[1]Huda Qasim ALGawwam

[2]Mohamad Mahdi Kassir

[3]Amir Lakizadeh

# Power-Efficient Dispatching of Time-Sensitive Requests in the Fog-Enabled Industrial Internet of Things

Journal of Electrical Systems

***Abstract:*** - Recent advancements in service offerings and the emergence of new application areas have reveal the limitations of cloud computing as a solely solution. One of these new areas is the industrial internet of things, where industrial processes are controlled in a network-based and automated way which brings many new challenges. To solve the challenges, fog computing has been introduced as a complementary paradigm to the cloud. Resource management is one of those challenges need further investigation. The industrial internet of things by its own characteristics has led traditional methods to be inefficient within this evolving ecosystem. Therefore, fully utilizing potential of the new system necessitates effective resource management. This manuscript aims to investigate the problem of resource management in the emerging cloud-fog ecosystem of the industrial internet of things. Considering the dynamic nature of users, this study proposes the algorithms for dynamic resource allocation and provisioning based on deadlines. We formulate the problem as a multiple criteria stochastic problem and introduced an online solution. Our proposed solution takes into account the unique characteristics of Fog computing and the dynamic behavior of users. To effectively handle the scheduling of requests, the research suggests utilizing the Lyapunov optimization method and Lyapunov drift theory. The Lyapunov method helps to merge and measure all the related objectives in terms drift. The objective function is optimized, in each time slot, using a drift-plus-penalty weight parameter. By adopting this approach, the system can operate efficiently while meeting the constraints and performance requirements of IoT applications.

***Keywords:*** Cloud, Fog/Edge Computing, Internet of Things (IoT), Industrial Internet of hings (IIoT), and Resource management.

## 1. INTRODUCTION

Cloud computing has transformed the way people access internet-based resources like processing power, storage, and communication capabilities. Cloud computing make such resources available on-demand to users who are located in different places [1]. Cloud computing is now a mature technology and used in many applications. However, the growing tendency to cloud computing and emerging new services and applications arise a need for a novel computing paradigm that can effectively satisfy the new Quality of Service (QoS) requirements.

Furthermore, the rise of smart systems including smart homes and smart cities to name some, heavily based on the Internet of Things (IoT). The IoT is characterized by its diverse nature and limited resources and is widely recognized as the next phase of the internet. The IoT by connecting various devices and facilitating collaboration between humans and machines, gives rise to a wide range of applications, encompassing networks involving machine-to-machine, human-to-machine, and human-to-human interactions. Cloud computing, by processing data and delivering services, is a key player in the IoT architecture [2]. Even so, IoT devices generate a massive amount of data that can make the network core busy and lead to congestion and delays. Therefore, time-sensitive applications and services such as industrial processes, face even greater challenges when applied to.

Sensors, messaging systems, mobile devices, and social networks have all been used to continuously generate a vast volume of data from a novel network architecture introduced by the IoT. Due to the distributed, complex, and dynamic nature of the IoT environment, there are several technical challenges that need to be addressed. These challenges include aspects such as latency, connectivity, capacity, cost, performance, scalability, and reliability [3]. Although the cloud is often relied upon to process data generated by IoT, it is not always possible to transfer all data to the cloud for storage and analysis [4]. The data transfer process consumes a large amount of network bandwidth and also due to double the time for some applications, the cloud cannot meet the access time requirements [5]. Cloud computing has been presented as an alternative solution to traditional cloud computing. The cloud gives the cloud geographically distributed resources, and its goal is to bring computing resources closer to the edge of the network, which strongly contradicts the functioning of the centralized nature

[1] [1,2,3] Department of Computer Engineering, University of Qom, Qom, Iran.

E-mail: Huda.qassim88@gmail.com, m.kaseer@alumni.iut.ac.ir, lakizadeh@qom.ac.ir

of the cloud [6]. Resource rounding has many benefits, including lower latency, faster response time, and increased security and privacy. There are indeed great benefits to cloud computing, but it is still new and has not been widely studied. By researching, studying, and understanding the complexities associated with monetary easing, it becomes possible to fully realize the potential and expected benefits of financial easing. This leads to a very urgent need for more in-depth research and studies into various aspects of cystic fibrosis.[7] One of the main tasks of cloud computing is to ensure easy integration and interaction between the cloud and IoT end devices by creating a continuous chain of interconnected resources. This connectivity gives efficient and robust processing and control of information at the edge and reduces the need to transfer information to a remote cloud server. On the other hand, FC can make real-time decisions, support low-latency applications, and save network bandwidth. Scientists and professionals are actively researching various aspects of CF to realize its true potential. This includes developing a robust FC architecture, efficient and differentiated resource management and implementation techniques, reliable communication protocols, and secure and reliable data processing mechanisms. In fact, identifying resources in the context of financial cooperation represents a fundamental challenge that must be considered with great care. Due to the large number of available fog nodes, it is necessary to select appropriate resources very efficiently when users submit tasks to the fog layer. Key considerations include allocating the number of resources required and selecting the ideal fog node to successfully complete the task. Addressing problems is very important for the effective and successful implementation of FC. Planning aims to optimize the utilization of resources and improve the overall performance of the system by strategically assigning tasks to various subordinates. Efficient scheduling plays an important and key role in achieving low latency operations. It avoids overloading a single device because it balances the load between fog nodes. When power management becomes necessary to extend the life of fog devices, efficient planning becomes very important [8]. In cloud computing, in order to solve the task problem, an innovative approach has been developed that requires assigning a set of tasks to distributed nodes with high limited computing performance and takes into account the unique characteristics of cloud computing and the dynamic behavior of users when receiving a request. The main goal of cloud computing is to develop programming and optimization techniques that improve system response time, reduce power consumption, and improve user experience in various wireless applications. To effectively address the challenges, this research proposes to apply Lyapunov optimization theory and Lyapunov drift theory. This proposal includes defining system constraints and objectives and creating queues. The objective function was optimized in each period using a drift and penalty term. This approach ensures that a system operates efficiently while meeting the constraints and performance requirements of IoT applications and services. A wide range of metrics are used to evaluate techniques used in resource allocation. These metrics have traditionally focused primarily on topics such as resource utilization and machine/task downtime, as they provide valuable insights into the performance and effectiveness of resource management strategies. However, one area that has received relatively less attention is allocating resources to efficiently meet user request deadlines. The goal of this research is to design algorithms for efficient resource management in FC environments, focusing on preserving task execution time while minimizing energy consumption. In real-world scenarios, it is important to ensure that resources are allocated to ensure tasks are completed in a timely manner. To address these challenges, we first formulate the problem as a stochastic problem. Then translate the restrictions and goals into Lyapunov's concepts of drift and penalty kicks. The finally obtained simplified optimization expression is solved using Lyapunov's method. To evaluate the effectiveness of the proposed algorithm, we designed and implemented a simulation environment in MATLAB. These simulations are specifically designed to reproduce the complex dynamics of the small nebula environment and enable comprehensive algorithm testing and analysis. By simulating different scenarios and workloads, the performance and effectiveness of resource management strategies are evaluated under different conditions. The goal of this research is to contribute, through careful experiments and analyses, to the development of resource allocation techniques that not only optimize resource utilization and reduce downtime, but also meet project implementation deadlines. Ultimately, the goal is to improve the overall efficiency, reliability and user experience of FC systems in a time-sensitive and energy-efficient manner.

