
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
 Volume 25, Issues 2, pp. 1152–1166, DOI 10.12694/scpe.v25i2.2620

OPTIMUM BATCH SCHEDULING MODEL FOR QUALITY AWARE DELAY SENSITIVE DATA TRANSMISSION OVER FOG ENABLED IOT NETWORK

NARAYANA POTU*, CHANDRASHEKAR JATOTH†, AND PREMCHAND PARVATANENI‡

Abstract. The emerging fog networks in the internet of things (IoT) applications provide flexibility and agility for service providers. The combination of fog nodes and edge nodes enable them to deliver a given network service. However, the selection of suitable edge and fog nodes and their scheduling still remain a research challenge. Finding a globally optimal scheduling of oversized data transmission over IoT applications for industrial requirements is crucial. Optimal batch scheduling has been regarded as a viable way to achieve optimal scheduling in other contemporary network models. This manuscript has projected an Optimum Batch Scheduling Model (OBSM) for Quality aware Delay Sensitive Data Transmission over Fog Enabled IoT Networks. A novel clustering technique has been proposed in this manuscript to group the transmission nodes (fog or edge nodes) and data packets, which further pairs each group of data with one of the corresponding node group to achieve delay sensitivity and other quality factors such as energy efficiency. The data scheduling between data and node group is drawn from the previous contribution -"Quality aware Energy Efficient Scheduling Model (QEESM) for Fog Enabled IoT Network". The simulation results have shown that, in terms of average make span rate, average round trip time, and energy consumption, the batch scheduling model OBSM performs noticeably better than the contemporary scheduling models. The OBSM scheduling model's average make-span rate, roundtrip time, as well as energy consumption per make span are 23.3 7.03, 17.8 5.2, and 11.33 6.9 joules, respectively, which conclusively demonstrate that the OBSM model outperforms the existing models. A novel batch scheduling algorithm has been proposed using a unique unsupervised learning approach that suggested to cluster the transmission requests and transmission channels in to multiple clusters.

Key words: Internet of Things, Industrial IoT, fog computing, edge nodes, Optimum Batch Scheduling Model, QALS

1. Introduction. Introduction. There is an increase in communication and computation latency and response times as a result of IoT devices generating a large volume, variety, and velocity of data [1].For instance, IIoT gives connectivity among production lines and customers such that customers might guide the process of production directly as stated in [2], and connectivity of IIoT enables exchange of data among overall industrial devices such that entire production process could rapidly change by adapting to novel products [3].The significant and fundamental part in IIoT is industrial networks.

In order to fulfill the crucial pre-requisites of industrial applications, the networks of industry have to transmit data, with accuracy in the control of data transmission, provision of sufficient bandwidth aimed at video-streams and handling each packet in huge communications Several industrial wireless and wired networks have been projected for managing these high pre-requisites scenarios. Yet, there has been no one single network architecture, covering all the industrial pre-requisites. Heterogeneous networks comprising of wired network(s) and several wireless field networks are being considered as the future solutions for the industrial networks.

IoT has been a word for computing systems that refers to things created for rationally linking the animal world through the web, also known as the "things oriented vision," as mentioned in [4]. The architecture of the Internet of Things may be reduced to three mechanisms: devices, hardware, as well as middleware, depending on this previous contribution [5]. The first section of the system consists of the sensors and actuators needed for direct human and device communication in the physical world. Next, it is the responsibility of the gateways to enable and centralise communication among diverse objects. The third component, middleware, which is often found in the cloud, saves the data collected from other components as well as condenses the knowledge based on that data. The "REST (Representational state transfer)" web service is typically used by IoT middleware to adopt the SOA (service-oriented architecture).

*University College of Engineering, Osmania University, Hyderabad, India (Corresponding author, 1potunarayana@gmail.com)
†National Institute of Technology, Raipur, India (chandrashekar.jatoth@gmail.com).
‡University College of Engineering, Osmania University, Hyderabad, India.

The environment of IoT has been characterized by extreme heterogeneity where several devices of numerous architectures communicate among themselves along with other services of internet as in [5] [6]. Within this heterogeneous architecture, the fundamental role of gateways is twofold: they enable interaction of minimal computation performance systems to access internet providers as well as assure the secure communication of the wireless sensor nodes while the middleware interacts with diversified network protocols. However, they are faced with challenges: the gateways of IoT accept device data from several different communication networks; they need to retransmission the statistics to internet as well as convert the data received for Web service. The gateways would be further required to get packet data from hundreds of sensor systems as well as actuators. The tasks require a minimal latency in transmission as well as rapid data processing. In several instances, process bottle necks occur in REST call response time and not in transmission of packet. Thus, the usage of buffers has become essential for receiving and processing of data at the device end. In view of the extreme diversity of sensor data gathered by the devices, maximal transmission priority needs to be accorded for authentication of novel devices or actuators calls in the network. The gateway needs to prioritize the urgent calls over regular ones in the incoming traffic. Thus, the gateway will have to address the problems related to huge amount of data volumes, prioritization, processing of packets, synchronous messaging, and performance difference among network mechanisms.

There could be an effect when the performance of various algorithms and protocols changes. Some of the sensor devices, for example, use TCP transport, while others use UDP, and still others don't have a transport layer at all. It is also possible to mention connection and physical-layer strategies, such as the (IEEE 802.11) Wi-Fi network or the (IEEE 802.15.4) ZigBee network. Furthermore, when the gateway uses HTTPS protocol, sensors often use device manufacturer's proprietary protocols in the layer of application. In addition, this cross-pollination of technologies adds IoT gateway's computational complexity. The packet's priority is to recognize and manage packets with the highest priority. For instance, sensors that has been evaluating something of extreme importance, sensor components that have demanded action through actuators, or even tasks of high priority, such as sensor authentication in a network before forwarding data packets. In the case of Synchronous Messaging, synchronous links between middleware and sensor devices may be a bottleneck because the middleware's for the following packet to be processed, a response is necessary. Additionally, these difficulties call for the application of IoT gateway methods including traffic shaping, packet discarding, and queue management. In order to provide effective and equitable treatment again for requests processed by the gateway, this study proposes an IoT gateway including QoS qualities based on a network prioritization algorithm.

1.1. Motivation. The evolving IoT is thought of as the next internet generation. Actuators, cars, phones, cars, and sensors are examples of things that interact to provide a service. Cloud computing is a new computing model that allows users to access a shared pool of resources on demand. The cloud and IoT are currently attracting both industry and academia's attention. Sensor data is sent to a gateway, which sends it to the cloud. Typically, the cloud stores, processes, transmit, and stores user-generated data. When data transmission from sensor nodes to the cloud fails, data is re-transmitted until it succeeds.

