are nearest to them. In this study, we tackle the challenge of efficiently associating IoT devices with appropriate edge nodes while considering the interplay of these two conflicting objectives. Here, the issue is to guarantee both optimal resource utilization on edge nodes and efficient task execution of IoT-generated activities. Task scheduling optimizes resource utilization, enhances system efficiency, and minimizes idle time by organizing and prioritizing tasks. It aids in meeting deadlines, improving throughput, and balancing workloads in diverse computing environments.

A. RESEARCH FOCUS

Several scheduling techniques have recently been presented for edge computing environments. These techniques' main goals are to improve and accelerate IoT task execution times at the edge and to optimize resource allocation. However, a key challenge with these approaches is that they often focus on optimizing certain factors that are applicable only to one side of the equation. This leads to an imbalanced situation where the requirements of one party are disregarded during the scheduling process. In recent times, game theory [29], [30] has emerged as a tool to generate wise scheduling choices that consider the interests of both sides. These strategies tend to have a business-centric orientation, incorporating business-related considerations into the construction of utility applications. The Quality of Service (QoS) [28] given to IoT services is maximized by the approach we use, in contrast, while simultaneously effectively regulating resource usage on the edge nodes.

B. INNOVATIVE PERSPECTIVE

In this research, we have proposed an intelligent autonomous scheduling technique in an IoT-edge context. This strategy employs game theoretic matching to make smart scheduling choices for edge and IoT gadgets. The method we employ differs from others in that it takes into account both the preferences and constraints of both types of devices when scheduling. Our approach aims to reduce the latency and duration of IoT service execution while also increasing the efficiency with which available resources on edge nodes are used. Here's a rephrased version of our summary

- We introduce an independent and smart scheduling solution for IoT services within edge computing environments using matching algorithm. This work stands out as a novel performance-centric method that takes into account the preferences and limitations of both edge and IoT aspects when formulating scheduling choices, as indicated in [7], [31], and [32].
- By formulating unique optimization challenges for individual stakeholders and suggesting specific constraints for each underlying problem, these difficulties are subsequently consolidated into a unified optimization problem, amenable to effective resolution using matching game theory.
- Developing preference metrics for both IoT devices and edge nodes, aiding them in evaluating and prioritizing

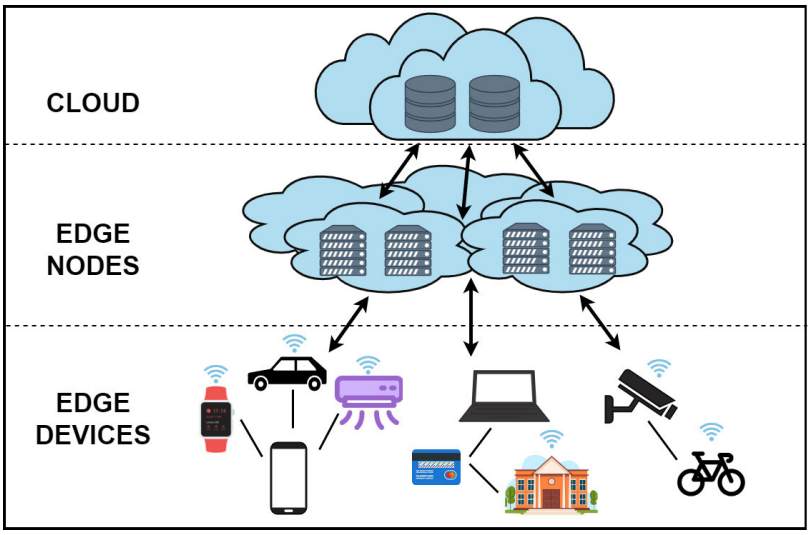

FIGURE 1. Architecture of edge computing.

each other using distinct criteria Furthermore, introducing methods for generating preferences will aid edge nodes and IoT devices in creating practical preference rankings aligned with the established preference metrics.

- The development of centralized and distributed scheduling algorithms By presenting two scheduling strategies, we want to make our solution more complete. Each kind has distinct benefits that apply in certain settings and applications. The concept of a centralized approach finds applicability in business scenarios where a central commanding organization becomes necessary to act as the focal point for all communication. The advantage of this tactic lies in the presence of a centralized governing body that assumes responsibility and addresses issues promptly upon emergence. The network is more secure [33] in such an environment because there is a centralized authority managing network scheduling. As an illustration, consider the network of weather stations consisting of microcontrollers that collect data on various environmental conditions across an expansive and remote region. Conversely, the decentralized approach could prove advantageous for rapidly creating small to medium-sized networks where there's no need for a large-scale and business-oriented setup. The above approach is also appropriate for gatherings with strict security considerations that lack confidence in a centralized entity to carry out the scheduling process. The preference-Based Stable Matching (PBSM) algorithm that we developed considers a scenario where there are n IoT devices expressing their clear preferences for various edge nodes, and at the same time, n available edge nodes express their distinct preferences for the targeted IoT devices. These preferences are flexible and can be defined as needed during the implementation process. Each IoT device receives its own edge node. The suggested approach then distributes each edge node to an IoT device, with the condition that each IoT device receives a single unique edge node.

In the second section, an examination of existing literature concerning task scheduling and associated algorithms in IoT environments is conducted. Moving on to the third section, difficulties with IoT and edge optimization aspects are formulated. The fourth section introduces the fundamentals and definitions of matching game theory, detailing preference functions and suggesting methods for creating preference lists. Transitioning to the fifth section, the proposed centralized, distributed matching, and PBSM algorithms are elucidated. In Section VI, the setup for the experiment and simulation is given in detail, and the explanation of the empirical results follows. Finally, in the concluding seventh section, a summary of the findings is presented.

II. LITERATURE REVIEW

In reference to the work outlined in [32], a study was presented addressing the tradeoff between makespan and cloud costs within the context of scheduling large-scale applications on a cloud platform. The study introduces an innovative scheduling algorithm referred to as the cost-makespan-aware scheduling heuristic. This algorithm aims to strike a balance between optimizing application execution performance and managing the requisite expenses associated with utilizing cloud resources. Additionally, the research proposes an effective strategy for task reassignment. This strategy leverages the critical path of a directed acyclic graph that models the application, enhancing the output schedules produced by the Cost-makespan-aware scheduling algorithm. These refinements were intended to meet user-defined deadline constraints and ensure the system's quality of service. In paper [33], an approach to scheduling tasks in the context of IoT is introduced, leveraging data mining techniques by the

authors. Their method involves the utilization of a prior task classification using an upgraded version of the prior rules-based algorithm; subsequently, they employ a unique strategy called TSFC (Task Scheduling in Fog Computing) for task scheduling.

This strategy primarily relies on time-based factors, disregarding the availability of resources at both the network (such as bandwidth) and fog node levels (comprising CPU, RAM, and power). Consequently, this approach poses a potential risk to the overall quality of service for these tasks. Alternatively, in the article [34], a resource allocation model using a decoupled methodology is introduced as Zenith, proposed by the authors as a fresh technique. This new strategy is built on an auction-based resource sharing mechanism, made possible by a contract that guarantees resource optimization for both fog nodes and service providers as well as the preservation of integrity. Nonetheless, their other abstracted layers, such as microdata centers (MDCs), must be put between the different endpoints in this design.This exhibits significant resource utilization and might not be appropriate for time-sensitive tasks. A task scheduling approach centered around prioritization tiers is proposed by Choudhari in their publication [7]. In the paper [41], the energy-aware scheduling of dependent tasks in heterogeneous multiprocessor systems is addressed by the author. Tasks and processors have been modeled, formulating an optimization problem to minimize task schedule length. An optimization problem is formulated to minimize task schedule length involving a task prioritization method and a weight-based energy distribution strategy, leading to the creation of a list-based energy-aware scheduling algorithm. Efficient task execution while meeting dependencies and energy constraints is ensured by this approach. Additionally, in the paper [42], an energy-efficient scheduling algorithm is introduced, leveraging an enhanced per-assignment strategy. This novel approach optimizes processor allocation, frequencies, and task start times while ensuring compliance with data dependency and energy constraints. In [43], a global and local attention-based reinforcement learning approach for UAVs' cooperative behavior control is devised by researchers. Motion, coordination models, and constraints are analyzed-focusing on collision avoidance, motion updates, and task execution for multiple UAVs. This is abstracted as a multi-constraint decision-making problem, and a multi-agent reinforcement learning algorithm is crafted. Inspired by human learning, the design incorporates a global-and-local attention mechanism, enabling cooperative behavior control and effective coordination among UAVs.

