Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Mobility-aware hierarchical fog computing framework for Industrial Internet of Things (IIoT)

Tariq Qayyum[1†], Zouheir Trabelsi[1*†], Asad Waqar Malik[2†] and Kadhim Hayawi[3†]

Abstract

The Industrial Internet of Things (IIoTs) is an emerging area that forms the collaborative environment for devices to share resources. In IIoT, many sensors, actuators, and other devices are used to improve industrial efficiency. As most of the devices are mobile; therefore, the impact of mobility can be seen in terms of low-device utilization. Thus, most of the time, the available resources are underutilized. Therefore, the inception of the fog computing model in IIoT has reduced the communication delay in executing complex tasks. However, it is not feasible to cover the entire region through fog nodes; therefore, fog node selection and placement is still the challenging task. This paper proposes a multi-level hierarchical fog node deployment model for the industrial environment. Moreover, the scheme utilized the IoT devices as a fog node; however, the selection depends on energy, path/location, network properties, storage, and available computing resources. Therefore, the scheme used the location-aware module before engaging the device for task computation. The framework is evaluated in terms of memory, CPU, scalability, and system efficiency; also compared with the existing approach in terms of task acceptance rate. The scheme is compared with xFogSim framework that is capable to handle workload upto 1000 devices. However, the task acceptance ratio is higher in the proposed framework due to its multi-tier model. The workload acceptance ratio is 85% reported with 3000 devices; whereas, in xFogsim the ratio is reduced to approx. 68%. The primary reason for high workload acceptation is that the proposed solution utilizes the unused resources of the user devices for computations.

Keywords: Hierarchical fog model, Industrial Internet of Things (IIoT), Resource management, Device utilization, OMNeT++

Introduction

The rapid growth in digital technology has brought a fast transition in almost every aspect of life. Real-time data access, automation, and device-to-device connectivity bring a rapid change in current industrial practices. In network science, scientists strive to bring intelligence closer to the source [1]. Therefore, the concept of fog computing has been proposed in recent trends. A conventional fog computing framework generally includes three layers; the lower level is the IoT device layer, where data is being generated. The next layer comprises fog brokers that deliver computation, storage, networking services, and the last cloud tier. The recent adoption of IoT is in the industrial environment, sometimes referred to as Industrial IoT (IIoT). It is the concept of effectively using the devices in such a way that they combine to give better yield [2]. In an industrial environment, resource utilization is a very common problem [3] where the few devices are overburdened compared to others. In the traditional industrial environment, most devices are static or move within a range. These devices stay in a predefined area; however, there are devices, such as robots and automobiles, that move freely inside the entire workshop.

The primary motivation behind this work is the technology adoption in IIoT, to facilitate delay-sensitive

†Tariq Qayyum, Zouheir Trabelsi, Asad Waqar Malik and Kadhim Hayawi contributed equally to this work.

*Correspondence: trabelsi@uaeu.ac.ae

[1] College of Information Technology, United Arab Emirates University, Al Ain 17551, Abu Dhabi, United Arab Emirates
Full list of author information is available at the end of the article

computation systems to manage the quality of services. Unfortunately, using the traditional cloud to support autonomous devices is not viable for many reasons, such as security, communication delay, and cloud data center scheduling policies. Also, the efficient resource utilization is another motive of this research. In IIoT, many resources get wasted due to poor resource management strategies.

Thus, the inception of highspeed wireless data rate and Multi-access Edge Computing (MEC) technologies impersonates a baseline for more advanced development for the IIoT environment [4]. A massive amount of data can be transferred to edge servers to train machine learning models, perform complex computations, and temporarily cache the data [5–8].

Furthermore, in existing work, fog nodes are placed at fixed locations to support the delay-sensitive applications [9, 10] or a federated approach is used for balancing workload [11–13]. Thus, fixed location deployment of a fog node can cause a single point of failure, and such solutions are not scalable.

In an IIoT environment, most of the resources always remain underutilized [14]; therefore, predicting the location of mobile nodes is very important to ensure the proper utilization of under-utilized resources [15]. Thus, the scalable approach can help improve the entire factory process, including assembly logistics and supply chain. Fog computing has emerged as the latest technology where edge and fog servers can help bring the computing infrastructure close to the IIoT devices and improve the Industry 4.0 architecture design [16].

Contribution – In this work, we removed the limitation of FogNetSim [17] where only selected devices can act as broker nodes to schedule the incoming computing requests. In FogNetSim, a framework followed the standard definition of fog computing where a selected number of devices located at the edge of the network act as fog nodes. However, such a system fails to perform well with many IoT devices. On the contrary, the current work focuses on scalability issues where every IoT device in the network can act as a fog node depending on the availability and certain constraints like energy level, computation power, etc. Also, the previous work focuses on the parallel placement of fog nodes at a second-tier, whereas this work proposed a hierarchical placement scheme. In the proposed multi-layer fog framework for IIoT, every device can work as an autonomous fog broker. Thus, the scalability issue raised in conventional fog networks is addressed with distributed systems. Here, we proposed the concept of multi-functional devices as fog nodes. Other salient features of the work are:

- Propose a multi-layer fog deployment framework for task scheduling and big data processing in an industrial environment.
- A probabilistic model is adopted to improve the efficiency of heretical placement of fog servers over traditional flat conventional placement in terms of computation and communication delay.
- A priority queuing technique is used to schedule the IIoT data and tasks in a multi-layer fog network.
- In the proposed scheme, the IoT device layer act as a fog device to perform complex computation tasks to minimize the communication overhead.
- A multi-layer scheduling algorithm that uses all layers to schedule incoming tasks.
- The localization module is the proposed work that enables the fog brokers to predict the location of mobile nodes.
- The proposed simulation framework is evaluated in terms of memory, CPU, communication delays, computation delays, and energy. It is further compared with other existing solutions regarding workload acceptance/completion ratio.

The rest of the paper is structured as follows: Section 2 covers the state of the art literature review about existing fog based IIoT frameworks; Section 3 cover the system model; system components are discussed in Section 4; results are discussed in Section 5.

Literature review

In this section, a state-of-the-art literature review is conducted to cover the existing fog solutions for IIoT, including the location-aware schemes to determine the location of moving nodes in an IIoT environment.

Fog computing in an industrial IoT – Kumar et al. [18] proposed a fog framework for IIoT networks that combines the technologies like blockchain and edge computing to solve the current IIoT problems like latency, task computation, and security. First, however, they evaluated the efficiency of the proposed framework in terms of network usage, power consumption, and latency with the simulation of a non-blockchain environment.

Chen et al. [19] proposed a Kronecker-supported fog-based optimized compression scheme for IIoT data that achieves better results. The proposed scheme first uses a k-means clustering algorithm to calculate the spatial correlation among IoT data to obtain better compression results with a low communication overhead. Then, the two-dimensional Kronecker-supported data compression mechanism at the fog node recovers data back to its original shape with high precision and accuracy. The communication overhead between fog and cloud is also minimized with this mechanism. Finally, an efficient algorithm is proposed to evaluate the simulation framework. The results show that the proposed scheme is energy efficient with good quality of service.

Chekired et al. [20] proposed a hierarchical technique for fog server placements for the IIoT. The requests are classified into two categories; high priority and low priority. The high-priority requests demand an urgent response. A workload scheduling algorithm is also proposed to offload requests to fog servers in different hierarchy tiers. The solution is evaluated using actual industrial data from the Bosch group and comparing the proposed solution with conventional strategies.

Liu et al. [21] proposed a multilevel indexing model for service discovery in the fog layer of IIoT. Service discovery is crucial because an efficient service discovery system can efficiently accommodate user requests. The proposed model is based on equivalence relation and named as "distributed multilevel (DM)-index model" for fog layer retrieval and service maintenance in IIoT to reduce redundancy and minimize retrieval and traverse time. It also narrows down the search space. The model is evaluated experimentally and theoretically and shows its effectiveness compared to the inverted index and sequence models.

Mubeen et al. [22] developed a prototype to offload controller tasks to fog or cloud in industrial control systems. Many experiments are performed to instigate the interplay of fog computing, cloud computing, and the IIoT. A mitigation mechanism is also applied to reduce network delay when controllers are offloaded to cloud or fog infrastructures.

Chen et al. [23] proposed an energy-efficient offloading for IIoT in fog network environments. The purpose is to minimize the energy consumption while offloading dynamic computation requests to the fog layer. The energy minimization computation offloading problem is formulated with energy, delay, and other network parameters. However, an algorithm is proposed to address the optimization problem with joint optimization of offloading ratio and transmission time. The dynamic voltage scaling technique is also used with the above solution to reduce energy consumption during offloading. The proposed solution jointly optimized transmission time, local CPU, transmission power, and offloading ratio.

Yu et al. [24] proposed a secure data deletion technique in industrial fog environments. However, this area is relatively less explored. This research proposes a framework where the IoT devices, fog, and cloud combine to form an industrial environment. Further, better control of the data is proposed in the fog-cloud architecture of IIoT. The proposed protocol takes advantage of the feature of attribute-based encryption. The theoretical evaluation shows good performance with the proposed protocol.