Our contribution in this research can be summarized as follows:

1. Effectively address the dynamic nature of user needs within the FC environment, while considering potential delays.

2. Efficiently handle application execution within the fog environment, considering the dynamic nature of resources and the limited resources available on devices.

3. Promote energy-efficient processing of applications to maintain the sustainability of the application execution environment.

The remaining sections of the paper are organized as follows: Section 2 provides a concise overview of the researchers' findings, including their limitations. It also explores open issues and potential research opportunities. In Section 3, we delve into the architecture, system model, and its key components. Section 4 focuses on the problem formulation and describes the objective functions employed. Section 5 introduces our proposed solution to address the research problem. Section 6 elaborates on the simulation setup and presents various simulation scenarios. Additionally, the results of the experiments are discussed within this section. Finally, Section 7 concludes the research, summarizing the key findings, and highlights future research directions for further exploration.

## 2. RELATED WORKS

In this section we provide a review of the most relevant works in the context of FC. We begin by investigating basic concepts such as the IoT, cloud computing, and FC, along with their respective characteristics. We then move on to explore resource allocation strategies and methods in both cloud and FC. Finally, we shift our focus to the Industrial Internet of Things (IIoT) and investigate resource allocation methods specifically tailored to fog-enabled IIoT environments.

### 2.1. INTERNET OF THINGS

A combination of the words 'internet' and 'thing', IoT refers to a global network of interconnected objects using standard communication protocols. The future entails collaborative and interactive networks of objects. Things are essential components with unique identifiers, embedded logic, and data transmission abilities. They include RFID tags, sensors, and actuators, enhancing physical objects into smart objects with sensing, processing, and networking capabilities. Smart objects serve as digital proxies within the IoT framework, while digital entities can also exist independently as autonomous agents providing network services.

Businesses will gain significant economic growth potential through the use of IoT-based services. Among the various sectors, healthcare and manufacturing are expected to have the most significant economic impact. In particular, healthcare applications and related IoT-based services, such as mobile health (mHealth) and telemedicine, which facilitate the efficient delivery of medical wellness, prevention, diagnostic, treatment and monitoring services through electronic media, will add between \$1.1 trillion and \$2.5 trillion annually to the global economy by 2025.

In general, it is expected that the IoT will contribute to an annual economic impact of between USD 2.7 trillion and USD 6.2 trillion by 2025. A wide range of IoT applications is included in this estimate. Figure 1 is an illustration of the forecast market share of the dominant IoT applications [9].

The substantial economic impact attributed to the IoT stems from its potential to enhance efficiency, reduce costs, and boost productivity across various industries. IoT technologies facilitate real-time data analysis, automation, and remote monitoring, enabling businesses to optimize their operations and unlock new revenue streams. Consequently, the IoT is positioned to emerge as a significant catalyst for economic growth in the forthcoming years.

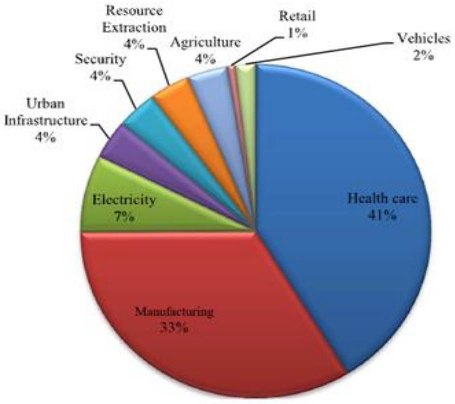

Figure 1. Expected market share of primary IoT applications by 2025

## 2.2. CLOUD COMPUTING

Cloud computing refers to the concept of accessing computing resources over the Internet, rather than relying solely on local or on-premises computing infrastructure. In a similar fashion to meteorological clouds, computing clouds possess characteristics such as being vague, open to interpretation, distributed, and unpredictable, much like the weather itself.

In recent times, the term "cloud computing" has become colloquially used to describe any software service accessed over the internet at a higher level, with customers not needing to be concerned about the specific location or hosting method of the service. Unlike traditional desktop personal computers or localized campus clusters, cloud computing organizes computing resources in a manner that allows for near-instantaneous access to seemingly abundant amounts of resources.

With its unique properties, cloud computing holds the promise of advancing us towards a state of ubiquitous computing, where computing resources are readily available, acquired, and consumed as a utility. Large-scale Data Centers (DCs) are essential for cloud computing. They house and manage compute, storage, and networking resources, serving as the backbone of cloud infrastructure. Cloud providers excel in operating and maintaining these facilities, enabling computing at a grand scale and leveraging economies of scale. This competitiveness positions cloud computing favorably compared to offline computing in multiple aspects.

Cloud computing services are typically categorized as either public or private. Public clouds, exemplified by major providers like AWS, Microsoft, Google, and IBM, offer cloud resources and services to anyone who wishes to procure them at market prices. Public clouds are available to the general public and are designed to serve a wide range of customers. Private clouds, on the other hand, are domain controllers that are managed similarly to the cloud, but only serve a specific group of customers, such as a specific company or organization. Thus, cloud computing includes both public clouds, which are accessible to the general public, and private clouds, reserved for specific groups of private customers. Traditional cloud computing uses remote domain controllers that are shared between multiple applications. However, the multi-tenant nature of these data centers can have a significant impact on the performance of individual applications. Despite the potential lack of usage, cloud operators are trying to balance operating costs and perceived performance by implementing resource management mechanisms in their data centers. Several factors affect the performance of cloud applications, including the round-trip time (RTT) between the remote user and services hosted in the cloud provider's data center. RTT includes both network latency and system latency generated by the cloud platform itself. These delays can impact response times and overall performance a customer experiences when interacting with cloud services. Cloud operators recognize the importance of managing these factors and are committed to optimizing resource allocation and utilization to reduce latency and provide satisfactory performance to their customers. Cost efficiency and performance are central issues in the design and operation of cloud data centers.

## 2.3. FOG COMPUTING

FC is a distributed computing infrastructure designed to handle the massive influx of devices connected to the Internet. It operates on a model in which data, computing power, and applications are concentrated on the edge rather than largely in the cloud. The main goal of FC is to bring these devices closer to the end user and effectively isolate them from cloud systems. FC is a sub-cloud layer that serves as an ideal transport medium for services and data in cloud infrastructure. It operates outside the traditional cloud environment and enables distributed delivery of cloud services, including computing, storage, workloads, applications, and big data, at any edge or anywhere on the Internet. FC integrates with core cloud services, transforming DCs into distributed, user-accessible cloud platforms by managing data at various edge points. In essence, FC shifts the focus of computing from the central core of the network to the edge, bringing computing power closer to where it is needed.

## 2.4. RESOURCE ALLOCATION IN CLOUD/FOG COMPUTING

Cloud computing is indeed a promising paradigm for IoT computing and commercial applications [10]. However, the cloud computing model is not well suited for time-critical applications due to its high latency and real-time interaction challenges. To overcome this limitation, FC has emerged as a solution. It enables application processing closer to the user by exploiting the available processing power of heterogeneous end and edge devices [11].

FC has gained importance in supporting time-sensitive applications associated with Intelligent IoT services, including smart cities and smart healthcare. For instance, in the domain of smart transportation, FC can be utilized to process real-time data and enable applications such as collision warning systems [12]. In this scenario, a vehicular FC service architecture is established using dedicated short-range communication, where roadside units serve as fog nodes along the road.