In article [35] examines centralized user clustering to divide IoT users into various groups in accordance with users' priorities. The cluster holding the utmost priority is tasked with offloading computations to the edge server, whereas the cluster with the lowest priority performs computations locally. The importance of the new edge computing paradigm and its contribution to the development of the IoT drive research in this area, as mentioned in [36]. This study emphasizes the significance of edge computing in the IoT environment. This study examines, identifies, and documents current, ground-breaking advancements in edge computing from an IoT standpoint. It creates a taxonomy to categorize and organize the literature on edge computing and outlines the essential conditions for the successful implementation of edge computing in the IoT. In paper [37], to efficiently handle resource allocation for EN. In order to maximize the use of edge computing resources within the restrictions of the available budget, it provides resource bundles and services. The suggested paradigm ensures equity while sharing resources (wireless channels, communication) between different edge service providers and customers (or groups of users). In the research [38], the computation offloading mechanism was described as a stochastic game and analyzed how numerous selfish users allocate resources in IoT edge computing networks. In order to resolve the game using the suggested IL-based MA-Q method, a multi-agent reinforcement learning framework is created. Simulations show that the suggested IL-based MA-Q method is capable of solving the specified issue and is more energy-efficient without incurring additional costs for channel estimation at the centralized gateway. For the IoT edge computing system, according to the paper [39], a resource allocation policy was suggested to increase the effectiveness of resource consumption. The proposed guideline aims to minimize the combined value of the extended total of the mean task fulfillment duration and the mean count of resource appeals over time. To address this challenge, a strategy grounded in deep reinforcement learning is employed. An enhanced deep Q-network learning technique was put forward, where many replay memories are used to retain the experiences independently with little effect from one another. A brand-new framework, as stated in [40] built around markets was put out to effectively distribute the resources of heterogeneous, capacity-limited edge nodes to several competing services at the network edge. The suggested framework creates a market equilibrium solution by appropriately pricing the geographically dispersed services, maximizing the use of edge computing resources, and also allocating the best resource bundles to the services given the available budgets. Mentioned were two distributed methods that quickly reached market equilibrium.

III. SYSTEM MODEL

In this section, we present an optimization problem formulation and an explanation of the related restrictions for the IoT-to-Edge scheduling problem in the subsequent section.

A. EDGE NODES OPTIMIZATION PROBLEM

The operating cost and traffic cost functions are very important to edge nodes. Both of these functions are thoroughly addressed in this section in terms of the numerous limitations that must be taken into account.

TABLE 1. List of definition.

Symbol	Explanation
e	A single edge node
a	A single IoT device
r	A single resource type
E	Set of active Edge nodes
A	Set of active IoT device
R	Set of resource types; r = 1,2,3...,R
$O_c(e)$	Operational cost on a certain edge node
$T_c(e)$	Traffic cost on a certain edge node
$T(e)$	Overall cost on a certain edge node
s_e^r	Amount of resources of type r on a edge node e
s_e	Resource vector of edge node e
N_{ea}	Network latency between e and a
W_{ae}	Bandwidth capacity on the link ae
C_e	CPU cost on a certain edge node e
X_e	RAM cost on a certain edge node e
Φ_e	Maximal amount of traffic that e can afford
$AvOpr$	Operational needs on a certain e or a
$AvRes$	Available resource capacity on a certain e or a
B_{ae}	A binary decision to schedule a certain IoT task coming from a certain a on e
$M(O,T)$	Total operations and traffic cost of the setup
ω	A matching relation between two entities
$\omega(e)$	A matching scheme of a edge node
$\|\omega(a)\|^r$	Combined resources r of edge nodes matched to an a
$a_1 \succ_e a_2$	Certain e prefers being matched to a_1 rather than a_2
$L(a)$	The preference list of IoT device a
$L(e)$	The preference list of edge node e

1) OPERATIONAL COST

CPU and memory have an impact on the edge nodes' operating expenses. The CPU cost (measured in MIPS) accounts for both the active CPU consumption cost associated with an individual edge node's IoT operations and also the idle CPU consumption cost. The cost of idle memory while the node is not in use is included in the memory cost (i.e., RAM), which defines how much memory is utilized by the edge node to support IoT operations. The operational cost $O_c(e)$ of $e \in E$ is defined in technical terms as follows:

$$O_c(e) = RAM_e + CPU_e \tag{1}$$

2) TRAFFIC COST

The devices in an IoT ecosystem must transmit the data collected at specific bandwidth rates to the edge nodes. Various availability values for the active physical links may exist at various times, depending on the underlying demand. To send and receive important information, edge nodes also occasionally need to connect with one another. In order to determine the traffic cost of the edge nodes, the bandwidth cost associated with each link as well as the latency between the present hops are taken into account. Technically, the traffic cost $T_c(e)$ associated with an edge node $e \in E$, which corresponds to an IoT device $a \in A$, can be expressed as follows:

$$T_c(e) = \sum_{a,e \in A,E} W_{ae} \cdot B_{ae} + \sum_{e,e' \in E | e \neq e'} W_{ee'} \cdot N_{ea} \tag{2}$$

Therefore, each edge node $e \in E$ must reduce the subsequent objective function:

$$T(e) = O_c(e) + T_c(e) \tag{3}$$

Some restrictions should be taken into account in order to do this:

Condition 1: No edge node shall have more resources allocated to it overall compared to the available resource capacity.

$$\sum_{a \in A} s_a^r \cdot B_{ae} \leq s_e^r, \quad \forall_{e \in E}, \ \forall_{r \in R}, \ \forall_{a \in A} \tag{4}$$

Condition 2: Any edge node should only receive a certain amount of traffic at a time in order to avoid overloading it.

$$T_c(e) < \phi_e, \quad \forall_{e \in E} \tag{5}$$

B. IOT DEVICES OPTIMIZATION PROBLEM

The least amount of lag possible is what IoT devices are mostly focused on [33]. The calculation of traffic expenses for each IoT device involves considering the network's bandwidth expenses and latency between consecutive hops. The traffic cost $T_c(e)$ of an IoT device $a \in A$ engaging with a edge node $e \in E$ can be described formally by saying:

$$T_c(e) = \sum_{a,e \in A,E} W_{ae} + N_{ea} \tag{6}$$

The following restrictions must be taken into account in order to reduce this cost:

Condition 1: Only one edge node should be allotted to each Internet of Things task at a time, such as,.

$$\sum_{e \in E} B_{ae} = 1, \quad \forall_{a \in A} \tag{7}$$

Condition 2: An IoT device's overall traffic expenses must be lower than its traffic capacity.

$$T_c(a) < \phi_a, \quad \forall_{a \in A} \tag{8}$$

C. GENERALIZED OPTIMIZATION DIFFICULTY

We therefore establish the general optimization difficulty that we intend to minimize in this study on the basis of the optimization problems for edge nodes and IoT devices that were described in the sections above:

$$\begin{aligned} & M(O,T) = O_c(e) + T_c(e) + T_c(a) \\ & \min_{B_{ae}} M(O,T) \\ & s.t \sum_{a \in A} s_a^r \cdot B_{ae} \leq s_e^r, \quad \forall_{e \in E} \ \forall_{r \in R} \ \forall a \in A \\ & T_c(e) < \phi_e, \quad \forall_{e \in E} \\ & \sum_{e \in E} B_{ae} = 1, \quad \forall_{a \in A} \end{aligned} \tag{9}$$

$$B_{ae} = 0, 1, \quad \forall_{a \in A} \ \forall_{e \in E} \tag{10}$$

IV. FORMULATION

In this section, we go over the main concepts of matching game theory, discuss the preference functions of edge nodes and IoT devices, and then demonstrate how to create preference lists for both entities in real-world scenarios.

A. MATCHING GAME CONCEPTS

Concept 1: The matching relation(ω) is the result of the IoT-to-Edge scheduling, where ω is a function such that $A \cup E \rightarrow 2^{A \cup E}$ satisfying the following requirements:

- $\omega(e) \subseteq A$ such that $|\omega(e)|^r \leq s_e'^r$.
- $\omega(a) \subseteq E$ such that $|\omega(a)|^r \leq s_e^r$ or $|\omega|_a = 0, \forall_{a \in A}, e \in E$ and $r \in R$, Where $|\omega|_{(a)} = 0$ demonstrates that a is not allotted.
- $a \in \omega(e)$ if $\iff \omega(a) = e, \forall_{a \in A}, \forall_{e \in E}$

Concept 2: A matching ω is considered blocked by an IoT-Edge pair (a, e) when there exists a pair if (a, e) where $a \in \omega(e)$ and $e \in \omega(a)$. Additionally, a must be preferred over $\omega(e)$ and e must be preferred over $\omega(a)$.

Concept 3: If a particular edge node e is using all of its resource capacity, it is said to be saturated. If a node still has some resources available, it will take any IoT job a as long as $s_e^r \leftarrow s_a^r, \forall_{r \in R}$.

Concept 4: When (1) Every IoT device a's is paired with an edge node and (2) there are not any blocking pairings, a matching relation ω is considered stable.