Mukherjee et al. [25] formulated the problem of task offloading in fog computing for the IIoT. In IIoT, the latency-driven applications are very challenging because, most of the time, IoT sensors work in an automated environment, and the control signal with minimum latency is necessary for such environments. The proposed strategy is evaluated through simulations, and the results show that the proposed strategy is effective and scalable.

Fu et al. [26] proposed a data storage and search scheme for IIoT utilizing fog-cloud technologies. However, the IIoT devices are placed in isolated and remote areas and hence are vulnerable. In this proposed technique, the data is first processed at edge servers, delay-sensitive data is stored locally in edge servers, and then data is processed in cloud servers and stored. The simulation results proved the efficiency of the proposed scheme.

Aazam et al. [27] proposed a fog-based framework that uses industry 4.0 where many IoT devices, machines, business processes, appliances, and personals interact with each other and generate a massive amount of data. To process this data in a time-sensitive manner, they proposed a fog-based solution where a middle layer called fog communicate with all devices and process data at the edge of the industry. Many use cases are also presented, and research changes are discussed.

Lin et al. [28] proposed a cost-efficient strategy for fog servers deployment in Industry 4.0 at logistic centers. The work investigates the placement of fog servers, gateways, edge servers, sensors, and clouds in Industry 4.0 to minimize the deployment cost. This NP-hard problem of facility location is also solved with a metaheuristic algorithm that uses a genetic algorithm to enhance computational efficiency and a discrete monkey algorithm to find quality solutions.

To summarize, the existing fog solutions focus on the horizontal placement of the resources that can lead to high delay when the tasks are large and split into multiple smaller tasks. Also, they do not focus on the energy, mobility, and IIoT devices aspects of the system.

Localization – In this section, we cover the literature review to explore the existing techniques used to find the location of mobile nodes in a fog computing platform. In a fog-cloud integrated environment, static and mobile IoT devices generate data to be processed at fog nodes or cloud; therefore, location is an essential decision parameter.

Chen et al. [29] proposed a weighted factors localization algorithm for fog computing. The proposed solution includes both specific and general localization. The evaluation is performed by comparing the proposed

Table 1 Fog Simulators Comparison

Framework	Platform Independent	Open Source	Fog Placement	Mobility & Localization Support	External Algorithm	Energy	Fog Federation	Cloud Integration	IIoT Devices as fog
SysML4IoT [34]	✓	✓	H	x	x	✓	x	✓	x
CrowdSenSim [43]	✓	✓	H	Limited	x	✓	✓	✓	x
DPWSim [35]	✓	✓	H	x	x	x	x	✓	x
SmartSim [36]	✓	✓	H	x	x	✓	x	✓	x
DISSECT-CF-IoT [38]	✓	✓	H	Limited	x	✓	x	✓	x
SimIoT [39]	✓	✓	H	-	x	NA	x	✓	x
RECAP [40]	✓	✓	H	x	x	✓	x	✓	x
MobIoTSim [41]	x	✓	-	x	x	NA	x	✓	x
BlockEdge [18]	✓	✓	H &V	x	x	✓	x	✓	x
iFogSim [42]	✓	✓	H &V	x	x	✓	x	✓	x
Abuhasel et al. [37]	✓	✓	H	x	x	x	x	✓	x
FogNet-Sim++. [17]	✓	✓	H	Limited	✓	✓	x	✓	x
Proposed	✓	✓	H &V	✓	✓	✓	✓	✓	✓

H horizontal or distributed, *V* vertical or hierarchical

algorithm with the other two algorithms in a simulation environment, and positive results are observed.

Guidara et al. [30] investigated the localization of mobile nodes in an indoor environment. The wireless nodes are placed inside a building or a premise, and these IoT devices collect data. The data is sent to a central processing node called fog, where the position of the unknown node is estimated. The proposed algorithm finds the location of a node with the shortest delay and without incorporating powerful processing nodes.

FogLight [31] is a localization solution for IoT devices that depends on visible light and relies on spatial encoding. Spatial encoding is produced when mechanical mirrors are flipped based on binary images inside a projector. It employs simple light sensors that can be used with gas meters, light switches, or thermometers with a discoverable location. In the proposed solution, the sensors units can perform localization with high accuracy and with minimum processing overhead and computation efficiency, finding the location of any low-power IoT device. The results show that FogLight finds a device's location with an accuracy of 1.8 mm.

Femminella et al. [32] proposed a distributed signaling protocol for service function localization. The different functions of the protocol include the peer discovery in the transport layer and signaling distribution which are then divided into two parts of signaling delivery, downstream and reverse path. Finally, the protocol is evaluated via natural experiments.

Bhargava et al. [33] Proposed a fog-based localization solution for ambient assistant personnel. The proposed system is a low-cost wsm-based wearable device and cloud gateway to find ambient assisted living locations. Using the given topology information, the distance covered by a device is calculated with direction values. The proposed algorithm is evaluated in both indoor and outdoor environments.

To conclude this discussion, many existing solutions ignore important factors like device-to-device communication, mobility, energy consideration, fog federation, and fog placement that can improve the performance of an IIoT-based simulation framework. The work proposed in [34–37] lacks mobility support, and allows only horizontal fog placement. Whereas, none of [17, 18, 34–42] support fog federation and IIoT devices as fog devices. A detailed comparison of these features is given in Table 1. The proposed work focuses on fog placement strategy in both directions (distributed and hierarchical), uses some IIoT devices as fog devices that meet certain criteria, finds the location of unknown mobile IIoT devices to improve reliability, and incorporate distributed fog locations that can work in a federated architecture to provide resources on demand.

System model

The abstract view of system architecture is shown in Fig. 1 where the IIoT system is divided into multiple layers. The first layer comprises U_n number of IoT sensors, static, and mobile devices. These devices are categorized into two classes, user devices that generate data and request computations and computationally strong devices that also volunteer their resources

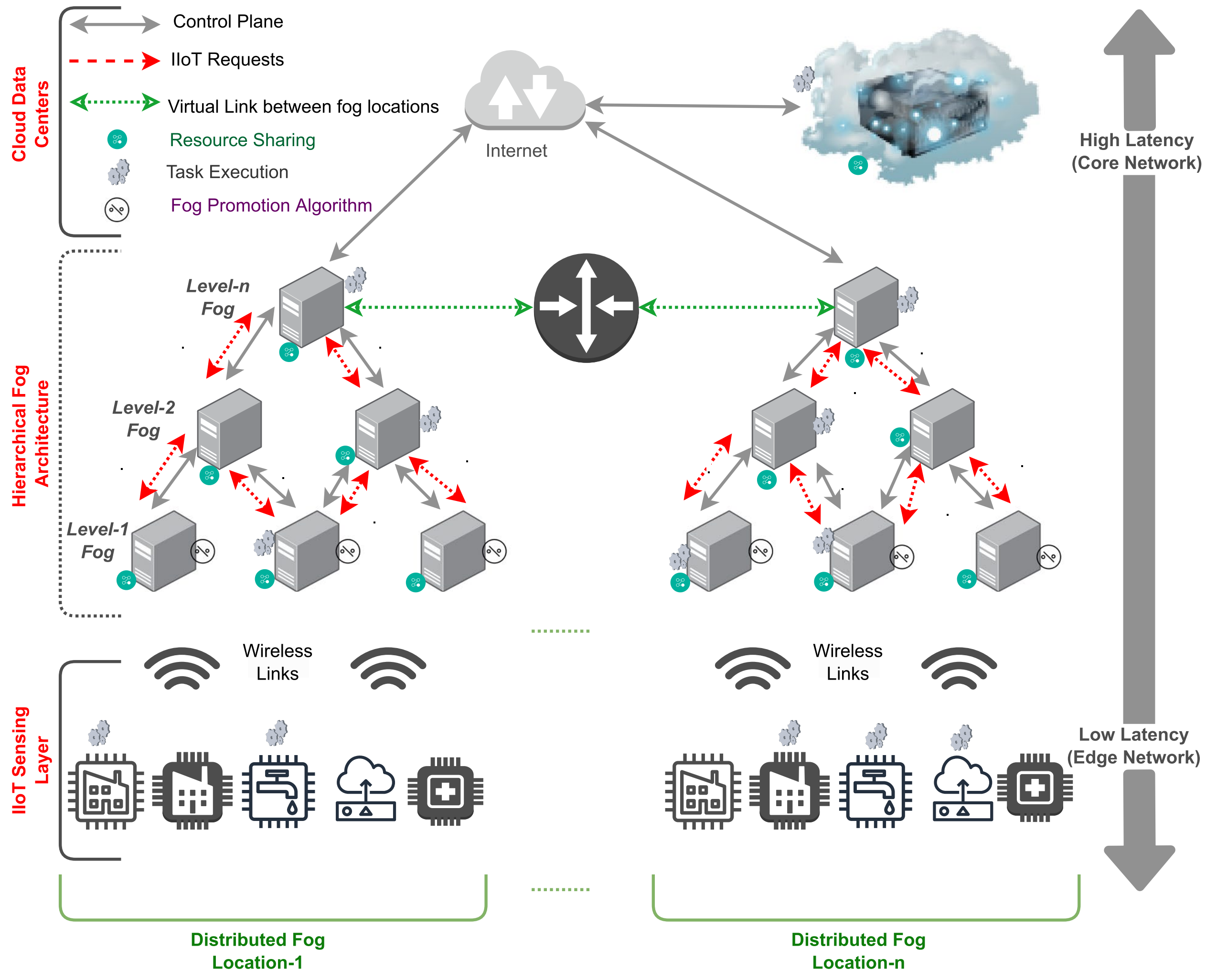

Fig. 1 System architecture – IIoT sensors and devices offload tasks via a wireless link between devices and fog nodes. The fog nodes are placed in a hierarchical model classified into multiple levels. Level-1 fog nodes are the intelligent nodes that decide whether to offload tasks to volunteer devices, same-level fog, higher-level fog, or a nearby fog location

to act as fog nodes. The second layer comprises fog devices that are classified into n number of fog devices placed in a hierarchical manner that receive computation requests from the lower layer and hierarchically execute tasks. The topmost layer is the cloud layer that performs complex executions. The module interactions of the proposed framework is explained in Fig. 2. There is L_m number of fog locations that are connected to a cloud data center. These fog nodes share the distributed workload in a federated way. The summary of notations is given in Table 2.