By leveraging the FC paradigm, time-sensitive applications can benefit from reduced latency and improved real-time interaction. FC complements cloud computing by extending the computing infrastructure to the network edge, enabling efficient and timely processing for applications that require immediate response and low latency.

Despite the increasing attention given to FC resources and task scheduling in recent times, the administration of FC resources and task scheduling are currently at the early stages of this paradigm [13]. For example, a placement strategy was introduced by Arshed et al. [14] to perform resource optimization in a FC environment. Instead of sending application modules directly to the cloud, its resource-aware scheduler assigns incoming application modules to fog devices, which then selectively route them to the cloud. Moreover, Liao et al. [15] presented a scheme for prioritizing application-dependent tasks. To solve the application-based mobile task scheduling problem, the dependencies are represented in a directed acyclic graphical model. This allowed them to model complex customization and planning for application-based applications. Using heuristics based on estimated priorities, the authors solve the NP-hard joint optimization problem of task allocation and scheduling. Moreover, Wang et al. [16] presented a new approach to calculate surface runoff. To effectively manage resource availability and optimize resource utilization, the authors present a model that represents dynamic computing and communication resources in the form of two different dynamic queues. Moreover, Sheikh Sofla et al. In [17] and Feng et al. In [18] extensive research on state-of-the-art relief mechanisms in FC environment. Her research provides a detailed overview of current work in this area and suggests potential directions for future research in this area. To capture the user's perceived service quality, some studies have presented mathematical models such as optimization theory [19], game theory [20], machine learning [21], and analytical hierarchy processes [22]. These models allow accurate evaluation and estimation of user service quality, which facilitates Network performance analysis and resource allocation. By incorporating these models into network performance analysis, valuable insights can be obtained for tuning and improving network performance. [23, 24].

## 2.5. INDUSTRY 4.0 AND IIOT

Industry 4.0 was introduced in 2011 as part of the High-Tech Strategy 2020 action plan. It is a strategic initiative of the German government aimed at revolutionizing the manufacturing process [28-30]. It promotes new technologies in the industrial environment and involves the convergence of the physical, digital, human and biological worlds. One key pillar of Industry 4.0 is implementing the IoT and Services (IIoT) in factory environments [31]. A key enabler of the Fourth Industrial Revolution is the IIoT, also known as the Industrial

Internet, the Internet of Everything and Internet 4.0 [32, 33]. It involves the development of a network of industrial devices [34, 35], including sensors, complex industrial robots and actuators. These devices are connected through communication technologies, allowing for the monitoring, analysis, delivery, collection, and exchange of data in a fast and efficient manner [36]. The Industrial Internet enables the realization of key features of Industry 4.0, such as horizontal integration through value networks, end-to-end engineering by integrating the digital and real worlds, and vertical integration by networking manufacturing systems.

The combination of IIoT and Industry 4.0 brings benefits like IT-OT convergence, improved asset performance, faster decision-making, and new business models [37]. It enables direct communication between OT components and centralized servers via an IT network, reducing operations and expanding business opportunities. This leads to the development of factories incorporating Cyber-Physical Systems [38, 39]. Despite the huge advantages that IIoT and Industry 4.0 have to offer, they are complicated to implement. The widespread digitization and networking of companies involved in this transformation can lead to the creation of different architectures by different authors, resulting in communication, networking and system interoperability challenges. [40, 41].

### 2.6. FOG-ENABLED IIOT

Numerous studies in the academic literature have showcased the practicality of FC in the context of Industry 4.0 [42, 43]. Furthermore, several placement strategies have been proposed to cater specifically to Industry 4.0-oriented applications (I4OAs). For instance, Verba et al. [44] have devised a profiling approach for I4OAs, enabling the placement of applications with improved service times while mitigating the impact of contextual variations. In a similar vein, Lin et al. [45] have presented a hierarchical platform that deploys Fog nodes, effectively addressing concerns related to application latency and capability constraints. Similarly, Chekired et al. [46] prioritize the placement of I4OAs based on their susceptibility to latency. Additionally, Wan et al. [47] have developed a policy that optimally distributes application workloads across manufacturing components while considering the energy consumption associated with data size. According to Xu et al. [48], the integration of IoT in industrial automation is currently limited in scope. While industrial automation IoT solutions are still evolving, fog-based methods have demonstrated the ability to meet the requirements of modern industrial systems. However, existing research [49, 50] primarily focuses on centralized computation architectures that rely on cloud computing for control processes and data monitoring in industrial automation. Many current approaches and solutions in industrial automation utilizing cloud computing tend to concentrate on higher-level aspects rather than the field level. In a study by the authors [49], a prototype was examined to explore the communication between IoT devices and a cloud-based controller dedicated to automation. However, this work only addresses network-induced delays in communication and overlooks delays and computational capacities imposed by the cloud server. Furthermore, the mitigation model lacks a concrete mathematical validation.

## 3. SYSTEM MODEL

In this section, we will delve into the proposed high-level architecture and system model presented in the study. Firstly, we will examine other related concepts that share similarities. Subsequently, a comprehensive overview of the architecture will be presented. Additionally, we will elucidate the notations employed throughout the remainder of the paper. Lastly, we will provide a mathematical description of the FC system and its associated constraints.

### 3.1 SIMILAR CONCEPTS

FC is similar to geo-distributed clouds, edge computing, mobile cloud computing, and cloudlets. These paradigms place computational resources close to users and share resource allocation models. Geo-distributed clouds have multiple DCs in different regions, offering low latency and fault tolerance. Edge computing pushes computation to the network edge for reduced latency. Mobile cloud computing offloads tasks from mobile devices to remote resources. Cloudlets are small-scale cloud-like resources at the network edge.

### 3.2 SYMBOLS AND SIGNS

To enhance text comprehension and facilitate tracking, this section aims to clarify the symbols and signs utilized throughout the remainder of the report, Table 1. This explanation will contribute to improved readability and comprehension of the subsequent content.

**Table 1: SYMBOLS AND SIGNS**

| Symbols | Definition |
|---|---|
| $t$ | Index of time slot |
| $P$ | collection of computing nodes |
| $K^t$ | Current request to be dispatched |
| $A(t)$ | Arrival rate |
| $i$ | Computing node index |
| $R_i(t)$ | Rate of incoming requests to computing node $i$ |
| $B_i(t)$ | Rate of serving requests at the computing node $i$ |
| $c(t)$ | Control action |
| $\Xi$ | Set of possible control decisions |
| $e_i(t)$ | Total energy consumption for processing node $i$ |
| $e_i^P(t)$ | Processing related energy consumption for node $i$ |
| $e_i^C(t)$ | Communication related energy consumption for node $i$ |
| $k$ | Index of the arrived request |
| $S_k$ | Processing demand of request $k$ |
| $D_k$ | Communication demand of request $k$ |
| $u_{k,i}$ | Time to Upload for request $k$ to node $i$ |
| $w_{t,i}$ | Amount of time that request $k$ wait to acquire the resources on node $i$ |
| $p_{k,i}$ | Amount of time that request $k$ needs to be processed on node $i$ |
| $d_{k,i}$ | Downloading time for request $k$ on the processing node $i$ |
| $L(t)$ | Lyapunov function |
| $Q$ | Vector of all real queues |
| $\theta$ | collection of all the queues |
| $\Delta(L_Q)$ | Drift in Lyapunov function with related to $Q$ |
| $\Delta(L_\theta)$ | Drift in Lyapunov function with related to $\theta$ |

### 3.3. QUEUE MODEL OF FOG

The processing nodes (both cloud and fog) act as service providers in the queuing model and are characterized by the Poisson process pattern. Each node is assigned a dedicated queue where incoming requests are directed and serviced at a predetermined rate. It is assumed that the requests are independent of one another and follow a uniform distribution. The system operates under the assumption that requests enter at a rate denoted as A(t) (for instance, a Poisson process with a mean arrival rate of "λ"). As such, the Poisson queuing model can be represented in the format depicted in Figure 2.