B. AN OPTIMAL FUNCTION FOR IOT DEVICES SELECTION

IoT devices have a natural inclination to assign tasks to edge nodes that achieve two objectives: firstly, they minimize the expense of task-related data transfer, and secondly, they capitalize on the largest pool of available resources. The overarching aim is to expedite job completion timelines. A transitive, comprehensive, and stringent preference relation $L(a)$ exists between each IoT device $a \in A$ and the set E of edge nodes. When there is a preference relation $e \succ_a e'$, a favors edge node e over edge node e'. Additionally, if an IoT device a is undecided about whether to join a edge node e or remain alone, a edge node e is said to be unacceptable to the device a. The provided explanation establishes the preference function of an IoT device in the following manner [23].

$$e_1 \succ_a e_2' \iff L_a(e_1) > L_a(e_2) \tag{11}$$

where (12), as shown at the bottom of the next page.

C. EDGE NODE PREFERENCE FUNCTION

In accordance with the consolidation policy of the edge nodes, the preferred list of edge nodes is created. An edge node particularly desires to increase the efficiency of its resource use by hosting the greatest number of IoT devices and making the best use of its resources. This suggests that edge nodes prefer IoT tasks to non-tasks, and vice versa. A well-defined preference relation $L_e(A)$ exists for each edge node $e \in E$ in relation to the set of IoT devices, establishing a transitive, comprehensive, and rigorous comparison. In this context, the notation $a1 \succ_e a2$ signifies that edge node e prioritizes receiving tasks from IoT device $a1$ over IoT device $a2$. Additionally, if a edge node e chooses to remain unmatched over being matched to an IoT device a, the IoT device is considered to be unacceptable to a(i.e., $0 \succ_e a$ then). Building upon this idea, the preference mechanism of an edge node can be formally described as stated in reference [29].

D. GENERATING A LIST OF PREFERRED IoT DEVICES

We outline how the preference list for each IoT device may be truly built using the criteria given in Algorithm 1 (is used by every IoT device) The method takes a set of edges that can perform IoT activities as input and generates a preference list L_a for each IoT device a that contains the edge nodes sorted by a according to how much they are preferred by that device. The procedure starts by exploring unvisited edge nodes (Line 4). Subsequently, it evaluates the compatibility between available resources ($AvRes$) and task requirements ($AvOpr$) (Line 5). The algorithm assesses latency with the underlying IoT device, recording results (Lines 6-8) for edges having sufficient resources. In cases of inadequate resources, the edge is removed, and the next unvisited one is considered (Lines 6-8). For assisting the IoT device in deciding its preferred order among the retained edges, the algorithm employs Eq. (12).

E. GENERATING A LIST OF PREFERRED EDGE NODE

By attempting to host as many IoT devices as they can, edge nodes primarily aim to increase their resource consumption.

Algorithm 1 Generating a List of Preferred IoT Devices a's

1: **Input**: Create a collection labeled E comprising operational edge nodes, with the initial node denoted as e_0.
2: **Output**: Preference list $L(a)$ of IoT device a
3: **while** there are still some unexplored edges in E **do**
4: Mark e_0 as visited after choosing it
5: **if** $AvRes(e_0) \leftarrow AvOpr(a)$, i.e., Resources are available for e_0 to complete the tasks **then**
6: Verify and note e_0's delay
7: **end if**
8: Go to the following unexplored edge in E
9: **end while**
10: Ranking the edge nodes using Eq. (12) and storing the results in $L(a)$.

Algorithm 2 Preference List Creation for Edge Node e

1: **Input**: Enumerate a collection of Internet of Things (IoT) A devices aiming to organize their functions on edge nodes.
2: **Output**: Preference list $L(e)$ of edge node e
3: In accordance with the IoT device's preference degree, sort the IoT devices using Eq. (14) and store the results in $L(e)$.
4: **while** $L(e)$ is not empty **do**
5: Get a_0 the head of $L(e)$
6: **if** $AvRes(e) < AvOpr(a_0)$ **then**
7: remove a_0 from $L(e)$
8: **else**
9: $a_0 = a_0$·next
10: **end if**
11: **end while**

This demonstrates that edge nodes favor more IoT device assignments over fewer or inadequate ones. In technique 2, we suggest a heuristic technique to assist edge nodes in creating their preference lists. The goal is to add the subsequent IoT device to the preference function at each stage that leaves the least amount of space after being attached to the underlying edge node.

The algorithm takes input about a collection of IoT devices to serve and generates a preference list for each edge node. This list ranks IoT devices based on how much the edge node prefers them, giving priority to devices that use fewer resources when assigned to the node. To find the preference ordering among the collection of potential edge nodes (Line 3), the program first applies Eq. This preference formulation prioritizes IoT devices that utilize minimal resources when allocated to edge nodes. The concept of sequential task execution is not invoked in this context. Consequently, We refrain from indicating that the edge node prioritizes any specific IoT task over others based solely on resource consumption. Instead, we discuss a strategy for task consolidation. Subsequently, the algorithm assesses whether the resource demand of each IoT device in the preference sequence, denoted as *AvOpr*, is less than or equal to the available resources on the executing edge node, represented as *AvRes* (Line 6). If an IoT task's resource requirements exceed the edge node's available resources, that respective IoT device is excluded from the list of favored devices (Line 7); The IoT device is retained in any other case. Follow this procedure until all IoT devices on the preference list have their resource requirements met.

V. EFFICIENT IoT-TO-EDGE SCHEDULING STRATEGY: A MULTI-TO-SINGLE MATCHING GAME

The pair of methods we propose are decentralized and centralized. It is important to note that the procedure for pairing IoT devices with edge nodes follows the approach outlined in reference [36]. The simultaneous deployment of Virtual Network Functions (VNFs) within substrate networks is the main emphasis of this study. In conducting our experiments, each scenario was executed 50 times to ensure consistency and reliability. We utilized a standardized hardware setup with processor Intel Core i7 running at a speed of 3.8 GHz and 32 GB of RAM specifications and employed 32-bit Windows operating system for all simulations.

A. CENTRALIZED MATCHING ALGORITHM

The group of edge nodes that are currently up and running and the group of IoT devices looking to organize their duties are used as inputs for the centralized scheduling approach. Each IoT job is embedded to a edge node as the output. Centralized matching algorithm is given in Algorithm 3. The process commences by evaluating unutilized edge nodes at saturation point (Line 4) and unassigned IoT devices (Line 5). If applicable, the edge node with the highest priority for the IoT device is chosen along with the first unassigned IoT device (Line 6). Line 7 Subsequently, the algorithm checks if the chosen edge node possesses adequate resources for the IoT device's requirements. If affirmative, the IoT device is scheduled to the fog node, decreasing its available resources accordingly (Line 9). However, if the edge node's resources are insufficient (Line 12), the IoT device is declined. To optimize further, the algorithm discards IoT devices with lower priority than the refused one (Line 13), diminishing complexity. To update preference lists considering changing edge node resources, emerging nodes, and departing nodes (Line 18), we repeat algorithms 1 and 2 before cyclically reiterating the entire process at fixed intervals (Line 19).

$$a_1 \succ_e e'_2 \iff L_e(a_1) > L_e(a_2) \tag{13}$$

where (14), as shown at the bottom of the next page.

B. ALGORITHMS FOR DISTRIBUTED MATCHING

As the IoT devices and edge nodes interact straight to perform the matching, the distributed execution of our approach does not require a central entity.

Every IoT device executes Algorithm 4, which produces a matching scheme between the edge and IoT devices after receiving as an input the underlying IoT device's preference list. To accomplish this, the procedure initially goes through the IoT device's set of preferences (Step 4), followed by the selection of the preferred edge node, which is the top-ranked one (Step 5). Subsequently, the IoT device transmits an *proposal* message to the chosen edge node (Step 6) and patiently anticipates a reply (Step 7).

The lowest-ranked edge on the IoT preference list is selected first. If this edge declines hosting the IoT tasks, the next highest-preference edge is chosen (Line 8). In case of a negative response, the process repeats until a willing edge is found, facilitating the matching between the IoT device and an edge (Line 11). This cycle repeats periodically (Line 15). Prior to this, Algorithm 1 is executed to adjust preference lists due to evolving edge resources, new additions, and departures, impacting the preference functions of both sides (Line 14).

Each individual edge node executes Algorithm 5, generating a matching arrangement between edge and IoT devices. The algorithm takes a queue of IoT devices that have

$$L_a(e) = \begin{cases} +\infty, \text{If } e \text{ provides the lowest traffic cost and the most resources to do the jobs,} \\ 0, \text{If } e \text{ proposals the lowest traffic cost or has the most resources available to complete the jobs;} \\ -\infty, \text{Otherwise} \end{cases} \tag{12}$$

Algorithm 3 Algorithm for Centralized Matching

1: **Input**: For IoT devices, set A, and for edge nodes, set E
2: **Output**: IoT devices embedded on edge nodes
3: **repeat**
4: **while** There are non-standard $\exists e \in E$ **do**
5: **while** Unmatched devices $\exists a \in A$ **do**
6: $a \leftarrow$ as the head of $L(e)$
7: **if** $s_e^r > s_a^r$ **then**
8: Match a to e
9: $s_e^r = s_e^r - s_a^p$
10: Remove a from $L(e)$
11: **else**
12: Decline a
13: Decline every a' in order to have $a \succ_e a'$
14: Proceed to the following edge node e
15: **end if**
16: **end while**
17: **end while**
18: **simulate** algorithms 1 and 2
19: **until** $\in$ elapses

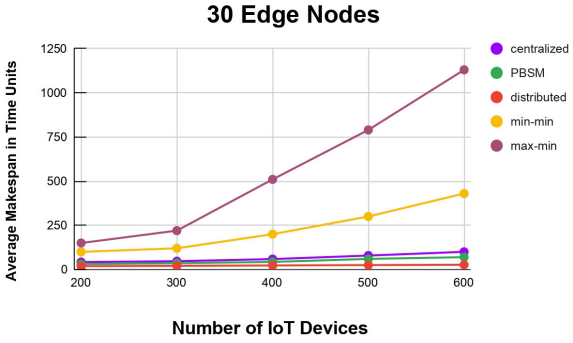

FIGURE 2. 30 Edge nodes.