The M/M/c queuing model is employed [44] where the system has multiple servers that contribute to executing tasks in the queue. Using the Poisson process, the arrival rate λ is calculated as in Eq. 1

$$\lambda_T = \sum_{F_k \in F_n}^{n} \lambda_{F_k}. \tag{1}$$

Where λ_{F_k} is the average arrival rate at k_{th} fog node F.

Execution delay – In the proposed system, there are multiple options to execute that task, and the system chooses the best available option. Initially, the priority is to execute the task on a local static/mobile device. However, the task is offloaded to lower-level fog nodes due to limited resources. This level of fog nodes is composed of volunteer devices willing to share their resources. However, if all the fog devices are busy and the system fails to meet the required deadline, the task is further offloaded to the second-tier fog nodes. Similarly, if the task has a delayed deadline, it is offloaded to the cloud otherwise. In

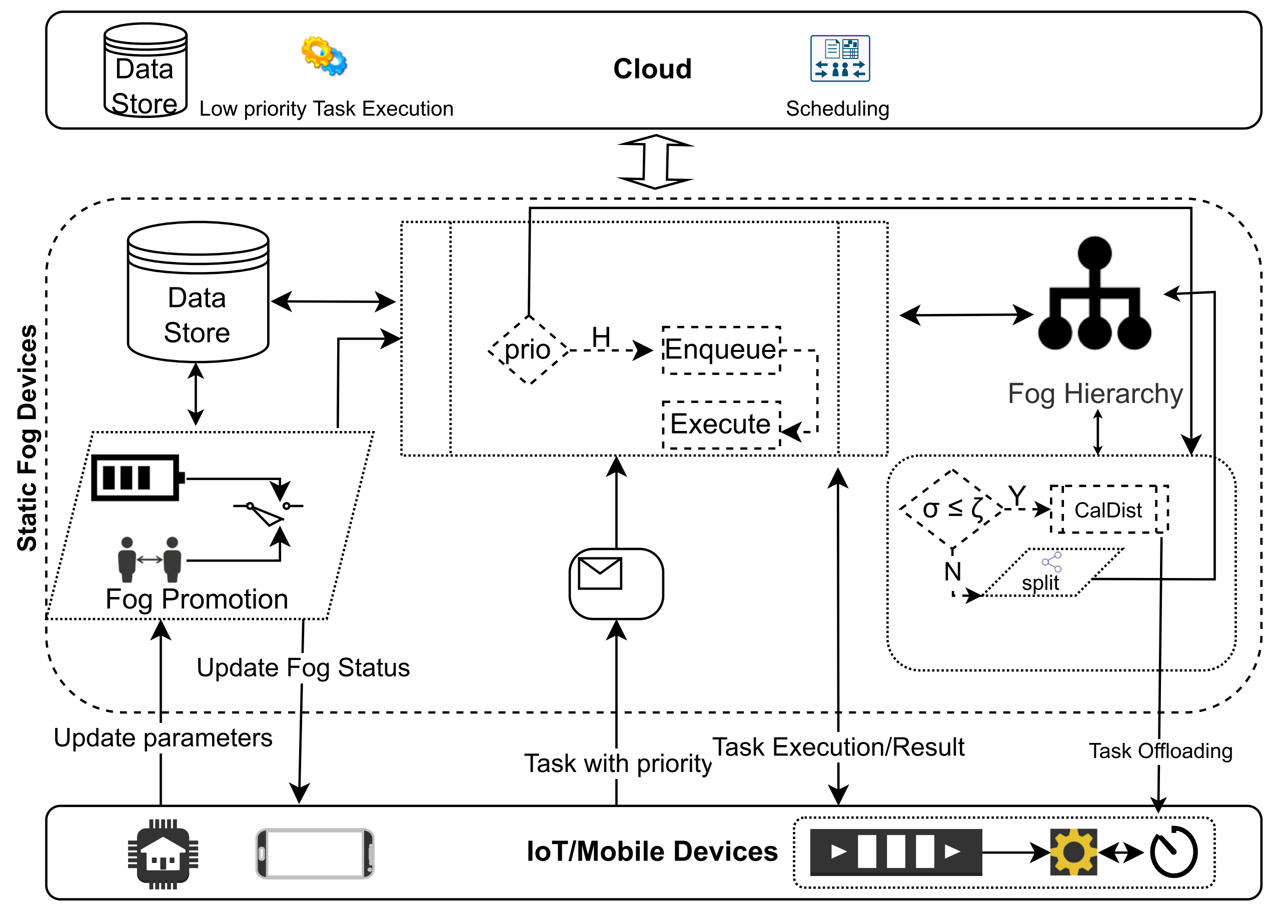

Fig. 2 Architecture of the proposed framework illustrating module interactions

case of task can be sub-divided into multiple independent sub-tasks where part of it can be executed locally and other to fog device; the execution delay δ is calculated as:

$$\delta = \max\{\delta_L, \delta_O\}. \tag{2}$$

Where δ_L and δ_O are the execution delay for locally executed, and remotely executed tasks respectively. These delays are calculated separately as follows:

1. **Local execution delay** – The local execution delay is computed based on the local tasks available in the device queue. In such cases, there is no transmission delay; thus, local execution delay is computed as:

$$\delta_L = \frac{\sum_{k \in \mathcal{K}} \sigma_k^L \upsilon_k}{\Upsilon_L}, \tag{3}$$

where Υ_L is the execution rate at a local device in terms of millions of instructions per second (MIPS), and $\mathcal{K}$ is tasks in the local queue.

2. **Remote execution delay** – The remote execution delay is computed based on the offloaded tasks. Further, the brokers share the workload with other fog nodes in the hierarchy and cloud servers. According to [25], additional delays can be observed like wireless uplink delay, wireless downlink delay, and network delay.

$$\delta_O = \{\delta_{FU} + \delta_{CU} + \max\{\delta_{FP}, \delta_{fU} + \delta_{CP} + \delta_{fD}\} + \delta_{FD} + \delta_{CD}\}. \tag{4}$$

where δ_{CU} and δ_{FU} are wireless uplink transmission latency of tasks processing in cloud and fog respectively. Correspondingly, δ_{CD} and δ_{FD} are their respective wireless downlink transmission latency. Furthermore, δ_{CP} and δ_{FP} are the processing latency of fog and cloud. Moreover, δ_{fD} and δ_{fU} are the downlink and uplink fronthaul latency of tasks processing in cloud.

According to [45] the wireless uplink transmission latency for cloud δ_{CU} and fog δ_{FU} are calculated as follows:

Table 2 Summary of Notations

Sr.	Symbol	Definition
1	U	List of user IIoT devices
2	V	List of volunteer IIoT devices
3	n_u	No. of devices in List U
4	n_v	No. of devices in List V
5	${V_c}^i$	Computation capacity of ith volunteer device
6	Υ_L	Local execution rate on mobile device
7	υ_k	Execution required by task σ_k
8	σ_s^i	The workload received at s_{th} server in level-i.
9	σ_k^L	The workload executed locally
10	σ_k^F	The workload executed in Fog
11	σ_k^C	The workload executed in Cloud
12	${\zeta_s}^1$	The capacity of s_{th} server in level-i.
13	Off_i	The offloaded workload
14	δ	The execution delay
15	F_n	Dedicated Fog Nodes
16	W_n	Gateway Nodes
17	U_n	User nodes seeking computation
18	V_n	User nodes volunteer resources
19	$\mathcal{D}_i^F$	Computation delay (queue + service)

$$\delta_{\mathrm{FU}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{F}} B_k^{\mathrm{in}}}{\beta_{\mathrm{U}}}, \delta_{\mathrm{CU}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{C}} B_k^{\mathrm{in}}}{\beta_{\mathrm{U}}}, \tag{5}$$

Similarly, the wireless downlink transmission latency for cloud and fog are calculated as follows:

$$\delta_{\mathrm{FD}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{F}} B_k^{\mathrm{out}}}{\beta_{\mathrm{D}}}, T_{\mathrm{CD}} = \frac{\sum_{k\in\mathcal{K}} \sigma k^{\mathrm{C}} B_k^{\mathrm{out}}}{\beta_{\mathrm{D}}}. \tag{6}$$