1.2. Problem statement. In the fog IoT prototype, users serve as data requesters and the IoT serves as a source of data for the cloud. When there is an internet access, users may be able to get the necessary sensor data from the cloud. All of these applications for cloud-based IoT integration, such as smart cities and buildings depend on IoT to consistently deliver sensory data to the cloud in response to user requests [7]. Since non-rechargeable batteries are used in most sensors, data sensing performance, transmission, and processing depletes battery power over time. Several models exist for optimizing power consumption and improving IoT reliability. However, power consumption reduction models have negatively impacted network reliability.

1.3. Organizing of the Manuscript. In the further sections of this manuscript, section-2 refers to the related work summary from the literature review. Section-3 provides insights into the materials and methods, proposed model narrative, its algorithm flow, and other key metrics that signify the mode. Section-4 provides insights into experimental study and section-5 refers to the conclusion based on the efficiency aspects estimated from the model.

2. Related Research. According to Liu et al., [8], an ADEC (adaptive dynamic energy consumption) improvement strategy was proposed based on the JNCC model. The Improved Software-Defined WSN was developed by Y Duan et al., [9]. (Improved SD-WSN). Managing the network and ensuring adequate coverage are the primary concerns. To reduce energy consumption, C Zhu et al., [10] proposed a novel WSN-mobile cloud computing integration scheme that includes two parts: 1) a prioritized sleep scheduling algorithm for WSN and 2) a time-based selective data transmission algorithm for mobile cloud computing.

Each sensor's computing load and energy consumption are both reduced thanks to their use of fog-cloud hierarchical network architecture. Wang et al., [11] that a single, energy-constrained source can be used to power multiple energy-constrained relays in a wirelessly powered Internet of Things by selecting the best power beacons (PBs) (IoT), have proposed it. Over the Rayleigh fading channel, each scheme has closed-form expressions for power outage, secrecy outage, and energy efficiency IoT-cloud service prototype by S Kim et al., used Docker Swarm-based container orchestration to ensure the service's reliability and tested its uptime [12].

Each application's delay bound and target reliability were guaranteed using a QoS framework for arbitrary hybrid wired/wireless networks by S Zoppi et al., [13]. An alternative scheduling algorithm for wireless sensor networks is also proposed. Three methods are recommended by Y Peng and colleagues [14]: short, ongoing, and noise-related faults. Using an algorithm that estimates probability-guaranteed limits on packet reception ratio, W Sun et al., [15] propose a smart grid solution. According to J Yuan et al., [16], trusting IoT edge devices requires a multi-source feedback information fusion approach. For large-scale IoT edge computing, they present a low-overhead trust evaluation mechanism. Traditional trust schemes can be replaced by a new algorithm that incorporates feedback from multiple sources. In order to allow any TSCH node to transmit or receive frames at any time, H Park et al., [17] and his colleagues developed STATIC TSCH scheduling. Large-scale smart meter networks can be built with TSCH. Broadcast and unicast slots were defined separately to prevent network control message collisions.

In order to create an IoT platform with high exile and reliability, S C Wang et al., [18] used fog and cloud computing in conjunction (IFCIoT). IFCIoT can be used in a variety of other applications and disaster monitoring systems. All fault-free nodes in the IFC IoT platform can come to an agreement on a protocol with the smallest number of messages, while still allowing for malicious and dormant components. D Purkovic et al., [19] have proposed an energy-efficient communication protocol. In order to collect environmental data with the least amount of energy possible, this protocol is used. It's designed for sensors with short battery lives and energy harvesting. Teach-in and Data Telegram packets are used to collect the data.

JuanLuo et. al [20] proposed to increase the system efficiency of fog devices as well as decrease service latency, a multi-cloud into multi-fog architecture is used, along with the construction of two different service models using containers. The MILP was used by Al-Shammari et al., [21] to investigate the energy efficiency of a smart city service embedding framework. Using this framework, IoT systems will be able to meet the virtual node and link requirements of business processes while using less power. Fog computing for health monitoring was investigated by Ida Syafiza M. Isa et al., [22]. They used a MILP model to process and analyze patient electrocardiogram signals. Use of the most energy-efficient locations for processing and networking servers.

PegahGazori et. al. [23] concentrated on scheduling tasks in fog-based IoT systems with the goal of minimizing length of service latency and computation cost while adhering to resource and deadline limitations. We have conducted research using the reinforcement learning technique to solve this issue [23]. An IoT-based real-time HVAC control system [24] was designed and implemented using thermal comfort, demand response, and user feedback. In order to forecast the room's thermal parameters, they use artificial neural networks trained on historical data. With the help of MILP, the HVAC control problem is optimized for energy efficiency and user satisfaction. Decentralized micro grid energy exchange mechanisms have been proposed by Laszka et al., [25]. Consumers no longer have to worry about their privacy or the security of the system when they trade energy. In order to quickly and securely clear offers, the platform uses a hybrid MILP solver approach. IoT service selection was proposed by M E Khanouche et al., [26]. Energy consumption is minimized while service quality requirements are met by this multi-objective optimization problem. Pre-selecting services that meet the quality of service (QoS) requirements of end-users is a goal of each others. Pareto's concept of relative dominance relations is used to select the best service. It is the user's preferences that determine the relative dominance of a potential service (QoS). Near-optimal solutions for large-scale IoT environments with thousands

of distributed entities can be found using the EQSA algorithm (about 98 percent).

The Contemporary models "QEESM (Quality Aware Energy Efficient Scheduling Model) [27]", "CFCA (Container based Fog Computing Architecture) [28]", and "QALS (Quantum approach of load scheduling) [29]" are competent methods to perform data transmission with minimal droop ratio across IoT networks using fog computing. However, none of the aforesaid methods considering the context of the oversized delay sensitive data transmission. Considering the scope of a scheduling model to perform optimal transmission of oversized delay sensitive data transmission over fog enabled IoT networks, this manuscript endeavoured to portray an Optimum Batch Scheduling model for Quality aware Delay Sensitive Data Transmission over Fog Enabled IoT Networks.