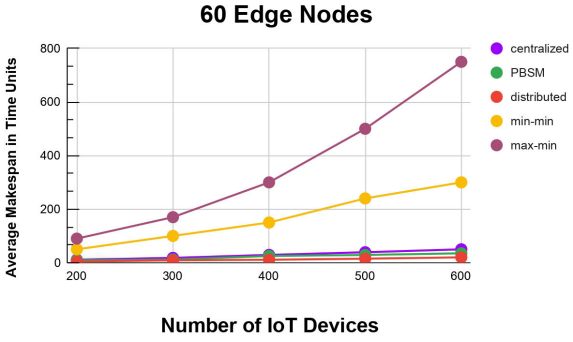

FIGURE 3. 60 Edge nodes.

submitted proposal messages to the respective edge node running Algorithm 4 as input.

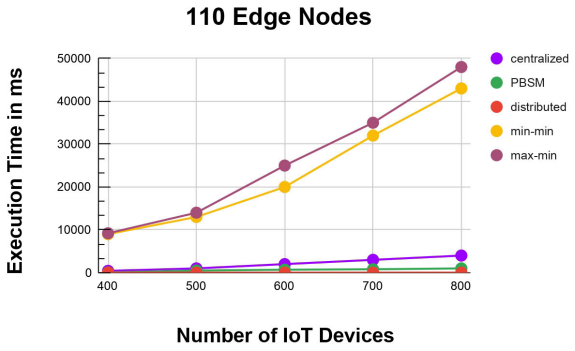

FIGURE 4. 110 Edge nodes.

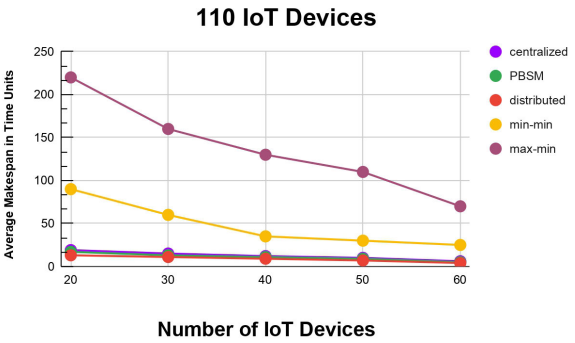

FIGURE 5. 110 IoT devices.

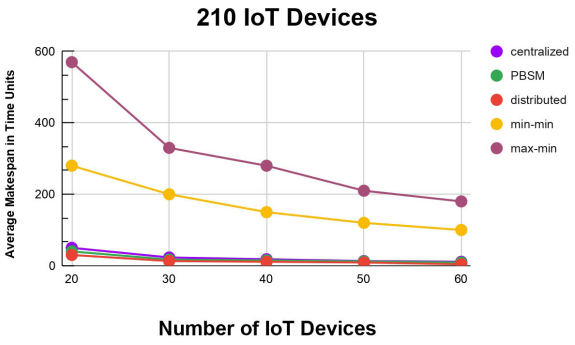

FIGURE 6. 210 IoT devices.

In order to determine if the input queue is empty or not, the algorithm first cycles through it (Line 4). A nonempty queue indicates that the underlying edge node has not yet checked any proposal messages from some IoT devices. If the queue has entries, the algorithm selects the most recent proposal message to evaluate whether the edge node possesses adequate resources for executing tasks produced by the Internet of Things device that sent the message. It also determines if the edge node is on the node's preferred list (Line 6). If these conditions are fulfilled, the edge node

$$L_e(a) = \begin{cases} +\infty, \text{ if the completion of } a's \text{ task consumes the least amount of space on the edge node} \\ -\infty, \text{ Otherwise} \end{cases} \quad (14)$$

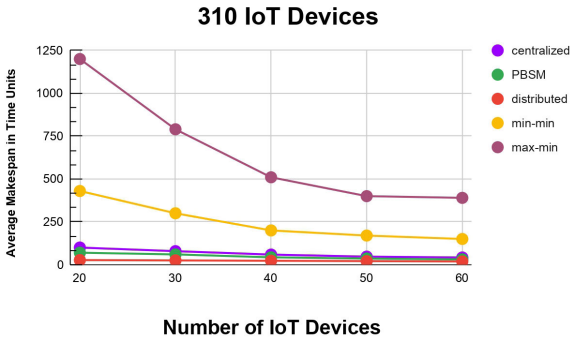

FIGURE 7. 310 IoT devices.

Algorithm 4 Distributed Matching Algorithm in IoT

1: **Input**: The number of choice $L(a)$ made by IoT device a
2: **Output**: Identification of coordinating edge and relevant IoT devices
3: **repeat**
4: **while** $L(a)$ is not free **do**
5: $e \leftarrow$ first item in $L(a)$
6: Dispatch proposal message proposal[e] to e
7: Await response $R[proposal[e]]$
8: **if** $R[proposal[e]] =$ "$decline''$ **then**
9: Move e to the last of $L(a)$
10: **else if** $R[proposal[e]] =$ "$accept''$ **then**
11: Conclude the process
12: **end if**
13: **end while**
14: **execute** algorithms 1
15: **until** $\in$ elapses

responds to the IoT device with an acceptance message, confirming its readiness to perform the tasks. Subsequently, the edge node adjusts its available resource quantity by subtracting the necessary amount required for executing the IoT activities (Line 7). The IoT device receives a decline message informing it that the edge cannot currently fulfill its tasks (Line 9), and the edge declines all other IoT devices whose position in the preference list is lower compared to that of the declined IoT device (Line 10). Conversely, this situation arises when the resources on the edge node are inadequate for the tasks of the IoT, or if the IoT device is not prioritized highly on the edge node's preference list. The underlying IoT devices are taken out of the queue in lines 11-12. The entire process is carried out on a regular basis after a predetermined amount of time (Line 16), but right before that, Algorithm 2 is again executed to obtain new updated preference lists according to the dynamic nature of the edge node resources, the emergence of new edge nodes and IoT devices, and the departure of current ones, all of which have an impact on the preference functions of both parties (Line 15).

The parameters for the PBSM comprise: the collection of n operational edge nodes denoted as E, the set of n active IoT

Algorithm 5 Algorithms for Distributed Matching - Edge Side

1: **Input**: IoT devices are submitting proposal messages to e in $Q(a)$ queue.
2: **Output**: IoT device mapping to a specific node e
3: **repeat**
4: **while** $Q(a)$ is not empty **do**
5: **if** $s_e^r > s_a^r a \in Q(a)$ **then**
6: Deliver i a response with $accept$, i.e., $R[proposal[f]] =$ "$accept''$
7: Change the e resource so that $s_e^r = s_e^r - s_a^r$
8: **else**
9: Deliver a a response with $decline$, i.e.,$R[proposal[f]] =$ "$decline''$
10: Decline all a' in a way that $a \succ_e a'$
11: Take a out of $Q(a)$
12: Take a' out of $Q(a)$
13: **end if**
14: **end while**
15: **execute** algorithms 2
16: **until** $\in$ elapses

devices labeled as A, the distinct order of preferences for the n edge nodes represented as $\succ_e = (\succ_{e1}, \succ_{e2}, \ldots, \succ_{en})$, and the unique preference order for the n IoT devices denoted as $\succ_a = (\succ_{a1}, \succ_{a2}, \ldots, \succ_{an})$.