The β_{D} and β_{U} are the wireless transmission rates [46] for uplink and downlink that can be calculated as follows:

$$\begin{aligned} \beta_U &= \Upsilon_U \log_2\left(1 + \frac{\rho_U \iota_U^2}{\Lambda_0}\right), \\ \beta_D &= \Upsilon_D \log_2\left(1 + \frac{\rho_F \iota_D^2}{\Lambda_0}\right) \end{aligned} \tag{7}$$

Where ρ_{U} and ρ_{F} are the transmission power of mobile devices and fog devices, respectively. The ι_{U} and ι_{D} are the wireless gain for uplink and downlink, which is constant and does not change with time. Moreover, Υ_{U} and Υ_{D} are the bandwidths of uplink and downlink. Finally, the noise power is Λ_0. When the computation speed (MIPS) at cloud and fog is defined as ζ_{C} and ζ_{F}, the cloud processing latency and fog processing latency can be calculated by [47].

$$\delta_{\mathrm{FP}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{F}} D_k}{\zeta_{\mathrm{F}}}, \delta_{\mathrm{CP}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{C}} D_k}{\zeta_{\mathrm{C}}}. \tag{8}$$

Finally, with ξ_{F} as the fronthaul capacity [48], the downlink and uplink fronthaul transmission latency tasks processed in the cloud are calculated as:

$$\delta_{\mathrm{fU}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{C}} B_k^{\mathrm{in}}}{\xi_{\mathrm{F}}}, \delta_{\mathrm{fD}} = \frac{\sum_{k\in\mathcal{K}} \sigma_k^{\mathrm{C}} B_k^{\mathrm{out}}}{\xi_{\mathrm{F}}}. \tag{9}$$

Hence, the total application latency will become

$$\delta = \max\{\delta_{\mathrm{L}}, \delta_{\mathrm{F}}, \delta_{\mathrm{C}}\}, \tag{10}$$

and δ_{F}, the fog execution latency, and δ_{C}, the cloud execution latency are given as:

$$\begin{aligned} \delta_{\mathrm{F}} &= \delta_{\mathrm{fU}} + \delta_{\mathrm{CU}} + \delta_{\mathrm{FP}} + \delta_{\mathrm{FD}} + T_{\mathrm{CD}} \\ \delta_{\mathrm{C}} &= \delta_{\mathrm{FU}} + \delta_{\mathrm{CU}} + \delta_{\mathrm{fU}} + \delta_{\mathrm{CP}} + \delta_{\mathrm{fD}} + \delta_{\mathrm{FD}} + \delta_{\mathrm{CD}}. \end{aligned} \tag{11}$$

Probabilistic model – A probabilistic model is presented in this section to evaluate the probability that the servers located at the lower level in the hierarchy can execute and schedule the received workload to the higher level. The $\sigma_1, \sigma_2, ...\sigma_n$ are random and independent distributed variables. The probability that level-1 servers can successfully serve and offload the assigned workload is given as [49]: $\mathbb{P}\left({\sigma_i}^1 \leq {Cap_i}^1\right)$ which is calculated as shown below:

$$\mathbb{P}\left({\sigma_1}^1 \leq {\zeta_1}^1, \ldots, {\sigma_s}^1 \leq {\zeta_s}^1\right) = \prod_{i=1}^{s^1} \mathbb{P}\left({\sigma_i}^1 \leq {\zeta_i}^1\right). \tag{12}$$

Where σ_s^1 is the workload received and ${\zeta_s}^1$ is the capacity of s_{th} server in level-1. According to the capacity of each server in level-1, there are two scenarios.

1. If ${\sigma_i}^1 \leq {\zeta_i}^1$, which means the workload received at i_{th} server in level-1 is less than the computation capacity of that server, the offloaded volume of the workload is null.
2. If ${\sigma_i}^1 > {\zeta_i}^1$, which means the amount of workload is greater than the capacity of the server, the amount of workload that the server will offload to the level-2 server is:

$$Off_i = {\sigma_i}^1 - {\zeta_i}^1. \tag{13}$$

Hence the Cumulative Distribution Function (CDF) of the workload is given as:

$$\mathbb{F}_{O_i{}^1}\left(\mathrm{Cap}_i{}^1\right) = \mathbb{P}\left(O_i{}^1 \leq \mathrm{Cap}_i{}^1\right) = \begin{cases} \mathbb{P}\left(\sigma_i{}^1 \leq \mathrm{Cap}_i{}^1 + \zeta_i{}^1\right) & \text{if } \mathrm{Cap}_i \geq 0 \\ 0 & \text{otherwise} \end{cases}. \tag{14}$$

The total workload offloaded by the level-1 servers to level-2 servers is calculated as

$$\sigma_{total} = \sum_{i=1}^{s^1} Off_i. \tag{15}$$

Location estimation – The mobile nodes follow different mobility models and can be localized in the physical area. Therefore, the location of an unknown mobile node can be estimated according to the location of some known nodes, and these known nodes might be static nodes or nodes whose locations are already estimated. Let's say that the coordinates of an unknown node u are (x_u, y_u). The coordinates of a known node k are x_k, y_k, and there are n nodes with known coordinates. The location of this unknown node is calculated as [50]:

$$(x_u - x_k)^2 + (y_u - y_k)^2 = d_{ku}^2 \tag{16}$$

where d_{uk} is the distance between unknown node u and k_{th} known node.

fog n number of known nodes

$$\begin{cases} -2x_1x_k - 2y_1y_k + x_k^2 + y_k^2 = d_{k1}^2 - x_1^2 - y_1^2 \\ -2x_2x_k - 2y_2y_k + x_k^2 + y_k^2 = d_{k2}^2 - x_2^2 - y_2^2 \\ \cdots\cdots\cdots\cdots \\ -2x_nx_k - 2y_ny_k + x_k^2 + y_k^2 = d_{kn}^2 - x_n^2 - y_n^2 \end{cases} \tag{17}$$

Suppose $Q_u = x_u^2 + y_u^2$, $R_k = x_k^2 + y_k^2$, $S = [x_k, y_k, R]^T$,

$$X = \begin{pmatrix} -2x_1 & -2y_1 & 1 \\ -2x_2 & -2y_2 & 1 \\ \vdots & \vdots & \vdots \\ -2x_n & -2y_n & 1 \end{pmatrix},$$

$$Y = \begin{pmatrix} d_{k1}^2 - Q_1 \\ d_{k2}^2 - Q_2 \\ \vdots \\ d_{kn}^2 - Q_n \end{pmatrix},$$

In matrix expression:

$$XC = Y \tag{18}$$

and

$$C = (X^T X)^{-1} XY \tag{19}$$

Finally, the position of unknown mobile node k in 2-dimensional plane is given as:

$$(x_k,\ y_k) = (C(1),\ C(2)) \tag{20}$$

Proposed framework

The proposed simulation framework provides an infrastructure where mobile and static nodes can become part of the simulation. The device can request resources from the other devices that have unused resources. Let us assume there is a n number of user devices that seek resources from m dedicated fog devices, and r represents the number of fixed brokers that receive these requests. The broker nodes find the most suitable fog device to offload the incoming request. Considering the device-to-device communication, fog devices send the result directly to the user device to avoid extra delay. The proposed framework allows end devices to volunteer resources and acts as fog nodes. Thus, the user node in the idle state offers its resources to be used for the other devices. This scheme is adopted to resolve the problem of resource under-utilization in an industrial environment. This dynamic transition from user node to fog node depends on several factors: resource availability, mobility, speed, acceleration, energy, and other contextual information.

Design components – Multiple layers characterize the architecture of the proposed framework. The core components of the proposed framework are discussed below.

IIoT device layer – The bottom-most layer is the device layer, also termed as IIoT-layer. It contains all devices available in an industrial environment, such as sensors, cyber-physical systems, robots, and automobiles. These devices have limited resources and communication range; some are placed at fixed locations, whereas others can freely move within the environment.

IIoT resource layer– In the proposed work, we have introduced this layer to provide cost-effective resource sharing. This is a virtual layer containing all IIoT devices that volunteer their resources. Thus, the IIoT device becomes part of the resource layer on accepting the resource sharing model. These IIoT devices share their idle resources to enhance system performance. As more devices become a part of this layer, it improves the quality of service and reduces the impact of the communication network.

IIoT fog broker – This layer is composed of dedicated fog servers/nodes placed hierarchically in multiple sub-layers to facilitate complex tasks. These servers/nodes have significantly high computation and storage resources and are more suitable for intelligence training models. These devices can also act federated to reduce the learning curve significantly.

Cloud-layer: This layer is composed of the cloud data center for batch processing, and storage of data for a longer duration.