3. Methods and Materials. The optimum batch scheduling model have been addressed to achieve delay sensitive and quality aware data scheduling In IoT networks with fog support. The issue of data frame loss is critical in IoT networks, which is due to overloaded transmission or by any crux of network quality issues that often admitted by IoT devices with weak resources. Hence, suboptimal data scheduling in IoT networks is a significant research issue to address. In order to this, earlier contribution has suggested a novel scheduling strategy for Fog enabled IoT networks that intended to achieve energy efficiency and other quality factors [27]. However, the overwhelmed data transmission sources of IoT networks, which has been connected to fog network based cloud services such as industrial IoT networks still causes considerable data frame drop ratio, which often critical to address in delay sensitive fog enabled IoT networks. To address this issue, this contribution has portrayed a batch scheduling strategy, which is a firm extension of the earlier contribution "Quality-aware Energy Efficient Scheduling Model (QEESM) for Fog Enabled IoT Networks". The suggested batch scheduling strategy is an unsupervised clustering technique that partitions the data frames to be transmitted over fog enabled IoT networks in to multiple clusters and then performs QEESM scheduling model on each cluster in the priority order of these clusters defined under delay sensitivity. The subsequent sections explore the methods and materials used in scheduling process portrayed [30], [31].

3.1. Data-frames and node clustering process. The process of clustering the nodes is initiated based on the estimation of distance amidst the idle times among the fitness function. Every cluster signifies the nodes based on the idle time overlap conditions. The fundamental function in the process of nodes clustering is to focus on the start and end time related to the nodes to observe overlap conditions. Accordingly, all the overlapping nodes are grouped into one. In line with the above the mentioned process, even the data-frames are grouped in order that the data-frames overlap based on the time-interval feasible for handling arrival and residual session span.

The process adapted in the case of the data-frames streaming to scheduler on basis of incremental arrival time, and the time interval nodes. The process refers to the conditions wherein the incremental start-time is considered, as well as the cluster - based method is triggered as a positive approach, to mitigate the risk of data-frame loss or inappropriate scheduling of the nodes. The objective is to overcome certain challenges of clustering which are imperative in the existing solutions.

In the proposed solution, the clustering process adapted is inspired by the traditional k-means clustering algorithm [32], [33]. However, the change considered in the model is about unrestricting the cluster counts. The emphasis is more about adhering to time series compatibility, wherein the newly arrived record relates to the recently formed cluster or shall be allowed to form a new cluster. The new cluster formation shall be based on the steps discussed in the sub-sections.

3.1.1. Node clustering. .When no more nodes satisfy the requirements, a cluster is considered complete. The process proceeds with the remaining nodes after a cluster's nodes are eliminated from the list. Until all nodes are clustered, this process is repeated. Idle nodes are effectively grouped using node clustering algorithms. Using a pre-sorted list, the methodical approach clusters nodes based on overlaps in idle time. Figure 3.1 illustrates how this algorithm forms clusters gradually to maximize network performance. Nodes are arranged to maximize idle times using this thorough clustering technique, which enhances network performance.

Algorithm to cluster the idle nodes

The objective of the algorithm is to cluster nodes based on their idle time intervals, optimizing network

performance in a fog-enabled IoT environment. The algorithm operates on a set of nodes with defined idle time intervals and employs a clustering mechanism inspired by k-means but adapted for time interval-based clustering.

1. **Initialization and Node Sorting:** Let $L = \{e_1, e_2, ..., e_n\}$ represent the list of nodes, sorted in ascending order based on the start times of their idle time intervals. This ensures $idle\_start(e_i) \leq idle\_start(e_{i+1})$ for all i.
2. **Cluster Centroid Initialization:** The first node in the list, e_1, is initialized as the centroid for the first cluster, $ctrd_j$, of the j^{th} cluster ncl_j.
3. **Cluster Formation:**
   - For each node e_i in nL, the algorithm checks if $idle\_start(e_i) < idle\_start(ctrd_j) + idle\_duration(ctrd_j)$.
   - If the condition is satisfied, e_i is added to ncl_j.
4. **Updating Clusters and Centroids:**
   - If new nodes are added to ncl_j, the algorithm sorts the nodes within the cluster in decreasing order of their idle time durations.
   - The node with the longest idle duration in ncl_j becomes the new centroid, $ctrd_j$.
   - If no new nodes are added, indicating that the cluster is complete, ncl_j is finalized.
5. **Iteration:**
   - Remove the nodes of ncl_j from nL and increment j for the next cluster formation.
   - Repeat the process until all nodes in nL are clustered.
6. **Termination:**
   - The process ends when nL is empty, indicating all nodes have been clustered.

This algorithm effectively groups nodes based on overlapping idle time intervals, optimizing the utilization of resources in a fog computing network. The mathematical approach ensures precision in clustering, leading to enhanced network performance and efficiency.

3.1.2. Data-frames clustering process for schedule.. The scheduling process for idle frame interval nodes is similar to that of data-frame clustering. Data frames are buffered and grouped by crucial transmission times by the scheduler shown in figure 3.2. In this process, data frames are arranged according to arrival times. The first data-frame in this ordered list, denoted as df, serves as the centroid of the first cluster, denoted by $dfcl_1$. Making this decision creates the foundation for cluster formation. The data-frame arrival times and centroid transmission times are compared using the clustering method. Data frames whose arrival times coincide with the centroid's transmission interval are received by the cluster $dfcl_1$. As a result, data frames with comparable transmission properties are grouped. During the process, the data-frame from $dfcl_1$ with the maximum transmission end time is used to reform the cluster centroid. To capture the most representative transmission characteristics of the cluster, dynamic centroid selection is essential. When the composition of a cluster remains constant between centroid formations, it is considered final. Following finalization, these clusters are eliminated from the list, and the remaining data frames are processed in the same manner. Until every data frame is grouped into clusters, this cycle is repeated. The interval start list is replaced by the data-frame list (DFL) and the arrival list of each data-frame, denoted as a(df), which are represented by the evolutionary clustering algorithm for this frame. The transmission time that is needed is $rtt(df)$. This cloning of the clustering process is depicted in the algorithmic representation below.