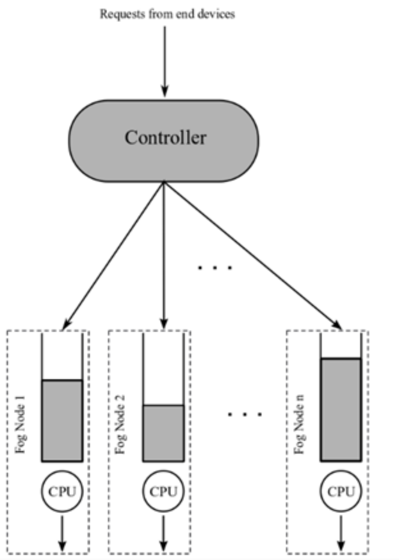

Figure 2: Queue model of system

Here, we focus on a cloud/fog environment that encompasses N processing nodes, a remote DC, and a substantial number of end devices. Consequently, there will be a total of N+1 processing nodes available to handle the requests. The collection of nodes in cloud-fog ecosystem is denoted by N={1,2,…,N}. These processing nodes communicate with objects via radio-based communication channels and are located in geographically wide spread locations.

Let us assume that the distribution of requests within the system and their scheduling at discrete time intervals, referred to as time slots denoted by t, will take place. Thus, we will consider the system to operate in discrete time, where each time slot corresponds to an integer value of t ∈ {0,1,2, ...}. It is also assumed that the system conditions, such as the status of processing nodes, the transmission channel, and link quality, remain constant throughout a fixed time slot but may vary between different time slots.

The controller performs its algorithm, at each time slot, to obtain the control action for assigning requests to the suitable processing nodes. These control decisions are defined as a set of ordered pairs represented by $c(t) = \{(k, i)|k \in K^t, i \in P\}$, where $K^t$ specifies the set of requests ready for distribution in time slot t, and P denotes the pool of computing nodes both in cloud and fog. Specifically, P = N ⋃ C, where C represents the cloud node, resulting in a total of |P| = N+1 nodes.

The requests in the system may have varying processing capacity requirements. Once a request enters the queue of a processing node, it will be processed by that node after a possible waiting period. The processing of requests at each node is considered a random process with an average processing time of μ.

Based on the control algorithm implemented on the management platform, which is distributed across software components deployed on gateways, requests will be allocated to either cloud or fog nodes. It is assumed that the allocation satisfies the condition $0 \leq R_i(t) \leq R_i^{max}$, where $R_i(t)$ represents the allocation of requests at time t to node i, and $R_i^{max}$ represents the maximum processing capacity of node i. The overall arrival rate of requests, denoted as A(t), is defined as the summation of all request allocations over N+1 nodes, such that $A(t) = \sum_{i=1}^{N+1} \quad R_i(t)$.

The management platform is responsible for maintaining and updating essential information about the fog nodes, including their available processing power, access delay, and other relevant parameters. It is assumed that the delays caused by exchanging control information, updating the controller's information, and introducing requests to the controller are negligible. In other words, the time interval Δt between two time slots is greater than the overhead time $t^{oh}$. This assumption allows us to address the problem in dynamic and evolving conditions, where we seek a solution under such time constraints.

Processing nodes can be modeled as queues. The fog pattern is represented by a set of queues and their relationships. The length of each queue at a given time determines the next state of the system, based on received requests and processing capability, using a dynamic equation **(**1**)**.

$$Q_i(t+1) = [Q_i(t) - B_i(t), 0] + R_i(t) \;\; for\ t \in \{0, 1, 2, \ldots\}. \tag{1}$$

The initial queue state, $Q_i(0)$, is a random variable with non-negative real values. $Q_i(t)$ represents the accumulated queue length at time t, indicating the pending workload. The stochastic processes $R_i(t)$ and $B_i(t)$ are sequences of random variables with real values over time slots t ∈ {0,1,2, ...}. $R_i(t)$ and $B_i(t)$ represent the incoming workload and the service rate of the node i during t-th time slot. Both $R_i(t)$ and $B_i(t)$ are non-negative real values. The service rate $B_i(t)$ can generally be determined using equation **(**2**)**.

$$f_m/\vartheta_i\,, \tag{2}$$

ϑ_i represents the average instruction execution time per node.

### 3.4. ENERGY CONSUMPTION MODEL

Each node's energy consumption ($e_i(t)$) includes both computation ($e_i^P(t)$) and communication ($e_i^C(t)$).

$$e_i^P(t) + e_i^C(t)\,, \tag{3}$$

The energy consumption of each processing node, dependent on the processor frequency ($f_m$), can be expressed as equation **(**4**)**.

$$e_i^P(t) = \sum_{k=1}^{|K_i^t|} \;(\alpha_i . f_m{}^3 + e_i^{P,ind}).(\frac{S_k}{B_i(t)}) \tag{4}$$

$e_i^{P,ind}$ is a non-speed-dependent component of processor power consumption. $|K_i^t|$ represents the count of requests allocated to i-th processing node during time slot t. $S_k$ indicates the processing requirement for request k.

The communication energy consumption in each processing node, dependent on the encoding level, is determined by equation **(**5**)**.

$$e_i^C(t) = \sum_{k=1}^{|K_i^t|} \left(\beta_i . r_i(t)\left(2^{\frac{x_i(t)}{r_i(t)}} - 1\right) + e_i^{C,ind}\right).\left(\frac{D_k}{x_i(t)}\right), \tag{5}$$

$e_i^{C,ind}$ represents the power consumption component independent of the signal encoding level. $D_k$ indicates the amount of data required for transmitting request k.

## 4. Problem Statement

In this section, we will present the mathematical formulation of the request dispatching problem. Initially, we will define the objective function, followed by formulating the main problem as a stochastic optimization problem. Lastly, we will rephrase the problem formulation using the principles of Lyapunov optimization.

### 4.1. Objectives

Numerous IoT services require efficient distribution for timely processing, as sending them to the cloud may result in delays or reduced quality. Minimizing service delivery time is crucial, achieved by distributing requests across cloud or edge computing nodes. Service delivery time (($\psi_k(t)$) encompasses various factors such as request transmission, queue waiting, processing, and response transmission, as shown in equation **(**6**)**.

$$\psi_k(t) = u_{k,i} + w_{t,i} + p_{k,i} + d_{k,i} \, , \qquad \textbf{(6)}$$

where, $u_{k,i}$ represent the upload time for request $k$ to the processing node $i$. The waiting time in the queue is denoted by $w_{t,i}$, while $p_{k,i}$ denotes the processing time. Lastly, $d_{k,i}$ signifies the time taken to send the processed request back to the requester.