In the proposed framework, the quality of service is maintained through the volunteer nodes. Both static and mobile nodes volunteer their unused resources to their nearby broker at level-0. The volunteer nodes are accepted or rejected based on specific criteria initially set at the start of simulation as given in Algorithm 1. These criteria include a requested node's minimum energy level and mobility. Energy is the most predominant factor for consideration because a volunteer node must have efficient energy to be promoted as a fog node. If the incoming node meets the energy criteria, the mobility is checked on second priority. However, for static nodes, distance is computed for acceptance. For the mobile node, direction plays a significant role in accepting the proposal. The node is rejected if moving away from the broker node. The accepted proposals are added to the list of available resources.

```
Require: $V$: list of volunteer nodes, $\sigma$: incoming request, $\Omega$: threshold energy
    to become fog, $L_k$: list of nodes with known locations
Ensure: Accept or Reject
 1: $\chi \leftarrow \sigma_{ParentNode}$
 2: $\rho \leftarrow \chi_{position}$                          ▷ get coordinates of node (x,y)
 3: if $\chi < \Omega$ then                                     ▷ energy is less than threshold
 4:     SendAck(Reject)                                         ▷ send rejection acknowledgment
 5: else
 6:     $d_{\chi,b} \leftarrow CalcDistance(\rho, b)$           ▷ calculate distance between node and
    broker eq 16
 7:     if $\chi_{type}$ is static then
 8:         if Signal Strength in Range then
 9:             $V \leftarrow \chi$                             ▷ accept & add in sorted list
10:         end if
11:     else
12:         if $b_{\chi}^{i-1} > b_{\chi}^{i}$ then
13:             SendAck(Reject)
14:         else
15:             $V \leftarrow \chi$                             ▷ accept & add in sorted list
16:         end if
17:     end if
18: end if
```

Algorithm 1 Fog Promotion Algorithm

Task execution algorithm – The fog node, volunteer IoT device, or a dedicated fog node receives the incoming workload, which is initially stored in the input queue. The node pop the workload from the top of the queue executes it and sends the result back to the requesting node or fog broker. According to [51] this process is elaborated in Algorithm 2.

```
Input $f$: fog node; $\sigma$ incoming job
Output None
 1: $\mathbb{T} \leftarrow 0$
 2: while true do
 3:     if $f.State$ is receiving then
 4:         $\mathcal{W} \leftarrow Receive(\sigma)$            ▷ Receive incoming workload
 5:         $Queue \leftarrow \mathcal{W}$                      ▷ Put incoming workload into queue
 6:     end if
 7:     if $queue \neq \emptyset$ then
 8:         $\mathcal{J} \leftarrow Queue.pop()$
 9:         $\mathcal{R} \leftarrow Execute(\mathcal{J})$       ▷ Execute job
10:         Send($\mathcal{R}$)                                 ▷ Send result to requesting fog node
11:     end if
12:     if $\mathbb{T}$ Expires then
13:         $\mathbb{T} \leftarrow 0$                           ▷ Reset timer
14:     end if
15: end while
```

Algorithm 2 Task Execution at Fog

Resource sharing algorithm – The user nodes send two types of workload, high priority and low priority workloads. The broker nodes at the first-level fog nodes in the hierarchy manage two types of queues for incoming tasks: low priority and high priority. Also, the broker nodes manage lists of volunteer fog devices sorted according to their computational resources. Further, the broker manages a two-dimensional list where each row represents a fog level, and each column represents fog nodes at that level. This list is also sorted according to each fog node's computational resources. The volunteer nodes and fog nodes periodically send beacons to broker nodes, updating the status of its resources, whereas the broker node updates the lists dynamically as shown in Algorithm 3. In this algorithm, lines 1-9 represent the high-priority workload that is tried to execute at the same broker node that minimizes the delay. Line 10-20 indicates the low-priority tasks that require QoS is the best-effort system struggles to place it at the nearest volunteer node. This is because the volunteer nodes have mobility, and there are chances of connection loss and retransmission; hence it might add additional delay. Line 21-34 covers the guaranteed quality of service through offloading the task to the dedicated fog nodes in the hierarchy.

```
Input $\sigma$ incoming job
$\sigma_{priority}$ Priority of incoming job
$L_G$ List of fog locations
$k$ Number of levels
$F[m][n]$ List of fog nodes
$V_k$ List of volunteer nodes in level $k$
Output Optimal solution
 1: if ($\sigma_{priority} == High$) then
 2:     $level \leftarrow 0$
 3:     if ($\sigma \leq \zeta_s^1$) then
 4:         $High - Enqueue(\sigma)$
 5:     else
 6:         Calculate $Off(\sigma)$                  ▷ Calculate offload task amount eq 13
 7:         $f \leftarrow F[0][n].Pop()$             ▷ Pop element from top of level-1 fog list
 8:         $Offload(Off, f)$                        ▷ Offload remaining workload
 9:     end if
10: else                                             ▷ ($\sigma_{priority} == Low$)
11:     if ($\sigma_{QoS} == 1$) then                ▷ When required QoS is best effort
12:         $volunteer \leftarrow V.Pop$
13:         $d \leftarrow +\infty$                   ▷ Set initial distance
14:     for each $v \in V$
15:         $dist \leftarrow CalculateDistance(\sigma.parent, v)$
16:         if ($dist \leq d$) then
17:             $d \leftarrow dist$
18:         end if
19:     end
20:         $Offload(\sigma, v)$                     ▷ Offload workload to volunteer with minimum distance
21:     else                                         ▷ When guaranteed QoS is required
22:         $level \leftarrow 1$
23:         $Off \leftarrow \sigma$
24:         while ($level \leq m$) do
25:             $fog \leftarrow 0$
26:             while ($fog \leq n$) do
27:                 if ($\sigma \leq \zeta_{fog}^{level}$) then
28:                     $F[level][fog]_{Low-Enqueue(\sigma)}$
29:                     Break;
30:                 else
31:                     $F[level][fog]_{Low-Enqueue(\sigma)}$
32:                     Calculate $Off(\sigma)$      ▷ Calculate offload workload amount 13
33:                     $fog \leftarrow fog + 1$
34:                 end if
35:             end while
36:             $level \leftarrow level + 1$
37:         end while
38:     end if
39: end if
```

Algorithm 3 Resource Sharing Algorithm

Table 3 Simulation parameters

#	Parameter	Value/Description
1	Fog Locations	2
1	N. of Levels	3
2	Fist Level Fogs(Brokers)	20-25
3	Level-2 & Level-3 Fogs	10-100
4	User Nodes	50-500
5	Volunteer User Nodes	100-300
6	Fog Node Compute Capacity	1200MIPS
7	Workload Inter-arrival	0.5-1.5s
8	Workload Size γ	2048
9	Cloud Data-center(s)	1
10	Level-1 App Name	BrokerApp
11	Fog App Name	ComputeFogApp
12	Volunteer User App	VolunteerApp
13	User App	IIoTmqttApp
15	Broker queueType	DropTailQueue
16	Fog queueType	FIFO
17	Radio Transmitter Power	3.5-5mW
18	WLAN Channel	54Mbps

Table 4 System specifications

#	Component	Value/Version
Computer (Memory & CPU)		
1	Processor	Intel(R) *Core*^{TM} i5 1 GHz
2	Core(s)	4
3	Threads	8
4	Memory	16 GB
5	Operating System	Ubuntu 16.04 LTS
Graphics & Display		
6	Resolution	1920x1080 pixels
7	OpenGL Renderer	Mesa DRI Intel(R) UHD
8	X11 Vendor	The X.Org Foundation
Tools & Technologies		
9	Omnet++	4.6
10	INET	3.2.4

Evaluation

The proposed simulation framework is benchmarked on Ubuntu 16.04 LTC with variable sensors and IIoT devices that offload workloads defined in terms of Millions of Instructions Per Second (MIPS). The workload is offloaded to the level-1 fog device that is referred to as fog brokers. However, the IIoT devices also volunteer for their resources to level-1 fog nodes. The level-1 fog nodes promote these IIoT devices to fog nodes depending on multiple factors, including residual energy, computing power, and available storage. The fog nodes are placed on three levels. In the rest of the fog levels, nodes are enriched regarding computation resources. The last level fog nodes are connected to a cloud data center. The network has multiple same setups, and each is denoted as a distributively located fog location. The framework's performance is evaluated in terms of CPU & memory usage, delay in constructing enriched GUI, network delay, workload computation time, workload acceptance ratio, and energy. The simulation parameters and system specification where simulation is deployed are given in Table 3, and Table 4 respectively. The simulation parameters in the table as presented in terms of range, e.g., the user nodes range from 50 to 500 means the nodes are increased to measure the impact on the overall performance of the proposed system. The graphs presented here show the total number of nodes, including level-1, level2, level-3, and volunteer nodes.

Initialization delay – The proposed framework is developed on the top of OMNeT++ [52] which provides an enriched GUI environment to view running simulation. However, this GUI construction is a compute expensive task and creates an additional one-time delay computed for the proposed framework, as shown in Fig. 3. Furthermore, this delay increases with the number of network nodes.

Memory and CPU usage – Memory and CPU usage are directly proportional to the number of network nodes. The results are obtained by combing all network components, including IIoT devices, Fog nodes, Cloud servers, and networking components like routers and switches. Figure 4 shows the memory and CPU usage increases with the number of nodes. This is because of the memory utilization with the increased number of different simulation modules and objects.

Workload completion time – The workload completion is measured for two scenarios; horizontal and vertical placement. The horizontal placement means distributed fog locations, and vertical placement means the hierarchical placement of fog nodes on different levels and in cloud data centers, as shown in Fig. 5(a). The arrival rate varies, and workload execution time is measured by varying the number of nodes as shown in Fig. 5(b). The task size is kept constant (2000 MIPS) in both scenarios.