Algorithm: Data-Frames Clustering

1. **Initialization:**
   - Define $\mathrm{dfL} = \{\mathrm{df}_1, \mathrm{df}_2, ..., \mathrm{df}_n\}$ as the list of data-frames, sorted in ascending order based on their arrival times, $a(\mathrm{df}_i)$.
   - Define $\mathrm{df}_i^{\mathrm{rt}}$ as the required transmission time of data-frame f_i.
   - Initialize the cluster set, $\mathrm{dfcl}_j = \phi$, and a temporary set, $\mathrm{ts} = \phi$.
   - Set the initial cluster centroid, trdf_j, as the first data-frame in dfL.
2. **Clustering Process:**
   - For each data-frame df_i in dfL:

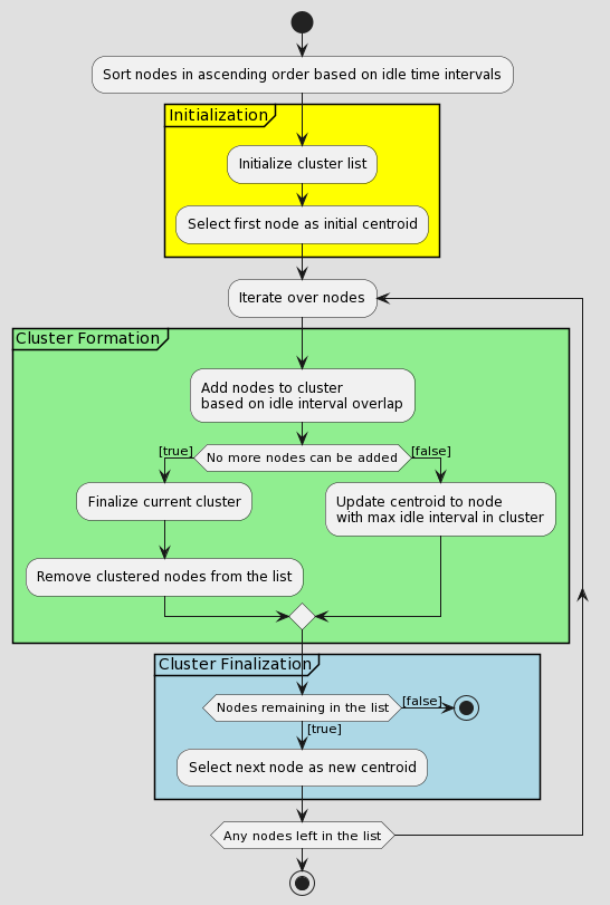

Fig. 3.1: FlowChart Representing of the OBSM Node Clustering

– If $a(\mathrm{df}_i) < a(\mathrm{ctrdf}_j) + \mathrm{df}_j^{\mathrm{rt}}$, add df_i to dfcl_j.

3. **Centroid Update and Cluster Finalization:**
   - If new data-frames are added to fcl_j, determine the data-frame with the maximum sum of arrival time and required transmission time, denoted as mrss.
   - Update ctrdf_j to the data-frame with the maximum mrss.
   - If no new data-frames are added, finalize dfcl_j and remove its elements from dfL.
4. **Iteration and New Cluster Formation:**
   - If dfL is not empty, increment the cluster index j and repeat the process with the remaining data-frames in dfL.
   - Set ctrdf_j as the first data-frame in the updated dfL, and initialize a new cluster dfcl_j.
5. **Termination:**
   - The process terminates when dfL is empty, indicating all data-frames have been clustered.

3.1.3. Ranking of the data-frame and cluster of nodes. . Each cluster of idle nodes is correlated with the data-frames using specific criteria, allowing the process to estimate three objectives. Distances are defined as the minimum start time of idle intervals for nodes within clusters and the minimum arrival time of data-frames. A comparison is also made between the data-frame transmission time and the mean idle time in clusters. Additionally, the process makes a comparison between the mean intervals of idle transmission times for nodes in the same cluster and the variance in data frame transmission times for nodes. Idle node clusters

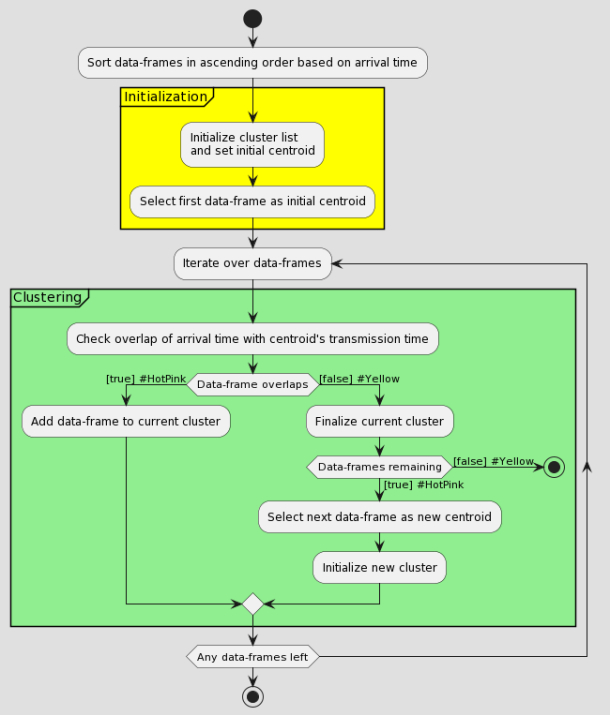

Fig. 3.2: Flow Chart Representing of the OBSM Data Frame Clustering

can be identified by classifying idle clusters by least arrival time data in descending order. Based on their sorted list position, these clusters are ranked according to arrival rate. The next step is to rank the data-frame clusters according to position and sort them according to mean transmission time. Based on average idle time intervals, clusters are ranked. Data-frame clusters are sorted in descending order using the variance in transmission times. Based on the variance of idle intervals, idle node clusters are ranked. The intervals between data transmission ranks, idle times, and transmission time deviation ranks are all estimated by this process. For every data-frame cluster, a notable cluster of nodes is selected in order to accomplish several objectives. The algorithmic ranking process for data-frame clusters is presented below. Please only use this process for Data-Frame Clusters.

The set of data-frame clusters, DFCL, is represented as $\text{dfcl}_1, \text{dfcl2}, ..., \text{dfcl}|\text{DFCL}|$. The algorithm involves cloning DFCL into a temporary set TC and creating ordered sets for data-frame clusters based on the least arrival time, average residual session span, and deviation in transmission times. The algorithm iterates through these sets, identifying data-frame clusters with the least arrival time, session span, and deviation. For each identified cluster, a rank is assigned based on its order in terms of arrival time, transmission time, and deviation. These ranks are used to move the data-frame cluster into respective ordered sets. The process continues until all clusters in TC are ranked.

Algorithm signifying discriminative ranks for data-frame clusters

1. **Initialization:**
   - Let $\text{DFCL} = \{\text{dfcl}_1, \text{dfcl}_2, ..., \text{dfcl}_n\}$ be the set of all data-frame clusters.
   - Define *a*oDFCL, *s*oDFCL, and *d*oDFCL as sets for arranging data-frame clusters based on least arrival time, average session span, and deviation in transmission time, respectively.
   - Initialize an index counter idx to 0.
2. **Ranking Based on Arrival Time:**
   - For each cluster dfcl_i in DFCL:
     - If dfcl_i is not in *a*oDFCL, find the cluster with the least arrival time and assign it to $\text{dfcl}_{i\text{a}}$.