Moreover, by utilizing equations **(**3**)** and **(**4**)** as a foundation, we can represent the total energy usage of the incoming requests during the current time slot in the form of equation **(**7**)**.

$$\sum_{k=1}^{|K^t|} \hat{e}(c_k(t)) \qquad \textbf{(7)}$$

Given the significance of time and energy in efficiently handling incoming requests within our proposed system, we establish the objective function. To this end, we denote the objective function as Φ. Φ is a function of two variables, $\psi(t)$ and $e(t)$. Thus, we can express Φ as $\Phi(t) = \Phi(\psi(t), e(t))$, or simply $\Phi(\psi, e)$. The equation **(**8**)** represents the formulation of the objective function Φ.

$$\Phi(t) = (\omega_1 . \psi(t)) + (\omega_2 \, . e(t)) \qquad \textbf{(8)}$$

### 4.2. Problem Definition

The challenge of allocating requests in a way that minimizes the average service time over the long term while also reducing resource energy consumption can be framed as a stochastic optimization problem, as represented by equation (9).

$$\begin{gathered} P1: \; \left( lim\, sup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \quad E\{\Phi(\psi(t), e(t))\} \right) \\ S.T \quad c(t) \in \Xi \\ \underline{Q}_i(t) < \infty, where\ i \in N \end{gathered} \qquad \textbf{(9)}$$

We can rewrite equation **(**9**)** to explicitly indicate its dependence on the control decision variable c(t) as shown in equation **(**10**)**.

$$\begin{gathered} P2: \; \left( lim\, sup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \quad E\{\Phi(\hat{\psi}(c(t)), \hat{e}(c(t)))\} \right) \\ S.T \quad c(t) \in \Xi \\ \underline{Q}_i(t) < \infty, where\ i \in N, \end{gathered} \qquad \textbf{(10)}$$

To tackle this multi-criteria optimization problem mentioned above, obtaining real-time information about the system is crucial. This includes parameters such as arrival rate, processing volume distribution, and request correlation. To overcome this hurdle, we adopt the Lyapunov function method, which guides decision-making based on the present state of the node and queue length. In the subsequent discussion, we will begin by examining the Lyapunov method and the notion of virtual queues. Subsequently, we will delve back into the primary problem and propose a solution employing the Lyapunov optimization method for problem P1 (or its equivalent, P2).

## 5. Lyapunov Optimization

In this section, we will present the optimal Lyapunov optimization approach. This method incorporates both the evaluation of queue length through the creation of the Lyapunov function variation and the integration of decision-making by introducing a weighted term that accounts for the cost function.

### 5.1. Lyapunov framework

Optimal decisions balance stability, cost, and performance to prevent suboptimal choices that increase costs or decrease performance unnecessarily. Lyapunov optimization minimizes the upper limit of the combined expression, incorporating function change and weighted cost, instead of solely minimizing function change $\Delta(L_Q)$.

$$\Delta(L_Q) + VE\{\pi(t)|Q(t)\}, \tag{11}$$

This approach involves a trade-off between reducing queue length and minimizing costs. The second part of the expression can encompass various cost reduction objectives such as power consumption and delay, as well as improving performance parameters like request execution speed and efficiency. We will refer to this combined concept as "cost" in the following.

### 5.2. Lyapunov Function

In computer science, the Lyapunov function method assesses stability and its related conditions. It models the system as queues using queuing theory. Stability is achieved when queue length is limited. The Lyapunov function represents congestion in queues. It is a positive, increasing, and continuous function. The Lyapunov function can be defined as equation (12).

$$\sum_{i=1}^{N+1} Q_i(t)^2 \tag{12}$$

### 5.3. Dispatching the Requests

Considering the Lyapunov function definition, the penalty function described in equation (6), (7) and (8), and the Lyapunov optimization equation (10) derived in the previous section, they can be rewritten as equation (13).

$$\begin{aligned}\Delta(L_\theta) + VE\{\Phi(t)|\theta(t)\} \\ &\leq B + VE\{\widehat{\Phi}(c(t))|\theta(t)\} \\ &-\sum_{i=1}^{N+1} Q_i(t)\mu i \\ &+\sum_{i=1}^{N+1} Q_i(t)E\{\hat{R}_i(c(t))|\theta(t)\}\end{aligned} \tag{13}$$

Hence, we are faced with the challenge of problem **(**13**)**, which involves distributing requests among various nodes to achieve stability, maintain efficiency, and adhere to the constraint of effectively utilizing clean energy resources.

$$minimize_{c(t)}: VE\left\{\widehat{\Phi}(c(t)\ )|\theta(t)\right\} + \sum_{i=1}^{N+1} \quad Q_i(t)E\{\widehat{R}_i(c(t))|\theta(t)\} \tag{14}$$

By employing the concept of minimizing the optimistic expected value, my objective is to reduce the right-hand side of the aforementioned inequality. The optimization algorithm, known as the "change-of-measure + cost" algorithm, monitors queue accumulation and node status at time t. Based on this information, it selects the optimal control decision, denoted as c(t), to achieve the most favorable placement. To accomplish this goal, I adopt a greedy approach that involves minimizing expression **(**14**)** across all queues and the current request.

$$minimize_{c(t)}: V\widehat{\Phi}(c(t)) + \sum_{i=1}^{N+1} \quad Q_i(t)\widehat{R}_i(c(t)), \tag{15}$$

The Lyapunov optimization approach operates under the belief that the queue accumulations within the system offer adequate information regarding its current state. Consequently, decision-making, as depicted in equation **(**15**)**, relies on the accumulation of system queues and the cost function. It remains independent of dynamic system information, such as request arrival rates, processing volumes, and their communication.

### 5.4. Proposed Algorithm

To address the problem outlined in equation **(**15**)**, we employ an online algorithm that operates in real-time, receiving continuous updates on requests and system status, and subsequently makes decisions. This algorithm aims to achieve the minimum achievable value by considering all potential random strategies. We summarize this algorithm as Algorithm 1. Importantly, the algorithm is designed to be executed on the controller, and it does not rely on knowledge of future requests or the subsequent state of the system.

**Algorithm1:** *Proposed Algorithm for Request Dispatching (PARD)*

```
Input: the set of computing nodes and incoming requests
Output: controller action $c^*(t)$
Preparation
    Preparation of the system parameters
While $t < t_{end}$ , do
  For $k = 1$ to $K^t$
    For $i = 1$ to $N + 1$
        $LDpP(j) = V\widehat{\psi}(c(t)) + \sum_{i=1}^{N+1} Q_i(t)\widehat{R}_i(c(t))$
        $j$++
    end for
  end for
  $c_k^*(t)$ =$arg\ min\ \{LDpP\}$
```

```
    Logically update the queues based on the model
end for
$c^*(t) = \bigcup_{k=1}^{K^t} c_k^*(t)$
Dispatch to $\alpha^*$
Update the Queues
End While
```

## 6. Experiments and Results

The effectiveness of the proposed method will be assessed in this section. Firstly, the simulation environment is described in detail, including the relevant settings and parameters. We then outline the benchmark methods that will serve as points of comparison. Finally, we present the simulation results based on the configuration presented earlier in this section.

### 6.1. Simulation Environment

To evaluate the proposed method and perform simulations, a simulator was designed and developed in the MatLab software in 2016. This simulator operates based on the model described in Figure 1.

For simulations, a test environment with 3 fog nodes, a remote cloud broker, and 20 IoT devices is used. Each node is identified by its processing power and has varying processing power and transfer rates. Request generation rates and data transfer rates of IoT devices are determined. Requests are assumed to be independent and follow a uniform distribution and Poisson process. Different request rates are considered for day and night, determined by Poisson processes with means of $\lambda_D$ and $\lambda_N$, respectively. Data transfer rates for each link are constant within a time slot but can change between slots. Channel conditions are randomly changed in simulations for realism. Each request has processing and communication requirements, assumed to be independent and follow exponential distributions with means $\gamma_1$ and $\gamma_2$, respectively.