Network latency – The network delay and congestion depend on the number of users offloading workloads and the offloading frequency, as shown in Fig. 6. Figure 6(a) shows that the network latency depends on the fog placement provided that workload frequency is constant. If the fog nodes are in a hierarchical architecture, the delay is minimal; however, it increases if the fog nodes are in horizontal architecture (distributed). The latency of the workloads offloaded to the cloud is the maximum

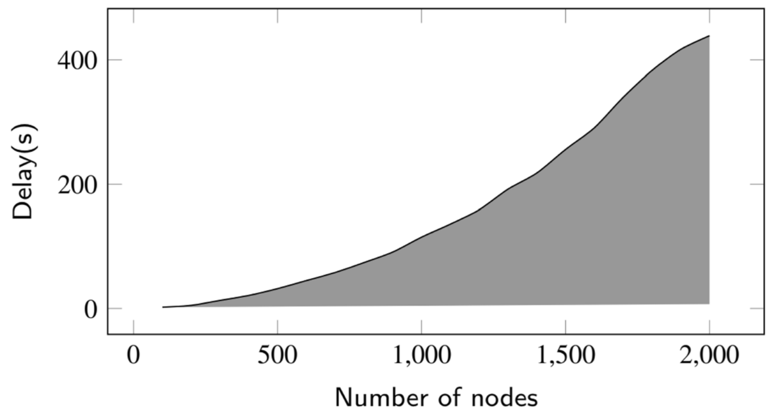

Fig. 3 One time component initialization and GUI construction delay

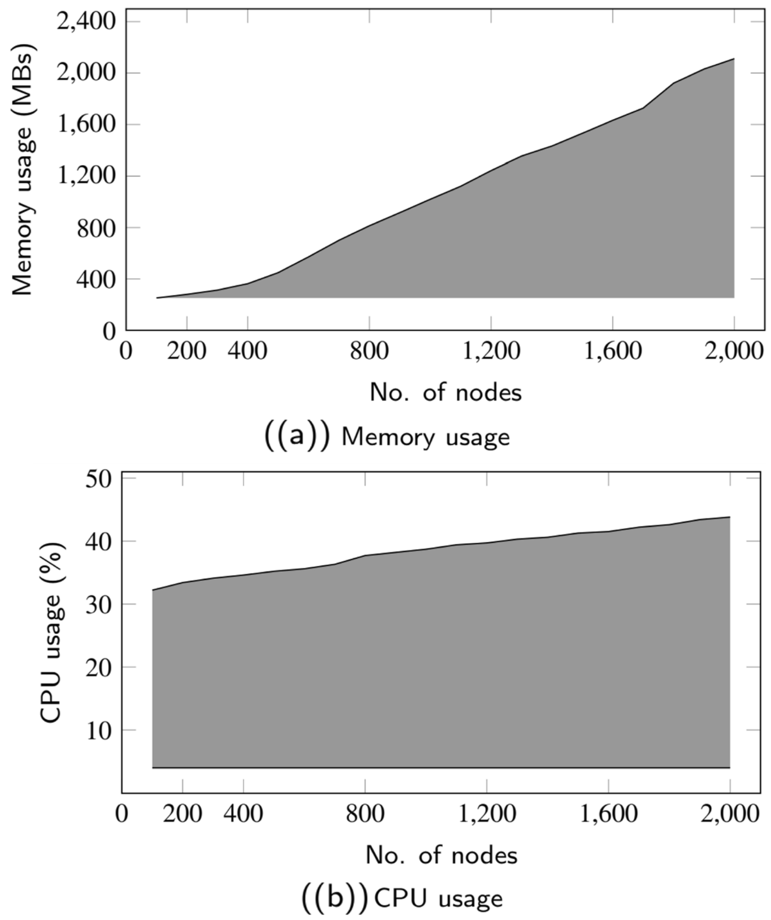

Fig. 4 Memory and CPU usage of the system with respect to number of nodes (IIoT + Fogs + Cloud + Network Components)

because an external network (internet) is used to offload workloads. However, the effect of workload frequency is shown in Fig. 6(b). The average network latency increases with the increase in workload arrival rate.

Residual energy – In the simulation, every IIoT functions in one mode: user mode when it offloads workloads or volunteer mode when executing the received workloads. Each device joins the network with a predefined

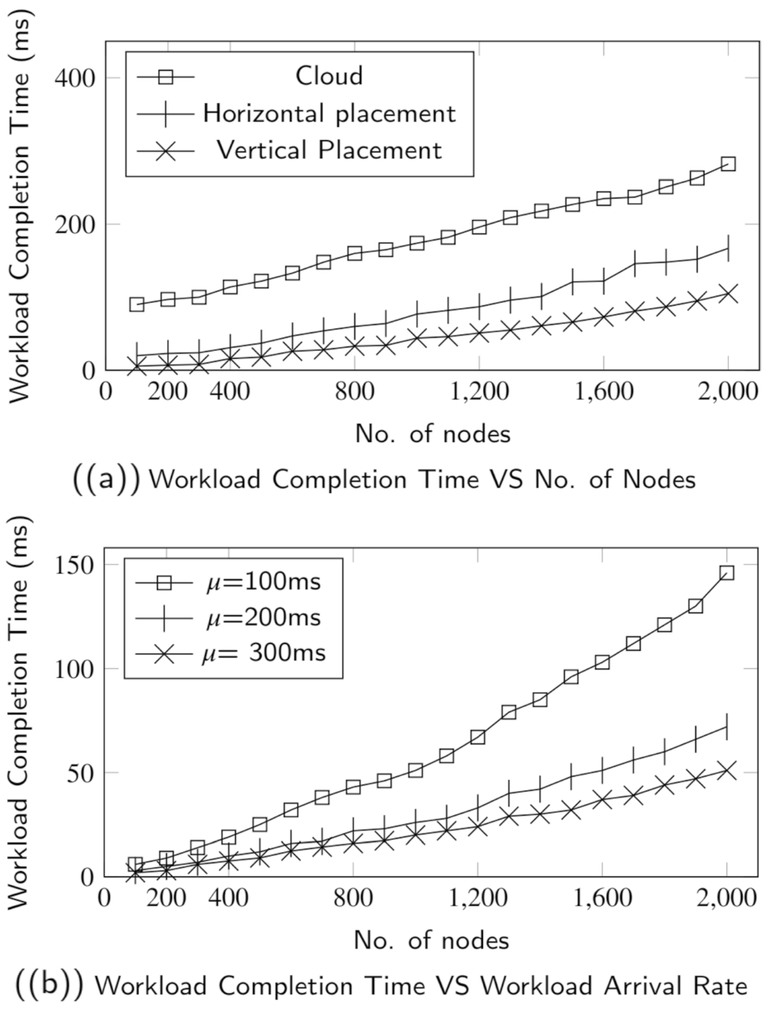

Fig. 5 Workload completion time with different arrival rate and network setup

energy value that reduces over time depending on the usage. For example, in Fig. 7, the residual energy of a volunteer node and two arbitrary random user nodes are given. When the energy of a node reduces to 0, it becomes inactive.

Workload acceptance ratio – The proposed framework is compared with FogNetSim++ [17] in workload acceptance ratio. Fig. 8 shows that the proposed framework outperformed and achieved a higher acceptance ratio when the number of nodes was increased.

Availability

The framework is developed using Omnet++ and Inet framework. It is an open-sourced project and is accessible in a GitHub repository[1].

[1] https://github.com/rtqayyum/IIoT-Fog/

Conclusion

To summarize, the inclusion of fog computing in IIoT shifts an IIoT system's performance to the next generation networks. Low latency is essential in many control applications of an IIoT system, and fog computing can enhance the system efficiency and risk of damage by providing a quick response to the respective machinery. The proposed framework provides a general framework to simulate IIoT fog networks and resolve the resource under-utilization problem by upgrading unused resources to compute and fog resources at a local level that will reduce delay and ensure the availability of resources. The localization module of the framework helps find the location of a mobile node in the network, and this information helps to make better decisions by resource allocation algorithm. The framework's efficiency is measured in terms of CPU, Memory, and GUI design delays. The performance of the resource sharing