   - Update aoDFCL by adding $\text{dfcl}_{i\text{a}}$ and assign it a rank based on its position in the ordered set.
3. **Ranking Based on Session Span:**
   - For each cluster dfcl_i in DFCL:
     - If dfcl_i is not in soDFCL, find the cluster with the least session span and assign it to $\text{dfcl}_{i\text{s}}$.
     - Update soDFCL by adding $\text{dfcl}_{i\text{s}}$ and assign it a rank based on its position in the ordered set.
4. **Ranking Based on Deviation:**
   - For each cluster dfcl_i in DFCL:
     - If dfcl_i is not in doDFCL, find the cluster with the least deviation and assign it to $\text{dfcl}_{i\text{d}}$.
     - Update doDFCL by adding $\text{dfcl}_{i\text{d}}$ and assign it a rank based on its position in the ordered set.
5. **Increment Index and Repeat:**
   - Increment the index idx.
   - Repeat the ranking process for each cluster in DFCL until all clusters have been ranked in aoDFCL, soDFCL, and doDFCL.
6. **Output:**
   - The output of the algorithm is the sets aoDFCL, soDFCL, and doDFCL with their respective ranks, representing the discriminative ranks of data-frame clusters based on arrival time, session span, and deviation in transmission time.

3.1.4. Collation of data-frame clusters and clusters of nodes. . The correlation of data-frame clusters and node clusters is based on the discriminative ranks allocated in the earlier process. For a specific data-frame cluster dfcl, a corresponding cluster of nodes ncl is determined based on several rank comparisons. These include the comparison of the rank of idle intervals in ncl with the mean transmission time rank of dfcl, the variance span rank in ncl with the variance span rank of dfcl, and the start time of idle intervals in ncl with the least arrival time rank of dfcl. Scheduling of data-frames is managed according to these paired clusters of nodes, ensuring an optimized correlation between node availability and data-frame transmission requirements.

3.2. Residual energy. The residual-energy pertaining to the n^{th} make-span at k^{th} transmission node is considerably the remains of energy observed from the earlier make-span ($(n-1)^{th}$ completion). Residual-energy re_n for the n^{th} make span is estimated as

$$re_n = (re_{(n-1)} + ce_{(n-1)} - coe_{(n-1)}) - co\tau$$

The notations $re_{(n-1)}$, re_n, $ce_{(n-1)}$, $co\tau$ are used for indicating the residual-energy for the $(n-1)^{th}$, n^{th} make-spans, wherein energy conserved for the period as the $(n-1)^{th}$ make-span, and the energy consumed at $(n-1)^{th}$ make-span and the energy consumed in the process of idle time amidst $(n-1)^{th}$ and n^{th} make-spans of the target node.

To assess the optimality for the projected $(n+1)^{th}$ the make-span, transmission node process constitutes energy efficiency, turnaround time ratio, and Task Arrival Time Interval ratio. The scheduling strategy proposed for handling the process is as follows. For every metric, turnaround time interval, the scope of process completion time interval, turnaround time interval, and task arrival interval are estimated for the make-span $(n+1)^{th}$, wherein the Max-State (mas), Min-State (mis), and Close-State (Cls) are observed in accordance to how the make-span close.

In furtherance, a two-dimensional matrix for every metric is depicted wherein the values for the status measures are handled in a matrix format depicted below Table 3.1.

Followed by, for every metric, the task arrival time interval, turn around time interval, and process completion time interval are handled effectively wherein the moving averages for every status measure are assessed based on the close-state, max-state, min-state, and initial-state. For every column c related to the two-dimensional matrix $M(\text{ins})$, $M(\text{mas})$, $M(\text{mis})$, or $M(\text{cls})$ referring to the status measures ins, mas, mis, and cls of every metric as follows: turnaround time interval (tti), task arrival time interval (tati), process com-

Table 3.1: The values for the status measures are handled in a matrix

Make span-ID	Initial-State	Max-State	Min-State	Close-State
1	$[ins_1^{mi}]^{-1}$	$[mas_1^{mi}]^{-1}$	$[mis_1^{mi}]^{-1}$	$[cls_1^{mi}]^{-1}$
2	$[ins_2^{mi}]^{-1}$	$[mas_2^{mi}]^{-1}$	$[mis_2^{mi}]^{-1}$	$[cls_2^{mi}]^{-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	$[ins_i^{mi}]^{-1}$	$[mas_i^{mi}]^{-1}$	$[mis_i^{mi}]^{-1}$	$[cls_i^{mi}]^{-1}$
$i+1$	$[ins_{(i+1)}^{mi}]^{-1}$	$[mas_{(i+1)}^{mi}]^{-1}$	$[mis_{(i+1)}^{mi}]^{-1}$	$[cls_{(i+1)}^{mi}]^{-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	$[ins_n^{mi}]^{-1}$	$[mas_n^{mi}]^{-1}$	$[mis_n^{mi}]^{-1}$	$[cls_n^{mi}]^{-1}$

pletion time interval (pcti), The necessary moving averages for each of the column are observed, for column c.

$$ma_i(c) = \frac{1}{\text{mac}} \left(\sum_{j=i}^{i+\text{mac}-1} c[j] \right), \quad \text{for } i = 1 \text{ to } (|c| - \text{mac}) \tag{3.1}$$

Eq 3.1 moving average referring to values of column c of the two-dimension matrix M of the metric *tti*, *tati*, or *pcti* Post the process, for every set of moving average, the Heikin-ashi equivalent is developed based on the open, low, high, as well as end values, which enables in deriving the candle patterns for representing the expected make-suitability span's for the associated metric.

Depending on the indicated nodes, the emphasis is optimality for all the metrics like the Heikin-Ashi candle patterns, which are estimated. For each node n, for every metric rt, rr, cl, the moving average with each $maM(rt), maM(rr), maM(cl)$, The following is how Heiken-ashi patterns are created. $ha - maM_i(o) = maM_i(o)$ // For the moving averages for the initial make-span-related indicators, Else, the Heiken-ashi moving averages are calculated as follows: Eq 3.2 to Eq 3.5

$$ha - maM_i(o) = \frac{ha - maM_{(i-1)}(o) + ha - maM_{(i-1)}(e)}{2} \tag{3.2}$$

$$ha - maM(e) = \frac{maM(o) + maM(h) + maM(l) + maM(E)}{4} \tag{3.3}$$

$$ha - maM(l) = \min(ha - maM(o), maM(l), ha - maM(e)) \tag{3.4}$$

$$ha - maM(h) = \max(ha - maM(o), maM(h), ha - maM(e)) \tag{3.5}$$

Subsequently, it determines each node's grade coefficient in the manner shown below.