To evaluate the proposed PARD method, we compare it with two widely used methods: Load Balancing (LB) and Random Distribution (Rnd). The LB method assigns requests to the node with the shortest queue, ensuring sufficient processing capacity. The Rnd method randomly distributes requests among available nodes during a time slot for comparison purposes.

### 6.2. Configuring the Simulation Environment

In this section we present the detailed configuration, specification and features of fog nodes. Furthermore, it will explain the parameters that describe the request arrival rate and its characteristics.

#### 6.2.1. Processing and Communication Capacity

Taking into account the simulation environment outlined in the preceding section, we are examining an environment comprising three fog nodes, one remote cloud node, and twenty IoT edge devices. Table 2 describes the attributes of the processing nodes (in cloud and fog) used in the simulation.

**Table 2: Nodes Configuration in Simulation Environment**

| ***Row*** | ***Node (Fog or Cloud)*** | ***Processing*** | ***Communication Capacity*** |
|---|---|---|---|

| | | *Capacity (MIPS)* | *(Mbps)* |
|---|---|---|---|
| *1* | Node_F1 | 1100 | U[2,5] |
| *2* | Node_F2 | 800 | U[2,5] |
| *3* | Node_F3 | 300 | U[2,5] |
| *4* | Node_Cloud | 4750 | 1 |

The specifications selection is determined by fog and cloud node characteristics. Fog nodes, located near edge devices, have lower processing power and higher sending rate benchmarks. Conversely, cloud nodes have higher processing power and lower send rate benchmarks.

#### 6.2.2. Request Arrival

As mentioned earlier, the rate at which requests arrive is assumed to follow a Poisson random process. During the day, the mean arrival rate is denoted as $\lambda_D$, while at night it is denoted as $\lambda_N$. The processing volume of each request, as well as the associated data volume, are assumed to follow exponential distributions. The means of these distributions are represented by $\gamma_1$ for the processing volume and $\gamma_2$ for the data volume. The specific values of $\lambda_D$, $\lambda_N$, $\gamma_1$, and $\gamma_2$, along with other relevant settings, are determined based on the information provided in Table 3.

**Table 3: Simulation Environment Configurable Parameters**

| *row* | $\lambda_D$ | $\lambda_N$ | $\gamma_1$ | $\gamma_2$ | *V* | *Time* | *iteration* |
|---|---|---|---|---|---|---|---|
| *1* | 0.2 | 0.1 | 0.5 | 0.5 | 6*10^4 | 1000 | 100 |

### 6.3. Evaluation Results

Based on the simulation environment described in the previous section, the configuration of the nodes described in Table II and the system parameters available in Table III, a comprehensive simulation with 100 iterations was performed. In the following, we will discuss the results obtained from these simulations.

The queue serves as a metric for measuring the number of requests waiting in line for service. A larger queue indicates increased congestion at the nodes. While minimizing the queue is desirable, it may come at the cost of factors such as energy consumption and efficiency. Hence, it is crucial to maintain the queue at an acceptable level while ensuring desirable efficiency.

In Figure 2 to Figure 5, we compare the performance of the proposed method with respect to queue length, service delay, the number of missed deadlines, and energy consumption, using benchmark methods as references.

#### 6.3.1. Queue Backlog

Figure 3 illustrates that the proposed method displays a lower queue backlog compared to the benchmark methods. In the proposed method, the accumulation rate is in the range of thousands, whereas the benchmark methods exhibit an accumulation rate in the range of hundreds of thousands. The reduced queue accumulation in the proposed method indicates enhanced stability when compared to the benchmark methods.

It is not surprising that the Rnd method is expected to exhibit unstable performance, as it distributes requests uniformly among processing nodes. Considering the different capacities of nodes, their different data transfer rates, and channel conditions at different times, a uniform distribution will cause excessive queue accumulation in slow nodes. On the other hand, although the SJQ method also distributes the load uniformly among the nodes, it leads to a higher queue accumulation compared to the proposed method, as it does not take into account the specific request conditions and system status. Therefore, we observe excessive queue accumulation (compared to the proposed method) in the processing nodes.

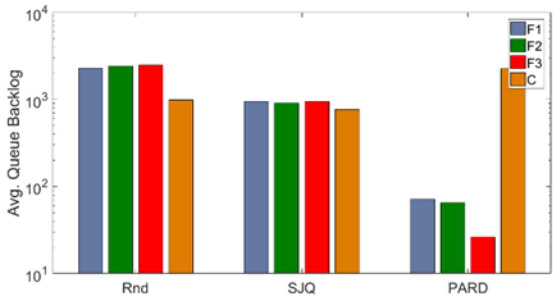

Figure 3 Comparing the queue backlogs of the proposed method with the benchmark methods.

### 6.3.2. Service Delay

The superior performance of the proposed methods in terms of service delay is shown in Figure 4. PARD outperforms the baseline methods with improvements of up to 15.6% and 24% in service delay compared to SJQ and Rnd, respectively.

### 6.3.3. Number of Deadline Misses

The performance of the proposed method in terms of the number of missed deadlines is shown in Figure 5. PARD exhibits superior performance compared to the baseline methods, as indicated in Figure 5.3. Specifically, considering the mentioned configurations, PARD achieves a reduction of 55.9% and 77.7% in the number of deadline misses compared to SJQ and Rnd, respectively.

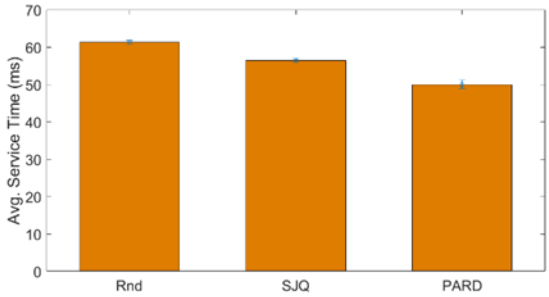

Figure 4 Comparing the service time for the proposed method and the benchmark methods.

### 6.3.4. Energy Consumption

The performance of the proposed method in terms of energy consumption is shown in Figure 6. PARD demonstrates superior performance in this aspect, as indicated by significantly lower energy consumption compared to the baseline methods. Specifically, PARD performs 66.6% and 83.3% better than SJQ and Rnd in terms of energy consumption.

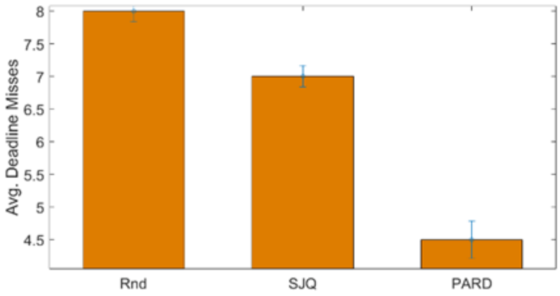

Figure 5 Comparing the number of deadline misses for the proposed method and the benchmark methods.

## 7. Analyzing the Obtained Results

This section summarizes the achievements and findings of this research. More specifically, in this section, we will analyze and evaluate in more detail the results that we have obtained from carrying out the simulations. We will analyze and discover meaningful relationships between various measured parameters.