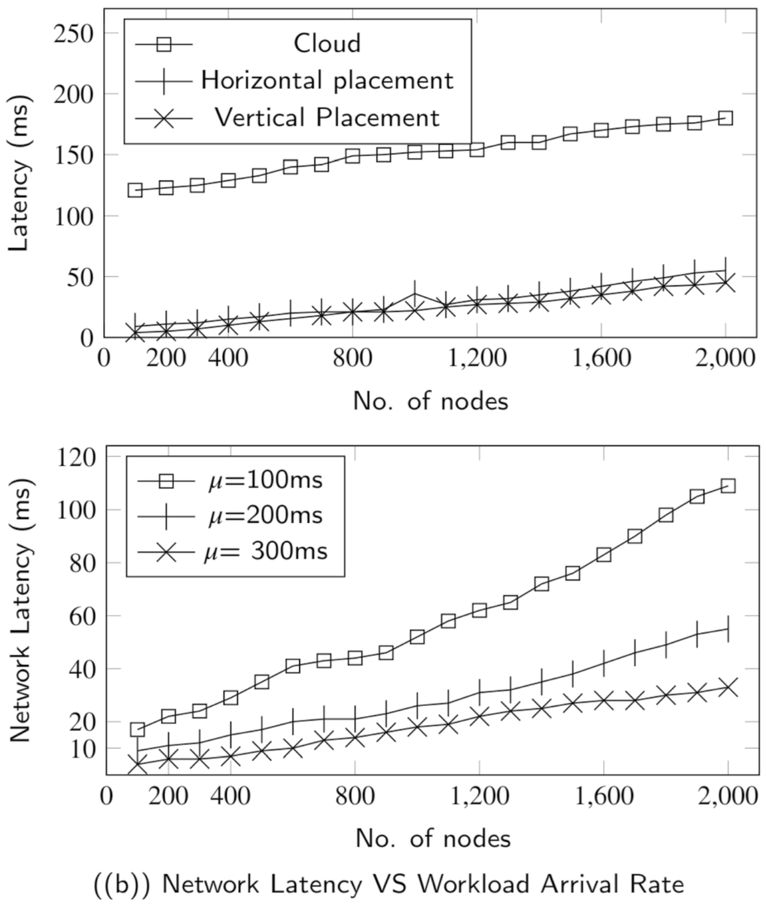

Fig. 6 Network latency time with different arrival rate and network setup

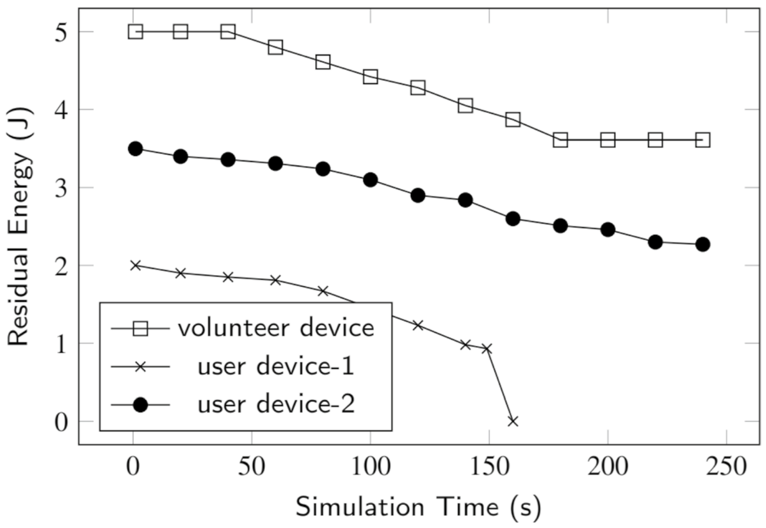

Fig. 7 Residual Energy of the IIoT devices in user mode where devices offload workloads, and volunteer mode when devices execute received workloads

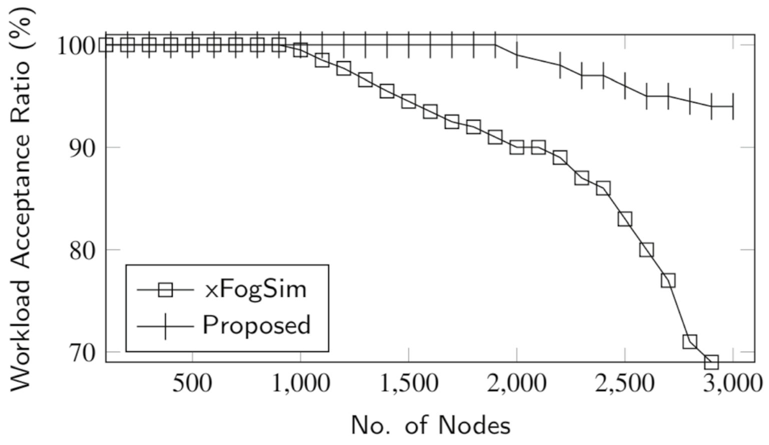

Fig. 8 The workload acceptance ratio compared to FogNetSim++ [17] when FognetSim++ starts dropping the incoming workloads when the number of user nodes is increased, and the proposed framework keep accepting the workloads

algorithm is compared with [17], and it is observed that cited framework starts dropping the incoming workloads after a specific time. In contrast, the proposed framework can manage resources better and receive workloads for a more extended time. Furthermore, in the future, additional rejection criteria can be added for volunteer nodes, such as the volunteer nodes being accepted or rejected based on specific security criteria, such as their level of trust. As a result, malicious nodes will not be able to disturb and attack the framework's efficiency and performance.

Acknowledgements
This work was supported by Zayed University Cluster Grant number R20140 and the United Arab Emirates University (UAEU) Program for Advanced Research (UPAR) under Grant number 31T122.

Authors' contributions
All authors contributed equally to the research. However, 'A.C' prepared the first draft, and 'B.D' contributed in design and refining the idea and paper. All authors read and approved the final manuscript.

Funding
This work was supported by the United Arab Emirates University (UAEU) Program for Advanced Research (UPAR) under Grant number 31T122 and the Zayed University Cluster Grant number R20140.

Availability of data and materials
Not applicable.

Code availability
Code is available in GitHub Rep under Section 6.

Declarations

Ethics approval and consent to participate
All authors contributed in the research.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]College of Information Technology, United Arab Emirates University, Al Ain 17551, Abu Dhabi, United Arab Emirates. [2]Department of Computing, SEECS, National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan. [3]College of Technological Innovation, Zayed University, Dubai 144534, Abu Dhabi, United Arab Emirates.

Received: 4 July 2022 Accepted: 23 September 2022
Published online: 28 October 2022

References

1. Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (MCC '12). Association for Computing Machinery, New York, pp 13–16. https://doi.org/10.1145/2342509.2342513
2. Xu H, Yu W, Griffith D, Golmie N (2018) A survey on industrial internet of things: A cyber-physical systems perspective. IEEE Access 6:78238–78259. https://doi.org/10.1109/ACCESS.2018.2884906
3. Li N, Xiao M, Rasmussen LK, Hu X, Leung VCM (2021) On resource allocation of cooperative multiple access strategy in energy-efficient industrial internet of things. IEEE Trans Ind Inf 17(2):1069–1078. https://doi.org/10.1109/TII.2020.2988643
4. Taleb T, Samdanis K, Mada B, Flinck H, Dutta S, Sabella D (2017) On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. IEEE Commun Surv Tutorials 19(3):1657–1681. https://doi.org/10.1109/COMST.2017.2705720
5. Sodhro AH, Pirbhulal S, de Albuquerque VHC (2019) Artificial intelligence-driven mechanism for edge computing-based industrial applications. IEEE Trans Ind Inform 15(7):4235–4243. https://doi.org/10.1109/TII.2019.2902878
6. Asim M, Wang Y, Wang K, Huang PQ (2020) A review on computational intelligence techniques in cloud and edge computing. IEEE Trans Emerg Top Comput Intell 4(6):742–763. https://doi.org/10.1109/TETCI.2020.3007905
7. Olmos JJV, Cugini F, Buining F, O'Mahony N, Truong T, Liss L, Oved T, Binshtock Z, Goldenberg D (2020) Big data processing and artificial intelligence at the network edge. In: 2020 22nd International Conference on Transparent Optical Networks (ICTON). pp 1–4. https://doi.org/10.1109/ICTON51198.2020.9203141