For every node n, for every metric a request receiving time (rr), round-trip time (rt), as well as computational-process-time (cl) are used for observing the average of the high-value (h), status-measures (open-value (o), end-value (e), as well as low-value (l)), as well as a variance, the root-mean-square length using the equation as: Eq 3.6 to Eq 3.14.

$$\langle rt_n \rangle = \frac{rt_n^o + rt_n^h + rt_n^l + rt_n^e}{4} \tag{3.6}$$

$$rt_n^\rho = \frac{\sqrt{\langle rt_n \rangle - rt_n^o}^2 + \sqrt{\langle rt_n \rangle - rt_n^h}^2 + \sqrt{\langle rt_n \rangle - rt_n^l}^2 + \sqrt{\langle rt_n \rangle - rt_n^e}^2}{4} \tag{3.7}$$

Table 3.2: Optimal Node IDs

Node ID	Quality Coefficient	Projected Energy Consumption	Residual Energy

Table 4.1: The Table Record ID, Metric Values

RECORD-ID	METRIC VALUES			
Node-ID Makespan-ID Metric-ID	High-value	Open-Value	End-value	Low-value

$$rt_n^{mc} = \langle rt_n \rangle + rt_n^{\rho} \tag{3.8}$$

$$\langle rr_n \rangle = \frac{rr_n^o + rr_n^h + rr_n^l + rr_n^e}{4} \tag{3.9}$$

$$rr_n^{\rho} = \frac{\sqrt{(\langle rr_n \rangle - rr_n^o)^2} + \sqrt{(\langle rr_n \rangle - rr_n^h)^2} + \sqrt{(\langle rr_n \rangle - rr_n^l)^2} + \sqrt{(\langle rr_n \rangle - rr_n^e)^2}}{4} \tag{3.10}$$

$$rr_n^{mc} = \langle rr_n \rangle + rr_n^{\rho} \tag{3.11}$$

$$\langle cl_n \rangle = \frac{cl_n^o + cl_n^h + cl_n^l + cl_n^e}{4} \tag{3.12}$$

$$cl_n^{\rho} = \frac{\sqrt{(\langle cl_n \rangle - cl_n^o)^2} + \sqrt{(\langle cl_n \rangle - cl_n^h)^2} + \sqrt{(\langle cl_n \rangle - cl_n^l)^2} + \sqrt{(\langle cl_n \rangle - cl_n^e)^2}}{4} \tag{3.13}$$

$$cl_n^{mc} = \langle cl_n \rangle + cl_n^{\rho} \tag{3.14}$$

In furtherance, sorts of the optimal nodes oN as an ascending order for the projected residual energy are estimated, as it can refer to the node scope in terms of completing the transaction, and sorting in open-value, feasible end-value. Also, in terms of the descending order for the respective quality coefficient, the nodes sort shall appear in the following dimension Table 3.2.

Thus, the scheduling strategy chosen as one of the significant issues in the listed nodes constitute to schedule based on the contextual factors considered, as necessary.

4. Experimental Study. The suggested Optimum Batch Scheduling Model (OBSM) for quality aware delay sensitive data transmission across fog enabled IoT networks has undergone an experimental evaluation to scale performance.

The experimental study performed in passive model, which executes the proposed and contemporary models QEESM, on known data (labelled data). The dataset has generated and used, which explored in contemporary model QEESM. The dataset format has been portrayed following Table 4.1.

This aforementioned dataset was created using a simulation based on fog-sim. The following table lists the simulation parameters that were employed (Table 4.2).

A total of 3600 internet protocol equipped sensors have been taken into account when creating the dataset. Each of them produces data at an average speed of 300 kbps. Through the scheduling gateway, these sensors are connected to the edge nodes. Additionally, these edge nodes are connected to a network of 50 fog nodes in a fog

Table 4.2: The Simulation Parameters

Parameter	**Specification**
Low power parameter	eDRX, Power saving mode
Latency parameter	greater than 10 seconds
Data Rate specification	25 kbps in download and 64 kbps in UL
Link budget parameter	above 164 dB (20dB GPRS)
Modulation scheme Uplink parameter Downlink parameter	 $\pi/4$-QPSK, $\pi/2$-BPSK, QPSK QPSK
Multiple access Downlink specification Uplink specification	 OFDMA SC-FDMA
Duplex Mode specification	FDD Half Duplex Type B
Frequency Range parameters	1,2,3,5,8,11,12,13,17,18,19,20,25,26,28,66,70 MHz
Supporting parameters	Uplink power control, HARQ

Table 4.3: Average Make-span Rate for Both Current Models and OBSM

Node Count	10	15	20	25	30	35	40	45	50
OBSM	13	15	15	20	28	30	29	29	31
QEESM	17	19	19	23	31	32	32	33	34
CFCA	19	22	22	26	35	36	35	36	37
QALS	35	37	41	44	46	49	51	54	55

computing system. The metric data collected from the most recent 60 make-spans in order are displayed one per node of the edge and fog networks. The dataset that was produced has 216,000 records in total. It's make-span rate on average, round-trip duration beside a varied set of fog devices, as well as variable transmission load are the metrics taken into account for performance analysis. The energy consumption ratio versus fluctuating load is another key metric that was evaluated.

By making comparisons the metric reported the results for OBSM with said correlating measure obtained values again from recent methods QEESM [27], CFCA [28], as well as (QALS) [29], it can be seen that the effectiveness of the OBSM has increased.

The presence of the nodes is indicated by the metric 'make-span rate,' which is proportional to each other. The lower make-span rate is predicted by Table 4.3 and Figure 4.1 to be inversely proportional to the total fog nodes. The typical make-span rate that the OBSM scheduling model has detected is 23.3 ± 7.03. When compared to the mean make-span rates 26.7 ± 6.62, 29.8 ± 6.95, and 45.8 ± 6.7 of the QEESM, CFCA, and QALS models, the mean make-span rate of the OBSM is noticeably low and has a little divergence.