### 7.1. Energy and Performance Tradeoff

As mentioned in the previous chapter, parameter V represents the level of attention given to the cost function versus queue stability. Therefore, the larger values for V show the greater focus on efficiency metrics (service delay and number of deadline violations) and the better the performance will be. However, on the other hand, this improvement in performance comes at a cost to the system, namely an increase in energy consumption. Therefore, a trade-off is observed between the performance metrics and consumption of energy. Increasing the value of parameter V will reduce service delay (and also the number of deadline violations), but we will also see an increase in energy consumption.

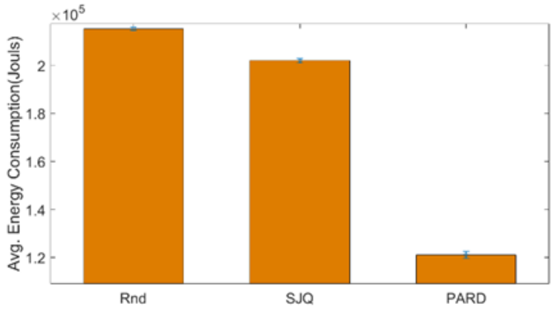

Figure 6 Comparing Energy consumption in the proposed method and the benchmark methods.

### 7.2. Parameter V

The V parameter is an important determinant of the performance of queueing systems. Parameter V controls the balance between the penalty function, in one side, and the objective and stability of real queues, in the other side. Higher value of V indicates a greater emphasis on efficiency metrics like service delay and deadline violations, resulting in improved performance.

However, enhancing the performance will increase the energy consumption in a trade-off. It is essential to consider this trade-off between efficiency and energy consumption when setting the value of parameter V. Finding the right balance between these two factors is essential for optimum system performance.

The specific requirements of the system will be a major factor in determining the value of the V parameter. For example, in systems with higher priority functions, a higher V value may be necessary to ensure that the functions can be performed effectively. Conversely, a lower V value may be preferred to minimize energy consumption in systems where energy consumption is a primary concern.

### 7.3. Scalability of the Proposed Methods

The system size has a direct influence on the system delay. The reason behind this impact is initiated from greater number of nodes in large system. Indeed, the dispersion and heterogeneity of the nodes require more time to move requests at different time intervals. Therefore, the controller may send requests to nodes that consume less energy at that moment, but are farther away from the requester. Consequently, we will see a grow in the number of missed deadlines. Additionally, with a growth in the number of processing nodes, energy consumption will also increase. However, on the other hand, an increase in the number of nodes has the potential to provide nodes with diverse capabilities, creating the opportunity for optimal use of energy resources.

## 8. **Conclusion**

This study explored cloud request distribution to reduce latency and maintain system stability. Stability involves limiting congestion in all queues and ensuring no queue is overwhelmed, resulting in desirable long-term system performance. The Lyapunov framework, a control theory method, was employed to evaluate and prove stability. By defining a Lyapunov function and controlling its changes, system stability can be maintained from an initially stable state.

Cloud nodes are geographically unevenly distributed, resulting in energy-consuming processing nodes and opportunities for innovation in power and energy management. Conventional methods face challenges due to node distribution, but intelligent power management methods are viable within the cloud architecture. However, heterogeneous fog nodes require alternative approaches. Utilizing clean energy resources at processing node locations is one such approach. Effective resource management should consider both processing and energy resources. Continuous power consumption by fog nodes complicates request distribution, requiring a balance between service delay, stability, and energy resource utilization. We propose a queue-based approach combined with Lyapunov optimization to address this sustainability challenge, presenting a dynamic allocation algorithm.

To assess the proposed method, a simulation environment was created and comprehensive tests were conducted under various system conditions. The proposed method's performance was compared with other existing methods. Several criteria, including service delay, deadline violations, energy consumption, and queue backlog, were selected to evaluate the proposed method. The simulated results show that the proposed method manages the requests in an effective way. The favorable performance of the proposed method under various conditions is confirmed by the evaluation and analysis..

In terms of future directions, one possible approach is to leverage containerization and micro-services architecture to enhance resource allocation and utilization efficiency. This can be achieved by dividing extensive applications into smaller, modular components, facilitating the distribution and scalability of these components across numerous devices and locations.

Aside from these technical solutions, it is essential to address broader organizational and policy matters to effectively support management of resource in FC.

## REFERENCES

[1] Koohang, A., Sargent, C. S., Nord, J. H., and Paliszkiewicz, J., "Internet of Things (IoT): From awareness to continued use," International Journal of Information Management, vol. 62, p. 102442, 2022.

[2] Ketu, S. and Mishra, P. K., "A contemporary survey on IoT based smart cities: architecture, applications, and open issues," Wireless Personal Communications, vol. 125, no. 3, pp. 2319-2367, 2022.

[3] Gupta, B. B. and Quamara, M., "An overview of Internet of Things (IoT): Architectural aspects, challenges, and protocols," Concurrency and Computation: Practice and Experience, vol. 32, no. 21, p. e4946, 2020.

[4] Jamsa, K., Cloud computing. Jones & Bartlett Learning, 2022.

[5] Marinescu, D. C., Cloud computing: theory and practice. Morgan Kaufmann, 2022.

[6] Bonomi, F., Milito, R., Natarajan, P., and Zhu, J., "Fog computing: A platform for internet of things and analytics," in Big data and internet of things: A roadmap for smart environments: Springer, 2014, pp. 169-186.

[7] Taneja, M. and Davy, A., "Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm," in Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on, 2017, pp. 1222-1228: IEEE.

[8] Hosseinioun, P., Kheirabadi, M., Kamel Tabbakh, S. R., and Ghaemi, R., "aTask scheduling approaches in fog computing: A survey," Transactions on Emerging Telecommunications Technologies, vol. 33, no. 3, p. e3792, 2022.

[9] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M., "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376, 2015.

[10] Desai, F. et al., "HealthCloud: A system for monitoring health status of heart patients using machine learning and cloud computing," Internet of Things, vol. 17, p. 100485, 2022.

[11] Sriram, G., "Edge computing vs. Cloud computing: an overview of big data challenges and opportunities for large enterprises," International Research Journal of Modernization in Engineering Technology and Science, vol. 4, no. 1, pp. 1331-1337, 2022.

[12] Xu, X., Liu, K., Xiao, K., Feng, L., Wu, Z., and Guo, S., "Vehicular fog computing enabled real-time collision warning via trajectory calibration," Mobile Networks and Applications, vol. 25, no. 6, pp. 2482-2494, 2020.

[13] Sharif, Z., Jung, L. T., Ayaz, M., Yahya, M., and Pitafi, S., "Priority-based task scheduling and resource allocation in edge computing for health monitoring system," Journal of King Saud University-Computer and Information Sciences, vol. 35, no. 2, pp. 544-559, 2023.

[14] Arshed, J. U. and Ahmed, M., "Race: resource aware cost-efficient scheduler for cloud fog environment," IEEE Access, vol. 9, pp. 65688-65701, 2021.

[15] Liao, H., Li, X., Guo, D., Kang, W., and Li, J., "Dependency-aware application assigning and scheduling in edge computing," IEEE Internet of Things Journal, vol. 9, no. 6, pp. 4451-4463, 2021.

[16] Wang, J., Liu, K., Li, B., Liu, T., Li, R., and Han, Z., "Delay-sensitive multi-period computation offloading with reliability guarantees in fog networks," IEEE Transactions on Mobile Computing, vol. 19, no. 9, pp. 2062-2075, 2019.