The PBSM's outcome consists of assigned pairs of IoT devices and Edge nodes. Initially in Algorithm 6, the setup ensures no engagement between IoT devices and Edge nodes (line 2-7). The A' structure in line 2 manages unassigned IoT devices. Moving to the allocation phase, the *while* loop in line 4 monitors Edge node assignments to IoT devices. The structure of the PBSM guarantees termination of the *while* loop when the output, Δ encompasses all IoT device-Edge node pairs. In line 10, the function labeled *rand_pick()* is employed to randomly select an IoT device from the set A'. The subsequent function, denoted as *extract()* and located in line 11, serves the purpose of retrieving the most favored edge node based on the preference profile a^* for each IoT device. This information is then stored within the data structure referred to as e^*. The condition presented in line 12 is evaluated, and if it is met, the IoT device denoted as a^* is inserted at the index of e^* using line 13. Once an IoT device represented by a^* is incorporated into Δ, it is subsequently removed from the set A^*. However, if the assessment in line 12 is unsuccessful, it indicates that a specific device e^* is contending with multiple proposals. To address the competitive landscape among IoT devices, a decision is taken in step 17 to single out an IoT device from the available options. Resolving conflicts involves utilizing the precise preference sequence of the prime edge nodes (e^*) and retaining the most favored IoT device among the proposals in $\Delta[e^*]$. The 19th step involves eliminating the assigned IoT device from set A' while incorporating declined

Algorithm 6 Preference Based Stable Matching Algorithm

Input: $A = a_1, a_2, \ldots, a_n$; $E = e_1, e_2, \ldots, e_n$; $\succ_a = (\succ_{a1}, \succ_{a2}, \ldots, \succ_{an})$; $\succ_e = (\succ_{e1}, \succ_{e2}, \ldots, \succ_{en})$.
Output: $\Delta \leftarrow \phi$

1: begin
/* Initialization Phase* /
2: $A' = A$
3: **for** each $i \in A$ **do**
4: $\Delta[i] \leftarrow \phi$ ▷ Initially, no IoT device is assigned a edge node
5: **end for**
6: **for** each $i \in E$ **do**
7: $\tilde{\Delta}[i] \leftarrow \phi$ ▷ Initially, no edge node is assigned a IoT device
8: **end for**
/* Allocation phase */
9: **while** $A' \neq \phi$ **do**
10: $a^* \leftarrow rand\_pick(A')$
11: $e^* \leftarrow extract(\succ_a^*)$
12: **if** $\tilde{\Delta}[e^*] == \phi$ **then**
13: $\tilde{\Delta}[e^*] \leftarrow a^*$
14: $A' \leftarrow A'\, a^*$
15: **else**
16: $a^* \succ \tilde{\Delta}[e^*]$
17: **if** $a^* \succ_{e^*} a^*$ **then**
18: $\Delta[e^*] \leftarrow a^*$
19: $A' \succ A'\, a^*$
20: $A' \succ A' \cup a^*$
21: **end if**
22: **end if**
23: **end while**
24: **return** Δ
25: **end**

IoT devices. Ultimately, line 24 of the PBSM function yields the conclusive pairings of IoT devices and Edge nodes within the system.

VI. FINDINGS FROM OBSERVATIONS AND ANALYSIS

In this part, we first explain the setting in which our simulations were run, followed by a discussion of the outcomes.

A. EXPERIMENTAL SETUP

We take into account Remote Patient Monitoring using IoT-based implantable or wearable sensors for the simulations. The experiment replicates an IoT sensor that continuously sends patient physiological characteristics, including temperature, blood pressure, and pacing, to the edges for analysis. We create our own scheduling application to run the simulations, and each edge node is given a CPU with a capacity between [3, 000, 9, 000] MIPS, a RAM with a capacity between [6, 000, 25, 000] MB, and a link bandwidth with a capacity between [9, 000, 25, 000] Mbps.

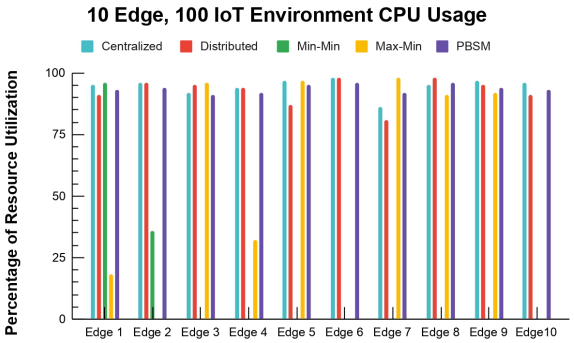

FIGURE 8. CPU utilization

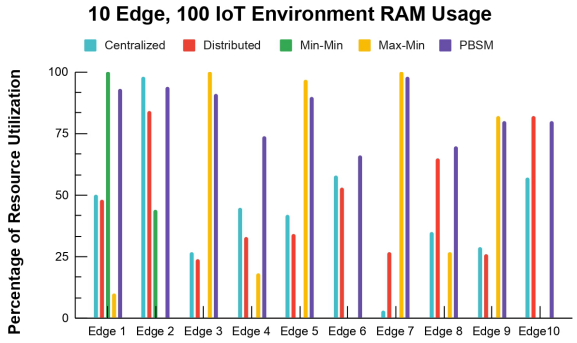

FIGURE 9. RAM utilization.

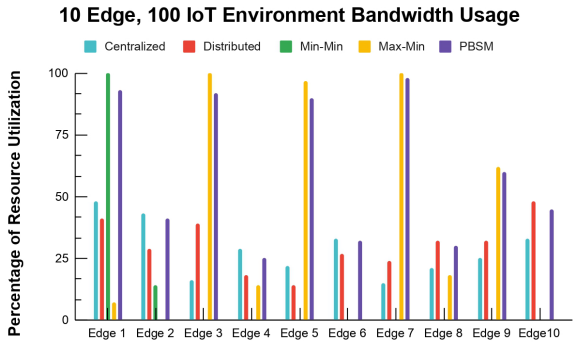

FIGURE 10. Bandwidth utilization.

But compared to edge nodes, IoT devices use far fewer resources. In particular, the IoT devices' CPU capacity is selected from a range of [400, 800] MIPS, their RAM capacity is selected from a range of [500, 1000] MB, and their link bandwidth capacity ranges from [600, 1000] Mbps.

Every set of edge nodes and IoT devices has a delay that varies between 200 and 6000 milliseconds. Since MaxMin [35] and Min-Min [35], [40] two widely used scheduling algorithms, are utilized often in the scheduling literature in many relevant fields, including fog, edge, and cloud computing, we compare our solution with theirs. By scheduling the quickest-finishing jobs first, the Min-Min approach favors them. When there are more large-sized activities than smaller-sized tasks, Min-Min's performance begins to suffer, resulting in inefficient resource use and

prolonged makespan. On the other hand, the basic notion of Max-Min is to prioritize the jobs with the longest execution times in order to reduce the overall makespan. Our approach differs from current alternatives by encompassing a broader spectrum of metrics. These metrics include factors like latency, CPU, RAM, and bandwidth, considered from both IoT devices and edge nodes perspectives. This sets our solution apart from the MinMin and Max-Min algorithms, that alone focus on the edge nodes aspect.

B. RESULTS FROM THE EXPERIMENT

Our simulations primarily explore three key aspects: the average amount of time required to complete an Internet of Things job, how efficiently edge nodes use their resources, and how long the solution takes to execute altogether.

We rigorously assess each indicator under a number of realistic circumstances in order to thoroughly examine the scalability of our system over a wide variety of real-world scenarios.

1) AVERAGE MAKESPAN

We commence by studying the time taken by different solutions for task completion. It is worth noting that this is the duration between the initiation and completion of a task. In the initial trials, the number of fog nodes remains constant, while we vary the quantity of IoT devices Figures 2, 3, 4. This investigation aims to reveal how the makespan is influenced by the number of IoT devices generating tasks. Our experiments encompass IoT device quantities ranging from 200 to 600, while the number of fog nodes is consistent Figures 2, 3, 4.

The initial finding from figures 2, 3, 4 is that, in contrast to the Max-Min and Min-Min techniques, our two suggested algorithms produce low makespan. Furthermore, in each of the instances, our distributed and PBSM approaches performed better than the centralized algorithm. The key benefit of the distributed version is that it allows edge nodes and IoT devices to simultaneously and independently make judgments based on their preference lists by decentralizing the matching process. The Min-Min and Max-Min approaches exhibit subpar performance in contrast to our solution. This is because they solely consider task completion time, disregarding vital factors such as resource usage and latency. These factors profoundly influence makespan. Another key insight from Figures 2, 3, 4 is that makespan rises with the growing count of IoT devices. This is due to the fact that having more IoT devices will result in a higher number of tasks being assigned to each individual edge node when compared to having a set number of edge nodes. This inevitably lengthens the time jobs must wait before beginning, increasing the entire timeline.

In Figures 5, 6, 7, we measure the makespan of the IoT jobs while also controlling the number of edge nodes that serve them and the number of IoT devices that are responsible for their creation. IoT devices in mentioned diagram is maintained at 110 Figure 5, 210 Figure 6 and 310 Figure 7.

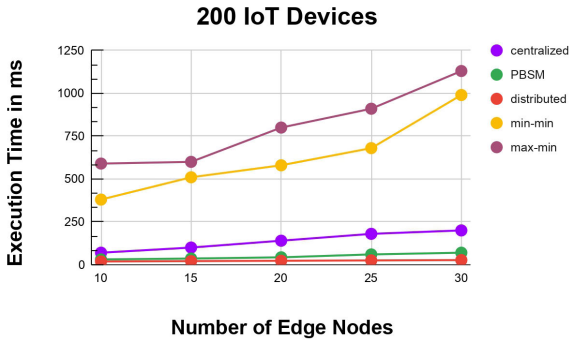

FIGURE 11. 200 IoT devices alongside diverse edge nodes.