For OBSM, QEESM, CFCA, QALS, other reliability of this research round frame has indeed been taken into account as well as compared (see Table 4.4, Figure 4.2), The statistics shown in Figure 4.2 clearly show that the model OBSM outperforms the current models QEESM, CFCA, QALS with shorter round-trip time. The mean of the average round-trip times recorded from OBSM, QEESM, CFCA, QALS is 17.8 ± 5.2, 33.96 ± 6.0, 40.36 ± 10.3, as well as 44.56 ± 14.3.

The anticipated round-trip time for a variable load is shown in Table 4.5 and Figure 4.3. When scaled against the modern models QEESM, CFCA, QALS, the model OBSM has shown a superior performance. The average difference here between round-trip time 14.9 ± 7.7 of the OBSM, and the round-trip time 32.3 ± 7.01 of the QEESM, 39.9 ± 12.9 of CFCA, and 44.9 ± 17.9 of QALS model.

Table 4.6 and Figure 4.4 show statistics for the essential objective energy consumption. The mean energy consumption in joules for every make-span for the proposed as well as current models, QEESM, OBSM, CFCA,

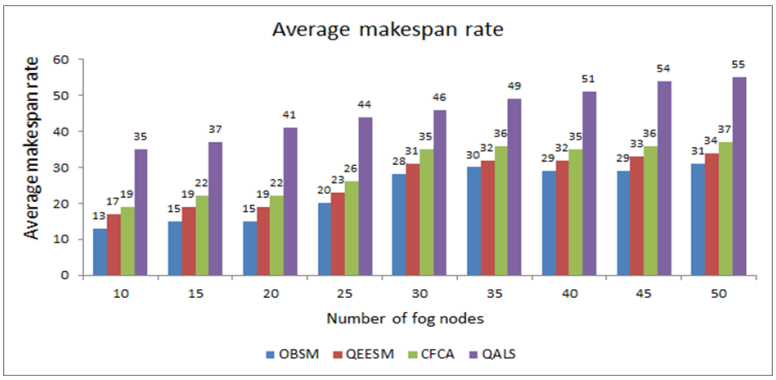

Fig. 4.1: The common make-span rates for the OBSM, QEESM, CFCA, and QALS

Table 4.4: Average Round-Trip Time for Fog Nodes with a Changeable Number of Nodes vs. a Fixed Load (Specify Constant Load Value)

Node Count	10	15	20	25	30	35	40	45	50
OBSM	21	25	22	21	19	18	15	11	8
QEESM	42	40	39	37	33	31	29	28	23
CFCA	44	42	43	40	36	35	33	33	27
QALS	49	46	47	44	40	38	36	36	30

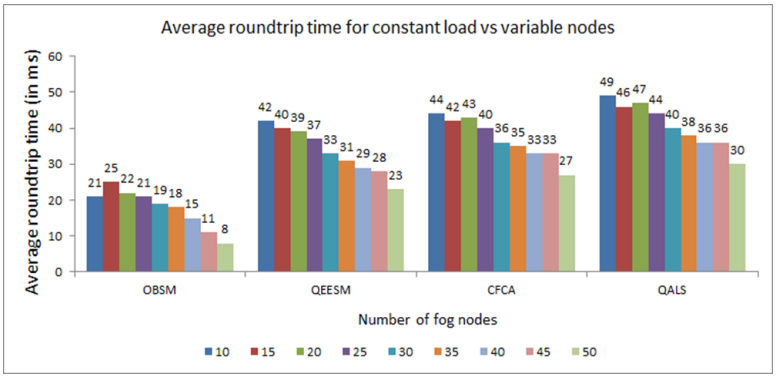

Fig. 4.2: The average round-trip time of OBSM, QEESM, CFCA, and QALS versus variable number of fog nodes

Table 4.5: Average Round-Trip Time for a Load Fixed vs a Different Number of the Fog Nodes

Load in KBPS	100	150	200	250	300	350	400	450	500
OBSM	5	8	9	14	14	14	16	22	32
QEESM	22	25	28	30	29	34	38	41	44
CFCA	28	30	32	34	35	40	45	46	51
QALS	33	35	37	38	40	44	48	52	55

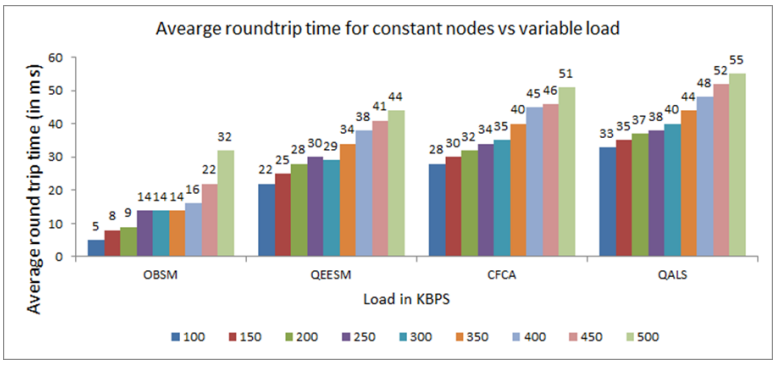

Fig. 4.3: Models OBSM, QEESM, CFCA, & QALS round-trip time statistics versus variable nodes

Table 4.6: Energy consumption for each make-span with a variable load

Load in KBPS	100	150	200	250	300	350	400	450	500
OBSM	1	2	6	9	12	15	17	18	22
QEESM	3	9	12	20	21	25	31	39	44
CFCA	8	14	17	24	26	29	35	44	49
QALS	12	18	21	29	31	35	39	48	53

as well as QALS, has been evaluated by comparing. The results show that the proposed model, including an average of 11.33 ± 6.9 joules for every make-span, outperforms a current model, QEESM with a consumption of 22.7 ± 12.8 joules per make-span, CFCA with a consumption of 27.4 ± 12.8, and QALS. In comparison to the energy used by the modern models QEESM, CFCA, and QALS, the OBSM uses less energy on average.

4.1. Critical Analysis. An experimental study on the Optimal Batch Scheduling Model (OBSM) assesses how well it manages delay-sensitive, quality-aware data transmission over fog-enabled Internet of Things networks. In this study, a specially generated dataset and an elaborate simulation setup are used to compare the OBSM with contemporary models such as QEESM, CFCA, and QALS. A widely used IoT simulation tool called fog-sim was used to create the dataset for this study. High, open, end, and low metrics for a thorough analysis are displayed in Table 4.1.

The simulation parameters in Table 4.2 are essential to the experimental setup used in this study. Examples include data rate specifications, latency parameters longer than 10 seconds, and low power parameters such as eDRX and Power Saving Mode. The modulation scheme, downlink and uplink parameters, and auxiliary features like HARQ and Uplink Power Control are all included in the simulation. The experiment's applicability to actual Internet of Things network scenarios is enhanced by meticulous parameter selection.