[17] Sheikh Sofla, M., Haghi Kashani, M., Mahdipour, E., and Faghih Mirzaee, R., "Towards effective offloading mechanisms in fog computing," Multimedia Tools and Applications, vol. 81, no. 2, pp. 1997-2042, 2022.

[18] Feng, C., Han, P., Zhang, X., Yang, B., Liu, Y., and Guo, L., "Computation offloading in mobile edge computing networks: A survey," Journal of Network and Computer Applications, p. 103366, 2022.

[19] Elarfaoui, A. and Elalami, N., "Optimization of QoS parameters in cognitive radio using combination of two crossover methods in genetic algorithm," Int'l J. of Communications, Network and System Sciences, vol. 2013, 2013.

[20] Namvar, N., Saad, W., Maham, B., and Valentin, S., "A context-aware matching game for user association in wireless small cell networks," in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 439-443: IEEE.

[21] Wang, J., Zhao, L., Liu, J., and Kato, N., "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," IEEE Transactions on emerging topics in computing, 2019.

[22] Kim, B., Cho, J., Jeon, S., and Lee, B., "An AHP-based flexible relay node selection scheme for WBANs," Wireless Personal Communications, vol. 89, pp. 501-520, 2016.

[23] Bairagi, A. K., Tran, N. H., Kim, N., and Hong, C. S., "QoS aware collaborative communications with incentives in the downlink of cellular network: A matching approach," in 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2016, pp. 1-6: IEEE.

[24] Saliba, A. J., Beresford, M. A., Ivanovich, M., and Fitzpatrick, P., "User-perceived quality of service in wireless data networks," Personal and Ubiquitous Computing, vol. 9, no. 6, pp. 413-422, 2005.

[25] Angelakis, V., Avgouleas, I., Pappas, N., Fitzgerald, E., and Yuan, D., "Allocation of Heterogeneous Resources of an IoT Device to Flexible Services," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 691-700, 2016.

[26] Ding, Y., Jin, Y., Ren, L., and Hao, K., "An intelligent self-organization scheme for the internet of things," IEEE Computational Intelligence Magazine, vol. 8, no. 3, pp. 41-53, 2013.

[27] Lin, R. et al., "Distributed optimization for computation offloading in edge computing," IEEE Transactions on Wireless Communications, vol. 19, no. 12, pp. 8179-8194, 2020.

[28] Kong, X. T. et al., "Cyber physical ecommerce logistics system: An implementation case in Hong Kong," Computers & Industrial Engineering, vol. 139, p. 106170, 2020.

[29] Shang, C., Bao, X., Fu, L., Xia, L., Xu, X., and Xu, C., "A Novel Key-value based Real-time Data Management Framework for Ship Integrated Power Cyber-Physical System," in 2019 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia), 2019, pp. 854-858: IEEE.

[30] Maddikunta, P. K. R. et al., "Incentive techniques for the internet of things: a survey," Journal of Network and Computer Applications, p. 103464, 2022.

[31] Malik, P. K. et al., "Industrial Internet of Things and its applications in industry 4.0: State of the art," Computer Communications, vol. 166, pp. 125-139, 2021.

[32] Chen, B., Wan, J., Shu, L., Li, P., Mukherjee, M., and Yin, B., "Smart factory of industry 4.0: Key technologies, application case, and challenges," Ieee Access, vol. 6, pp. 6505-6519, 2017.

[33] Jha, S., Tariq, U., Joshi, G. P., and Solanki, V. K., Industrial Internet of Things: Technologies, Design, and Applications. CRC Press, 2022.

[34] Bosi, I., Rosso, J., Ferrera, E., and Pastrone, C., "IIot Platform for Agile Manufacturing in Plastic and Rubber Domain," in IoTBDS, 2020, pp. 436-444.

[35] Tan, S. Z. and Labastida, M. E., "Unified IIoT cloud platform for smart factory," in Implementing Industry 4.0: Springer, 2021, pp. 55-78.

[36] Jiang, B., Li, J., Yue, G., and Song, H., "Differential Privacy for Industrial Internet of Things: Opportunities, Applications, and Challenges," IEEE Internet of Things Journal, vol. 8, no. 13, pp. 10430-10451, 2021.

[37] Oztemel, E. and Gursev, S., "Literature review of Industry 4.0 and related technologies," Journal of Intelligent Manufacturing, vol. 31, no. 1, pp. 127-182, 2020.

[38] Xu, L. D., Xu, E. L., and Li, L., "Industry 4.0: state of the art and future trends," International Journal of Production Research, vol. 56, no. 8, pp. 2941-2962, 2018.

[39] Vaidya, S., Ambad, P., and Bhosle, S., "Industry 4.0–a glimpse," Procedia Manufacturing, vol. 20, pp. 233-238, 2018.

[40] Serpanos, D. and Wolf, M., "Industrial internet of things," in Internet-of-Things (IoT) Systems: Springer, 2018, pp. 37-54.

[41] Tayeb, S., Latifi, S., and Kim, Y., "A survey on IoT communication and computation frameworks: An industrial perspective," in Computing and Communication Workshop and Conference (CCWC), 2017 IEEE 7th Annual, 2017, pp. 1-6: IEEE.

[42] Mahmud, R., Toosi, A. N., Ramamohanarao, K., and Buyya, R., "Context-aware placement of Industry 4.0 applications in fog computing environments," IEEE Transactions on Industrial Informatics, vol. 16, no. 11, pp. 7004-7013, 2019.

[43] Afrin, M., Jin, J., Rahman, A., Tian, Y.-C., and Kulkarni, A., "Multi-objective resource allocation for Edge Cloud based robotic workflow in smart factory," Future Generation Computer Systems, vol. 97, pp. 119-130, 2019.

[44] Verba, N., Chao, K.-M., Lewandowski, J., Shah, N., James, A., and Tian, F., "Modeling industry 4.0 based fog computing environments for application analysis and deployment," Future Generation Computer Systems, vol. 91, pp. 48-60, 2019.

[45] Lin, C.-C. and Yang, J.-W., "Cost-efficient deployment of fog computing systems at logistics centers in industry 4.0," IEEE Transactions on Industrial Informatics, vol. 14, no. 10, pp. 4603-4611, 2018.

[46] Chekired, D. A., Khoukhi, L., and Mouftah, H. T., "Industrial IoT Data Scheduling based on Hierarchical Fog Computing: A key for Enabling Smart Factory," IEEE Transactions on Industrial Informatics, 2018.

[47] Wan, J., Chen, B., Wang, S., Xia, M., Li, D., and Liu, C., "Fog computing for energy-aware load balancing and scheduling in smart factory," IEEE Transactions on Industrial Informatics, vol. 14, no. 10, pp. 4548-4556, 2018.

[48] Da Xu, L., He, W., and Li, S., "Internet of things in industries: A survey," IEEE Transactions on industrial informatics, vol. 10, no. 4, pp. 2233-2243, 2014.

[49] Mubeen, S., Nikolaidis, P., Didic, A., Pei-Breivold, H., Sandström, K., and Behnam, M., "Delay mitigation in offloaded cloud controllers in industrial iot," IEEE Access, vol. 5, pp. 4418-4430, 2017.

[50] Nain, G., Pattanaik, K., and Sharma, G., "Towards edge computing in intelligent manufacturing: Past, present and future," Journal of Manufacturing Systems, vol. 62, pp. 588-611, 2022.