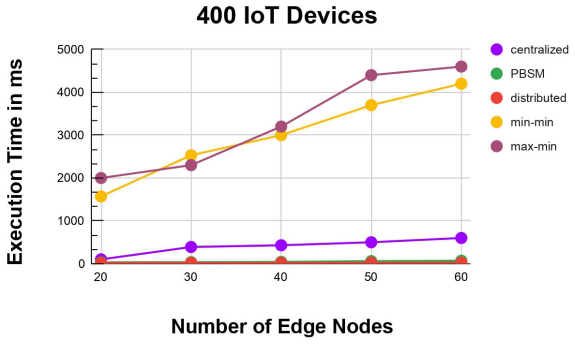

FIGURE 12. 400 IoT devices alongside diverse edge nodes.

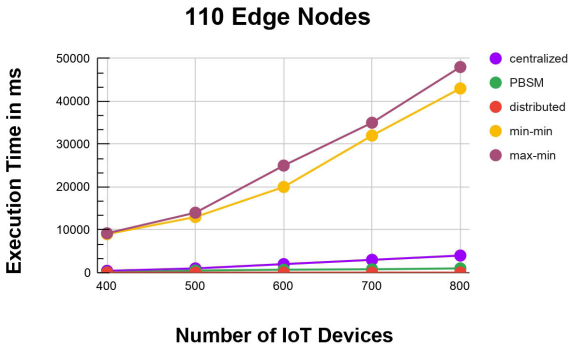

FIGURE 13. 110 edge nodes alongside diverse IoT devices.

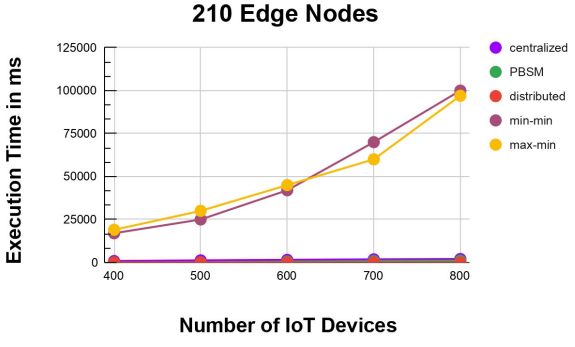

FIGURE 14. 210 edge nodes alongside diverse IoT devices.

The results indicate that expanding the edge node count for a fixed IoT job set leads to a decrease in makespan.

This is attributed to the higher availability of edge nodes, reducing waiting times and task completion duration. The outcomes depicted in Figures 5, 6, 7 highlight that our distributed, PBSM, and centralized algorithms outperform the Min-Min and Min-Max techniques in terms of makespan. Moreover, the distributed approach slightly surpasses the centralized method when dealing with varying IoT task volumes. Overall, our approach demonstrates superior makespan reduction and scalability compared to Min-Min and Min-Max methods, especially with the growth of IoT devices and edge nodes.

2) RESOURCE UTILIZATION

We evaluate resource consumption on edge nodes in various studies where these nodes are assisting IoT tasks. The utilization of CPU, RAM, and link bandwidth is quantified. Resource consumption is crucial since it helps us determine how much money each of the compared techniques will cost edge providers. In order to investigate the scalability of the various investigated methodologies, we run four distinct experiments for this set of tests, modifying the quantity of IoT devices and edge nodes in different configurations.

We examine the CPU, RAM, and bandwidth usage in Figures 8, 9, 10 for a system with 10 edge nodes and 100 IoT devices. Further tests are also conducted in this section.

From the provided data, we can infer that as the number of edge nodes increases while keeping the IoT jobs constant, there is a noticeable reduction in resource consumption per edge node. This aligns with expectations, as a higher quantity of edge nodes distributing the same workload leads to a lighter load on each node. Additionally, the study indicates that both of our algorithms consistently achieve near maximum resource utilization across different combinations of IoT and edge device quantities.

Nevertheless, the resource utilization of the Min-Min and Max-Min algorithms exhibits variability, as specific edge nodes experience periods of high utilization while others remain partially or entirely idle. In contrast, our proposed solution's preference function considers the resource consolidation approach of individual edge nodes. This enables these nodes to optimize resource usage effectively, minimizing the necessity for deploying extra edge nodes to cater to IoT devices. As a result, this approach proposals the benefit of lightening edge providers' financial responsibilities.

3) EXECUTION TIME

The execution time required by the various solutions under consideration is the third measure we take into account in our investigation. To achieve this, we design four distinct trials using various combinations of edge node and IoT device numbers. We adjust the number of edge nodes in Figure 11 from 10 to 30 and a certain amount of IoT devices at 200. In Figure 12, we deploy 400 IoT devices at a fixed number while varying the number of edge nodes from 20 to 60. In Figure 13, we increase the number of IoT devices from 400 to 800 while fixing the number of edge nodes at 100. In Figure 14, we modify the number of IoT devices The execution time required by the various solutions under consideration is the third measure we take into account in our investigation. The initial inference from the data indicates that an escalation in the count of IoT devices across several examined methodologies corresponds to a proportional increase in execution time. This pattern emerges to provide broader task allocation capabilities to edge nodes via diverse algorithms. Furthermore, with a surge in the quantity of edge nodes, there is also a corresponding elevation in the run time of the diverse algorithms. This phenomenon is attributable to the expanded inputs that IoT devices require to take certain inputs into account when formulating their preference lists in the presence of additional edge nodes. These findings also highlight that our decentralized and centralized algorithms exhibit shorter execution times compared to the Min-Min and Max-Min methodologies. In comparison to the centralized variant, our distributed solution demonstrates improved execution speed. The basis for this is that in the distributed version of our technology, edge nodes and IoT devices can connect with one another directly without the need for a middleman. This helps to reduce the total communication overhead between the various nodes.

4) CONTRAST AGAINST THE IDEAL OUTCOME

We contrast our solution's performance with that of the best option. The appendix contains the comparison's findings.

VII. CONCLUSION

In this research, we employed game theory to tackle the issue of scheduling time-sensitive IoT services in edge computing environments. Our strategy comprises of two main parts: IoT devices and edge nodes may be evaluated by each other by: (1) including preference functions, and (2) using both centralized and distributed intelligent matching algorithms. as well as important indicators like latency and resource utilization, to facilitate the allocation of IoT services to suitable edge nodes, considering the preferences of all parties involved. The key benefit of our approach over the state-of-the-art is that we generate scheduling decisions while taking the preferences and limits of both IoT devices and edge nodes into account.

We evaluate how well our method performs in comparison to two widely used scheduling methods, Min-Min and Max-Min, used in cloud and edge computing contexts. The results show that our method works better than Min-Min and Max-Min by achieving up to a 30% to 40% increase in efficiency for resource utilization on edge nodes. Additionally, it reduces the execution time of IoT services by a factor of 2 to 9. Our solution also boasts faster execution times and enhanced scalability. In our future work, we will focus on an auction-based approach for better utilization of resources in a monetary environment.

REFERENCES

[1] K. Bousselmi, Z. Brahmi, and M. M. Gammoudi, "QoS-aware scheduling of workflows in cloud computing environments," in *Proc. IEEE 30th Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Mar. 2016, pp. 737–745.

[2] E. Elghoneimy, O. Bouhali, and H. Alnuweiri, "Resource allocation and scheduling in cloud computing," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Jan. 2012, pp. 309–314.

[3] O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing Amazon EC2 spot instance pricing," *ACM Trans. Econ. Comput.*, vol. 1, no. 3, pp. 1–20, Sep. 2013.

[4] A. Gharaibeh, A. Khreishah, M. Mohammadi, A. Al-Fuqaha, I. Khalil, and A. Rayes, "Online auction of cloud resources in support of the Internet of Things," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1583–1596, Oct. 2017.

[5] A. Bandyopadhyay, S. Misra, D. Nath, and V. Sarkar, "Automatic smart parking one-sided free slot booking using Internet of Things (IoT)," in *Advances in Medical Physics and Healthcare Engineering* (Lecture Notes in Bioengineering). Singapore: Springer, 2021, pp. 463–472, doi: 10.1007/978-981-33-6915-3_46.

[6] D. Evans, "The Internet of Things: How the next evolution of the internet is changing everything," in *Cisco Internet Business Solutions Group*, 2011, pp. 1–11.

[7] T. Choudhari, M. Moh, and T.-S. Moh, "Prioritized task scheduling in fog computing," in *Proc. ACMSE Conf.*, Mar. 2018, p. 22.

[8] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Optimal load distribution for the detection of VM-based DDoS attacks in the cloud," *IEEE Trans. Services Comput.*, vol. 13, no. 1, pp. 114–129, Jan. 2020.