8. Zhou Z, Chen X, Li E, Zeng L, Luo K, Zhang J (2019) Edge intelligence: Paving the last mile of artificial intelligence with edge computing. Proc IEEE 107(8):1738–1762. https://doi.org/10.1109/JPROC.2019.2918951
9. Adhikari M, Srirama SN, Amgoth T (2020) Application offloading strategy for hierarchical fog environment through swarm optimization. IEEE Internet Things J 7(5):4317–4328. https://doi.org/10.1109/JIOT.2019.2958400
10. Jayawardene N, Fernando P (2019) Hybrid approach for enabling hierarchical fog networks in an iot deployment. In: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), vol 2. pp 517–521. https://doi.org/10.1109/COMPSAC.2019.10258
11. Tajalli SZ, Mardaneh M, Taherian-Fard E, Izadian A, Kavousi-Fard A, Dabbaghjamanesh M, Niknam T (2020) Dos-resilient distributed optimal scheduling in a fog supporting iiot-based smart microgrid. IEEE Trans Ind Appl 56(3):2968–2977. https://doi.org/10.1109/TIA.2020.2979677
12. Ramli MR, Daely PT, Lee JM, Kim DS (2019) Bio-inspired service provisioning scheme for fog-based industrial internet of things. In: 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). pp 1661–1664. https://doi.org/10.1109/ETFA.2019.8869402
13. Shi C, Ren Z, Yang K, Chen C, Zhang H, Xiao Y, Hou X (2018) Ultra-low latency cloud-fog computing for industrial internet of things. In: 2018 IEEE Wireless Communications and Networking Conference (WCNC). pp 1–6. https://doi.org/10.1109/WCNC.2018.8377192
14. Qiu T, Chi J, Zhou X, Ning Z, Atiquzzaman M, Wu DO (2020) Edge computing in industrial internet of things: Architecture, advances and challenges. IEEE Commun Surv Tutorials 22(4):2462–2488. https://doi.org/10.1109/COMST.2020.3009103
15. Lu Y, Richter P, Lohan ES (2018) Opportunities and challenges in the industrial internet of things based on 5g positioning. In: 2018 8th International Conference on Localization and GNSS (ICL-GNSS). pp 1–6. https://doi.org/10.1109/ICL-GNSS.2018.8440903
16. Bajic B, Rikalovic A, Suzic N, Piuri V (2021) Industry 4.0 implementation challenges and opportunities: A managerial perspective. IEEE Syst J 15(1):546–559. https://doi.org/10.1109/JSYST.2020.3023041
17. Qayyum T, Malik AW, Khan Khattak MA, Khalid O, Khan SU (2018) Fognetsim++: A toolkit for modeling and simulation of distributed fog environment. IEEE Access 6:63570–63583. https://doi.org/10.1109/ACCESS.2018.2877696
18. Kumar T, Harjula E, Ejaz M, Manzoor A, Porambage P, Ahmad I, Liyanage M, Braeken A, Ylianttila M (2020) Blockedge: Blockchain-edge framework for industrial iot networks. IEEE Access 8:154166–154185. https://doi.org/10.1109/ACCESS.2020.3017891
19. Chen S, Wang Z, Zhang H, Yang G, Wang K (2020) Fog-based optimized kronecker-supported compression design for industrial iot. IEEE Trans Sustain Comput 5(1):95–106. https://doi.org/10.1109/TSUSC.2019.2906729
20. Chekired DA, Khoukhi L, Mouftah HT (2018) Industrial iot data scheduling based on hierarchical fog computing: A key for enabling smart factory. IEEE Trans Ind Inf 14(10):4590–4602. https://doi.org/10.1109/TII.2018.2843802
21. Miao D, Liu L, Xu R, Panneerselvam J, Wu Y, Xu W (2018) An efficient indexing model for the fog layer of industrial internet of things. IEEE Trans Ind Inf 14(10):4487–4496. https://doi.org/10.1109/TII.2018.2799598
22. Mubeen S, Nikolaidis P, Didic A, Pei-Breivold H, Sandström K, Behnam M (2017) Delay mitigation in offloaded cloud controllers in industrial iot. IEEE Access 5:4418–4430. https://doi.org/10.1109/ACCESS.2017.2682499
23. Chen S, Zheng Y, Lu W, Varadarajan V, Wang K (2020) Energy-optimal dynamic computation offloading for industrial iot in fog computing. IEEE Trans Green Commun Netw 4(2):566–576. https://doi.org/10.1109/TGCN.2019.2960767
24. Yu Y, Xue L, Li Y, Du X, Guizani M, Yang B (2018) Assured data deletion with fine-grained access control for fog-based industrial applications. IEEE Trans Ind Inf 14(10):4538–4547. https://doi.org/10.1109/TII.2018.2841047
25. Mukherjee M, Kumar S, Mavromoustakis CX, Mastorakis G, Matam R, Kumar V, Zhang Q (2020) Latency-driven parallel task data offloading in fog computing networks for industrial applications. IEEE Trans Ind Inf 16(9):6050–6058. https://doi.org/10.1109/TII.2019.2957129
26. Fu J, Liu Y, Chao H, Bhargava BK, Zhang Z (2018) Secure data storage and searching for industrial iot by integrating fog computing and cloud computing. IEEE Trans Ind Inf 14(10):4519–4528. https://doi.org/10.1109/TII.2018.2793350
27. Aazam M, Zeadally S, Harras KA (2018) Deploying fog computing in industrial internet of things and industry 4.0. IEEE Trans Ind Inf 14(10):4674–4682. https://doi.org/10.1109/TII.2018.2855198
28. Lin C, Yang J (2018) Cost-efficient deployment of fog computing systems at logistics centers in industry 4.0. IEEE Trans Ind Inf 14(10):4603–4611. https://doi.org/10.1109/TII.2018.2827920
29. Chen Y, Sun E, Zhang Y (2019) A weight factors localization algorithm in fog-supported wireless sensor networks. In: 2019 IEEE 11th International Conference on Communication Software and Networks (ICCSN), pp 268–274. https://doi.org/10.1109/ICCSN.2019.8905271
30. Guidara A, Fersi G, Derbel F (2020) Lookup service for fog-based indoor localization platforms using chord protocol. In: 2020 International Wireless Communications and Mobile Computing (IWCMC). pp 345–350. https://doi.org/10.1109/IWCMC48107.2020.9148348
31. Ma S, Liu Q, Sheu PC (2018) Foglight: Visible light-enabled indoor localization system for low-power iot devices. IEEE Internet Things J 5(1):175–185. https://doi.org/10.1109/JIOT.2017.2776964
32. Femminella M, Reali G, Valocchi D (2016) A signaling protocol for service function localization. IEEE Commun Lett 20(7):1325–1328. https://doi.org/10.1109/LCOMM.2016.2564960
33. Bhargava K, McManus G, Ivanov S (2017) Fog-centric localization for ambient assisted living. In: 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC). pp 1424–1430. https://doi.org/10.1109/ICE.2017.8280050
34. Costa B, Pires PF, Delicato FC (2016) Modeling iot applications with sysml4iot. In: 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). pp 157–164. https://doi.org/10.1109/SEAA.2016.19
35. Han SN, Lee GM, Crespi N, Heo K, Van Luong N, Brut M, Gatellier P (2014) Dpwsim: A simulation toolkit for iot applications using devices profile for web services. In: 2014 IEEE World Forum on Internet of Things (WF-IoT). pp 544–547. https://doi.org/10.1109/WF-IoT.2014.6803226
36. Chen D, Irwin D, Shenoy P (2016) Smartsim: A device-accurate smart home simulator for energy analytics. In: 2016 IEEE International Conference on Smart Grid Communications (SmartGridComm). pp 686–692. https://doi.org/10.1109/SmartGridComm.2016.7778841
37. Abuhasel KA, Khan MA (2020) A secure industrial internet of things (iiot) framework for resource management in smart manufacturing. IEEE Access 8:117354–117364. https://doi.org/10.1109/ACCESS.2020.3004711
38. Dissect-cf-iot simulator. https://github.com/andrasmarkus/dissect-cf/tree/pricing. Accessed 01 Feb 2021
39. Sotiriadis S, Bessis N, Asimakopoulou E, Mustafee N (2014) Towards simulating the internet of things. In: 2014 28th International Conference on Advanced Information Networking and Applications Workshops. IEEE, Victoria. pp 444–448
40. Byrne J, Svorobej S, Gourinovitch A, Elango DM, Liston P, Byrne PJ, Lynn T (2017) Recap simulator: Simulation of cloud/edge/fog computing scenarios. In: 2017 Winter Simulation Conference (WSC). pp 4568–4569. https://doi.org/10.1109/WSC.2017.8248208
41. Pflanzner T, Kertesz A, Spinnewyn B, Latré S (2016) Mobiotsim: Towards a mobile iot device simulator. In: 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), pp 21–27. https://doi.org/10.1109/W-FiCloud.2016.21
42. Gupta H, Vahid Dastjerdi A, Ghosh SK, Buyya R (2017) ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. Softw Pract Experience 47(9):1275–1296
43. Fiandrino C, Capponi A, Cacciatore G, Kliazovich D, Sorger U, Bouvry P, Kantarci B, Granelli F, Giordano S (2017) Crowdsensim: a simulation platform for mobile crowdsensing in realistic urban environments. IEEE Access 5:3490–3503. https://doi.org/10.1109/ACCESS.2017.2671678
44. Gautam N (2012) Analysis of queues: methods and applications. CRC Press, Boca Raton
45. Ma Z, Xiao M, Xiao Y, Pang Z, Poor HV, Vucetic B (2019) High-reliability and low-latency wireless communication for internet of things: Challenges, fundamentals, and enabling technologies. IEEE Internet Things J 6(5):7946–7970. https://doi.org/10.1109/JIOT.2019.2907245
46. Chen J, Zhang L, Liang YC, Kang X, Zhang R (2019) Resource allocation for wireless-powered iot networks with short packet communication. IEEE Trans Wirel Commun 18(2):1447–1461. https://doi.org/10.1109/TWC.2019.2893335

47. Deng R, Lu R, Lai C, Luan TH, Liang H (2016) Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. IEEE Internet Things J 3(6):1171–1181. https://doi.org/10.1109/JIOT.2016.2565516
48. Liu CF, Samarakoon S, Bennis M, Poor HV (2018) Fronthaul-aware software-defined wireless networks: Resource allocation and user scheduling. IEEE Trans Wirel Commun 17(1):533–547. https://doi.org/10.1109/TWC.2017.2768358
49. Jin H, Zhu X, Zhao C (2019) Computation offloading optimization based on probabilistic sfc for mobile online gaming in heterogeneous network. IEEE Access 7:52168–52180. https://doi.org/10.1109/ACCESS.2019.2909971
50. Shen S, Yang B, Qian K, Jiang X (2015) An efficient localization algorithm in wireless sensor networks. In: 2015 Third International Symposium on Computing and Networking (CANDAR). pp 291–294. https://doi.org/10.1109/CANDAR.2015.99
51. Qayyum T, Trabelsi Z, Malik AW, Hayawi K (2021) Multi-level resource sharing framework using collaborative fog environment for smart cities. IEEE Access 9:21859–21869. https://doi.org/10.1109/ACCESS.2021.3054420
52. OpenSim. Omnet++ discrete event simulator. https://omnetpp.org/. Accessed 20 Nov 2021

Publisher's Note