[9] T. D. Braun, H. J. Siegal, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen, and R. F. Freund, "A comparison study of static mapping heuristics for a class of meta-tasks on heterogeneous computing systems," in *Proc. 8th Heterogeneous Comput. Workshop*, 1999, pp. 15–29.

[10] K. R. Shetti, S. A. Fahmy, and T. Bretschneider, "Optimization of the HEFT algorithm for a CPU-GPU environment," in *Proc. Int. Conf. Parallel Distrib. Comput., Appl. Technol.*, Dec. 2013, pp. 212–218.

[11] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Technol. Sci.*, Nov. 2011, pp. 320–327.

[12] A. Bandyopadhyay, S. Mukhopadhyay, and U. Ganguly, "Allocating resources in cloud computing when users have strict preferences," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2016, pp. 2324–2328.

[13] K. Kaur, S. Garg, G. Kaddoum, E. Bou-Harb, and K. R. Choo, "A big data-enabled consolidated framework for energy efficient software defined data centers in IoT setups," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2687–2697, Apr. 2020.

[14] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 4th Quart., 2018.

[15] S. Garg, K. Kaur, S. Batra, G. Kaddoum, N. Kumar, and A. Boukerche, "A multi-stage anomaly detection scheme for augmenting the security in IoT-enabled applications," *Future Gener. Comput. Syst.*, vol. 104, pp. 105–118, Mar. 2020.

[16] A. Bandyopadhyay, V. Kumar Singh, S. Mukhopadhyay, U. Rai, F. Xhafa, and P. Krause, "Matching IoT devices to the fog service providers: A mechanism design perspective," *Sensors*, vol. 20, no. 23, p. 6761, Nov. 2020.

[17] A. Bandyopadhyay, T. S. Roy, V. Sarkar, and S. Mallik, "Combinatorial auction-based fog service allocation mechanism for IoT applications," in *Proc. 10th Int. Conf. Cloud Comput., Data Sci. Eng.*, Jan. 2020, pp. 518–524.

[18] A. Sinha, V. Mishra, A. Bandyopadhyay, S. Swain, and S. Chakraborty, "Fair resource allocation in fog computing by using a game theoretic approach," in *Proc. Int. Conf. Data Anal. Insights*, 2023, pp. 125–134.

[19] A. Bandyopadhyay, F. Xhafa, S. Mallik, P. Krause, S. Mukhopadhyay, V. K. Singh, and U. Maulik, "A framework for allocation of IoT devices to the fog service providers in strategic setting," in *Proc. Int. Conf. P2P, Parallel, Grid, Cloud Internet Computing.*, 2020, pp. 340–351.

[20] H. Sami and A. Mourad, "Dynamic on-demand fog formation offering on-the-fly IoT service deployment," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 1026–1039, Jul. 2020.

[21] J. Santos, T. Wauters, B. Volckaert, and F. De Turck, "Fog computing: Enabling the management and orchestration of smart city applications in 5G networks," *Entropy*, vol. 20, no. 1, p. 4, Dec. 2017.

[22] H. Tout, C. Talhi, N. Kara, and A. Mourad, "Selective mobile cloud offloading to augment multi-persona performance and viability," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 314–328, Apr. 2019.

[23] T. Dbouk, A. Mourad, H. Otrok, H. Tout, and C. Talhi, "A novel ad-hoc mobile edge cloud offering security services through intelligent resource-aware offloading," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 4, pp. 1665–1680, Dec. 2019.

[24] L. Liu, D. Qi, N. Zhou, and Y. Wu, "A task scheduling algorithm based on classification mining in fog computing environment," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–11, Aug. 2018.

[25] N. Bessis and C. Dobre, *Big Data and Internet of Things: A Roadmap for Smart Environments*. Cham, Switzerland: Springer, 2014.

[26] H. Otrok, A. Mourad, J.-M. Robert, N. Moati, and H. Sanadiki, "A cluster-based model for QoS-OLSR protocol," in *Proc. 7th Int. Wireless Commun. Mobile Comput. Conf.*, Jul. 2011, pp. 1099–1104.

[27] 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.

[28] P. G. Lopez, A. Montresor, D. J. Epema, A. Datta, T. H. Higashino, A. L. Iamnitchi, and M. Barcellos, "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, 2015.

[29] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game," *IEEE Trans. Services Comput.*, vol. 11, no. 1, pp. 184–201, Jan. 2018.

[30] B. Yang, Z. Li, S. Chen, T. Wang, and K. Li, "Stackelberg game approach for energy-aware resource allocation in data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 12, pp. 3646–3658, Dec. 2016.

[31] N. Moati, H. Otrok, A. Mourad, and J.-M. Robert, "Reputation-based cooperative detection model of selfish nodes in cluster-based QoS-OLSR protocol," *Wireless Pers. Commun.*, vol. 75, no. 3, pp. 1747–1768, Apr. 2014.

[32] X.-Q. Pham, N. D. Man, N. D. T. Tri, N. Q. Thai, and E.-N. Huh, "A cost- and performance-effective approach for task scheduling based on collaboration between cloud and fog computing," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 11, Nov. 2017, Art. no. 155014771774207.

[33] X.-Q. Pham and E.-N. Huh, "Towards task scheduling in a cloud-fog computing system," in *Proc. 18th Asia–Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Oct. 2016, pp. 1–4.

[34] S. Torabi, E. Bou-Harb, C. Assi, E. B. Karbab, A. Boukhtouta, and M. Debbabi, "Inferring and investigating IoT-generated scanning campaigns targeting a large network telescope," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 402–418, Jan. 2022.

[35] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "*Zenith*: Utility-aware resource allocation for edge computing," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Jun. 2017, pp. 47–54.

[36] 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.

[37] N. Hassan, S. Gillani, E. Ahmed, I. Yaqoob, and M. Imran, "The role of edge computing in Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 110–115, Nov. 2018.

[38] D. T. Nguyen, L. B. Le, and V. Bhargava, "Price-based resource allocation for edge computing: A market equilibrium approach," *IEEE Trans. Cloud Comput.*, vol. 9, no. 1, pp. 302–317, Jan. 2021.

[39] X. Liu, J. Yu, Z. Feng, and Y. Gao, "Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing," *China Commun.*, vol. 17, no. 9, pp. 220–236, Sep. 2020.

[40] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in IoT edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, Jun. 2020.

[41] J. Chen, Y. He, Y. Zhang, P. Han, and C. Du, "Energy-aware scheduling for dependent tasks in heterogeneous multiprocessor systems," *J. Syst. Archit.*, vol. 129, Aug. 2022, Art. no. 102598.

[42] J. Chen, P. Han, Y. Zhang, T. You, and P. Zheng, "Scheduling energy consumption-constrained workflows in heterogeneous multi-processor embedded systems," *J. Syst. Archit.*, vol. 142, Sep. 2023, Art. no. 102938.

[43] J. Chen, T. Li, Y. Zhang, T. You, Y. Lu, P. Tiwari, and N. Kumar, "Global-and-local attention-based reinforcement learning for cooperative behaviour control of multiple UAVs," *IEEE Trans. Veh. Technol.*, 2024.

[44] G. O. Young, "Synthetic structure of industrial plastics," in *Plastics*, vol. 3, no. 2, J. Peters, Ed., 2nd ed. New York, NY, USA: McGraw-Hill, Nov. 1964, pp. 15–64. [Online]. Available: http://www.bookref.com
[45] W.-K. Chen, *Linear Networks and Systems*. Belmont, CA, USA: Wadsworth, 1993, pp. 123–135.
[46] J. U. Duncombe, "Infrared navigation—Part I: An assessment of feasibility," *IEEE Trans. Electron Devices*, vol. ED-11, no. 1, pp. 34–39, Jan. 1959.
[47] E. P. Wigner, "Theory of traveling-wave optical laser," *Phys. Rev.*, vol. 134, pp. A635–A646, Dec. 1965.
[48] H. Kurss and W. Kahn, "A note on reflector arrays," *IEEE Trans. Antennas Propag.*, vol. AP-15, no. 5, pp. 692–693, Sep. 1967.
[49] E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the earth's atmosphere," Aerospace Corp., Los Angeles, CA, USA, Tech. Rep., TR-0200 (4230-46)-3, Nov. 1988.
[50] J. H. Davis and J. R. Cogdell, "Calibration program for the 16-foot antenna," Elect. Eng. Res. Lab., Univ. Texas, Austin, TX, USA, Tech. Memo. NGL-006-69-3, Nov. 1987.
[51] *Transmission Systems for Communications*, Western Electric Co., Winston-Salem, NC, USA, 1985, pp. 44–60.
[52] *Motorola Semiconductor Data Manual*, Motorola Semiconductor Products Inc., Phoenix, AZ, USA, 1989.
[53] P. B. Kurland and R. Lerner, *The Founders' Constitution*. Chicago, IL, USA: Univ. Chicago Press, 1987. [Online]. Available: http://press-pubs.uchicago.edu/founders/
[54] ZOmega Terahertz Corp. (2014). *The Terahertz Wave eBook*. Accessed: May 19, 2014. [Online]. Available: http://dl.z-thz.com/eBook/zomega_ebook_pdf_1206_sr.pdf
[55] P. B. Kurland and R. Lerner, *The Founders' Constitution*. Chicago, IL, USA: Univ. Chicago Press, 1987, Accessed: Feb. 28, 2010. [Online]. Available: http://press-pubs.uchicago.edu/founders/
[56] J. S. Turner, "New directions in communications," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 1, pp. 11–23, Jan. 1995.
[57] W. P. Risk, G. S. Kino, and H. J. Shaw, "Fiber-optic frequency shifter using a surface acoustic wave incident at an oblique angle," *Opt. Lett.*, vol. 11, no. 2, p. 115, Feb. 1986.
[58] P. Kopyt, B. Salski, P. Zagrajek, D. Janczak, M. Sloma, M. Jakubowska, M. Olszewska-Placha, and W. Gwarek, "Electric properties of graphene-based conductive layers from DC up to terahertz range," *IEEE Trans. THz Sci. Technol.*, vol. 6, no. 3, pp. 480–490, May 2016, doi: 10.1109/TTHZ.2016.2544142.
[59] PROCESS Corporation. *Intranets: Internet Technologies Deployed Behind the Firewall for Corporate Productivity*. Boston, MA, USA. Presented at INET96 Annual Meeting. [Online]. Available: http://home.process.com/Intranets/wp2.htp
[60] R. J. Hijmans and J. van Etten. (Jan. 12, 2012). *Raster: Geographic Analysis and Modeling With Raster Data*. [Online]. Available: http://CRAN.R-project.org/package=raster
[61] Teralyzer. *Lytera UG, Kirchhain, Germany*. Accessed: 25, 2023. [Online]. Available: http://www.lytera.de/Terahertz_THz_Spectroscopy.php?id=home
[62] U.S. House. (Jan. 11, 1991). *Sense of the Congress on Approval of Military Action*. 102nd Congress, 1st Session. [Online]. Available: http://LEXISLibrary:GENFEDFile:BILLS
[63] L. M. R. Brooks, "Musical toothbrush with mirror," Patent D 326 189, May 19, 1992.
[64] D. B. Payne and J. R. Stern, "Wavelength-switched passively coupled single-mode optical network," in *Proc. IOOC-ECOC*, Boston, MA, USA, 1985, pp. 585–590.
[65] D. Ebehard and E. Voges, "Digital single sideband detection for interferometric sensors," in *Proc. 2nd Int. Conf. Optical Fiber Sensors*, Stuttgart, Germany, Jan. 1984.
[66] G. Brandli and M. Dick, "Alternating current fed power supply," U.S. Patent 4 084 217, Nov. 4, 1978.
[67] J. O. Williams, "Narrow-band analyzer," Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, USA, 1993, pp. 75–95.
[68] N. Kawasaki, "Parametric study of thermal and chemical nonequilibrium nozzle flow," M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.
[69] A. Harrison, private communication, May 1995.
[70] B. Smith, "An approach to graphs of linear forms," to be published.
[71] W. M. McKeeman, "Representation error for real numbers in binary computer arithmetic," *IEEE Trans. Electron. Comput.*, vol. EC-16, no. 5, pp. 682–683, Oct. 1967.
[72] *IEEE Criteria for Class IE Electric Systems*, IEEE Standard 308, 1973, pp. 1–25.
[73] *Letter Symbols for Quantities*, ANSI Standard Y10.5-1968, 1968.
[74] R. Fardel, M. Nagel, F. Nüesch, T. Lippert, and A. Wokaun, "Fabrication of organic light-emitting diode pixels by laser-assisted forward transfer," *Appl. Phys. Lett.*, vol. 91, no. 6, Aug. 2007, Art. no. 061103.
[75] J. Zhang and N. Tansu, "Optical gain and laser characteristics of InGaN quantum wells on ternary InGaN substrates," *IEEE Photon. J.*, vol. 5, no. 2, Apr. 2013, Art. no. 2600111.
[76] S. Azodolmolky, "Experimental demonstration of an impairment aware network planning and operation tool for transparent/translucent optical networks," *J. Lightw. Technol.*, vol. 29, no. 4, pp. 439–448, Sep. 15, 2011.

ANJAN BANDYOPADHYAY received the M.Tech. degree in information security from the Department of Information Technology, NIT Durgapur, West Bengal, India, and the Ph.D. degree from NIT Durgapur. He is currently an Assistant Professor with the Kalinga Institute of Industrial Technology, Bhubaneswar, Odisha, India. He is broadly interested in algorithmic game theory (mechanism design). He has published many conference and journal papers in esteemed conferences and journals. His current research interests include cloud computing, fog computing, metaverse, the IoT, healthcare, crowd-sourcing, and image processing. He was a recipient of a number of Best Paper Awards in many conferences, such as 3PGCIC. He also received the Visvesvaraya Ph.D. Fellowship under MHRD for the Ph.D. degree. He is a reviewer of many journals and conferences.

VAGISHA MISHRA is currently pursuing the B.Tech. degree with the Kalinga Institute of Industrial Technology, Bhubaneswar, with a profound passion for cutting-edge technologies. Her research interests include cloud and fog computing, the IoT environments, and game theory.

SUJATA SWAIN received the B.Tech. degree in computer science and engineering from BPUT University, India, and the M.Tech. and Ph.D. degrees in computer science and engineering from the Indian Institute of Technology Roorkee, India. She is currently an Assistant Professor with the School of Computer Science and Engineering, Kalinga Institute of Industrial Technology Deemed to be University, Bhubaneswar. Her research interests include service-oriented computing, metaverse, medical imaging, and healthcare systems.

KALYAN CHATTERJEE received the B.Tech. and M.Tech. degrees from the Department of Computer Science and Engineering, State Government University, West Bengal, India. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Nalla Malla Reddy Engineering College, Hyderabad, Telangana, India. He is broadly interested in machine learning and data science. He has more than seven years of teaching experience in B.Tech. colleges in India. He has published journal articles in esteemed journals as well as conference papers. His current research interests include air pollution, climate, the IoT, big data, and healthcare. He is a professional member of ACM.

SWETA DEY received the M.Tech. degree from the Department of Computer Science and Engineering, Indian Institute of Technology, Dhanbad, India, in 2018. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Indian Institute of Technology Ropar, India. Before joining the Ph.D. degree, she has almost two years of research experience with the Indian Institute of Technology, Kharagpur, India. Her current research interests include environmental science, weather research, machine learning (ML), and deep learning (DL).

SAURAV MALLIK (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, Jadavpur University, Kolkata, India, in 2017. The Ph.D. studies were conducted with the Machine Intelligence Unit, Indian Statistical Institute, Kolkata, India. He was a Postdoctoral Fellow with the School of Biomedical Informatics, The University of Texas Health Science Center at Houston, Houston, TX, USA; and the Division of Bio-Statistics, Department of Public Health Sciences, University of Miami Miller School of Medicine, Miami, FL, USA. He is currently a Postdoctoral Fellow of environmental epigenetics with the Harvard T. H. Chan School of Public Health, Boston, MA, USA. He has coauthored more than 130 research articles with a Google H-index of 19. His research interests include computational biology, bioinformatics, data mining, bio-statistics, and pattern recognition. He was a Research Associate of the Council of Scientific and Industrial Research, MHRD, Government of India, in 2017. He was a recipient of the Emerging Researcher in Bioinformatics Award from the Bioclues BIRD Award Steering Committee, India, in 2020. He is an editor of many journals.

AMAL AL-RASHEED received the Ph.D. degree in information systems from King Saud University, in 2017. She is currently an Associate Professor with the Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University (PNU), Riyadh, Saudi Arabia. She has been involved in many projects related to learning technologies, cyber security, and virtual reality. Her contributions to research projects in academia led to the publication of papers in many journals and conferences. Her research interests include education, knowledge management, data mining, data analytics, cyber security, and natural language processing. In 2017, she was awarded the Research Excellence Award, by PNU, for her publications during performing the Ph.D. degree.

MOHAMED ABBAS received the B.Sc. degree in electronics engineering from Mansoura University, College of Engineering, Egypt in 1998, and the M.Sc. and Ph.D. degrees in computer engineering from Mansoura University in 2002 and 2008, respectively. He worked as an Assistant Professor at the Department of Communications and Computer Engineering, College of Engineering, Delta University. He is currently working as a Full Professor with the Department of Electrical Engineering, King Khaled University, Abha, KSA. His areas of interest are intelligent systems, medical informatics, nanotechnology, and bioinformatics.

BEN OTHMAN SOUFIENE received the M.S. degree from the University of Monastir, in 2012, and the Ph.D. degree in computer science from Manouba University, in 2016, with a focus on secure data aggregation in wireless sensor networks. From 2016 to 2023, he was an Assistant Professor of computer science with the University of Gabes, Tunisia. His research interests include the Internet of Medical Things, wireless body sensor networks, wireless networks, artificial intelligence, machine learning, and big data.