3600 internet protocol-equipped sensors are considered in the experiment as part of a network setup. A fog computing system is created by connecting these sensors through edge nodes to a network of fifty fog nodes. A critical analysis is conducted on the make-span rate, round-trip duration, energy consumption ratio, and over variable fog device and transmission load counts. An extensive setup like this offers a scalable and realistic OBSM testbed.

A thorough comparative analysis of OBSM is given in Tables 4.3 through 4.6 and Figures 4.1 through 4.4. A crucial network node indicator, the make-span rate, is displayed in Table 4.3 and Figure 4.1. The OBSM exhibits a lower make-span rate across node counts, suggesting improved task handling. Figures 4.2 and 4.3 and Tables 4.4 and 4.5's round-trip time comparisons, which show that OBSM reduces transmission delays more quickly, corroborate this.

The analysis of energy consumption (Table 4.6, Figure 4.4) is an important aspect of the study, showing that OBSM is more energy-efficient than the other models. This finding highlights the applicability of the

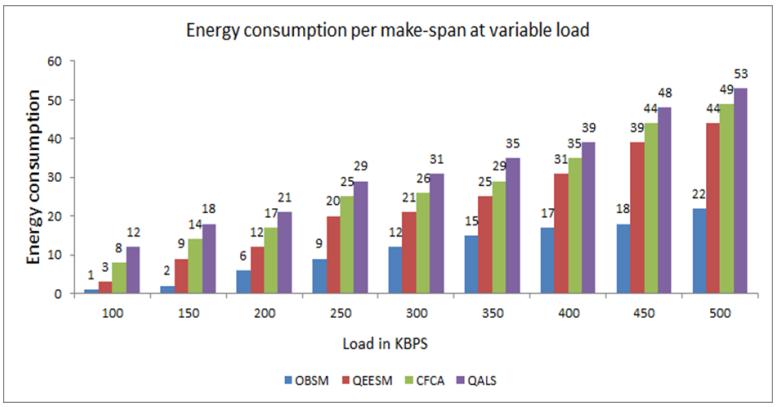

Fig. 4.4: The consumption of energy used for every make-span in variable load that was observed using the OBSM, QEESM, CFCA, & QALS methods

model for energy-efficient and sustainable Internet of things applications.

The reviewer recommended enhancing visual clarity, particularly in the flow chart (Figure 3.1), despite the study's strengths. To gain a deeper understanding of the system's functioning, a more comprehensive discussion of OBSM's algorithms and their impact on model performance is warranted.

5. Conclusion. The important problem of scheduling large-scale, time-sensitive data transmission over fog-enabled IoT networks in industry is addressed in this manuscript. Current scheduling models perform well for small to moderately delayed data transmissions, but not well for highly delayed data. This difference is closed by the fog-enabled IoT network-specific Optimal Batch Scheduling Model (OBSM). When it comes to performance, OBSM outperforms QEESM, CFCA, and QALS. OBSM's make-span rate is 23.3±7.03, which shows that it is superior due to its lower average and minimal deviation. With 17.8±5.2, OBSM beats QEESM (33.96±6.0), CFCA (40.36±10.3), and QALS (44.56±14.3) in terms of average round-trip times. And in terms of energy efficiency, OBSM shines. Its energy consumption per make-span is lower (14.9±7.7 joules) than that of QALS (31.8±12.8), CFCA (27.4±12.8), and QEESM (22.7±12.8). These findings show how well the OBSM model performs in fog computing environments when handling large-scale, delay-sensitive data transmission. In terms of make-span rate, round-trip time, and energy consumption ratio, OBSM performs better than current models. Further studies will look more closely at IoT network task and workflow scheduling enabled by fog computing. Enhancing data transmission strategies in these crucial technological ecosystems requires this.

REFERENCES

[1] C. Paniagua and J. Delsing, *Industrial frameworks for internet of things: A survey*, IEEE Systems Journal, 15 (2020), pp. 1149–1159.

[2] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, *Edge computing in industrial internet of things: Architecture, advances and challenges*, IEEE Communications Surveys & Tutorials, 22 (2020), pp. 2462–2488.

[3] W. Wu, Z. Zhao, L. Shen, X. T. R. Kong, D. Guo, R. Y. Zhong, and G. Q. Huang, *Just Trolley: Implementation of industrial IoT and digital twin-enabled spatial-temporal traceability and visibility for finished goods logistics*, Advanced Engineering Informatics, 52 (2022), pp. 101571.

[4] S. Alam, S. T. Siddiqui, A. Ahmad, R. Ahmad, and M. Shuaib, *Internet of things (IoT) enabling technologies, requirements, and security challenges*, in Advances in Data and Information Sciences: Proceedings of ICDIS 2019, Springer Singapore, 2020, pp. 119–126.

[5] F. Amin, R. Abbasi, A. Mateen, M. A. Abid, and S. Khan, *A step toward next-generation advancements in the internet of things technologies*, Sensors, 22 (2022), pp. 8072.

[6] M. Bansal, M. Nanda, and M. N. Husain, *Security and privacy aspects for Internet of Things (IoT)*, in 2021 6th International Conference on Inventive Computation Technologies (ICICT), IEEE, 2021, pp. 199–204.

[7] B. C. Csáji, Z. Kemény, G. Pedone, A. Kuti, and J. Váncza, *Wireless multi-sensor networks for smart cities: A prototype system with statistical data analysis*, IEEE Sensors J., 17 (2017), pp. 7667–7676.

[8] X. Liu, N. Xiong, W. Li, and Y. Xie, *An optimization scheme of adaptive dynamic energy consumption based on joint network-channel coding in wireless sensor networks*, IEEE Sensors J., 15 (2015), pp. 5158–5168.
[9] Y. Duan, W. Li, X. Fu, Y. Luo, and L. Yang, *A methodology for reliability of WSN based on software defined network in adaptive industrial environment*, IEEE/CAA J. AutomaticaSinica, 5 (2018), pp. 74–82.
[10] C. Zhu, Z. Sheng, V. C. M. Leung, L. Shu, and L. T. Yang, *Toward offering more useful data reliably to mobile cloud from wireless sensor network*, IEEE Trans. Emerg. Topics Comput., 3 (2015), pp. 84–94.
[11] Y. Wang, W. Yang, X. Shang, J. Hu, Y. Huang, and Y. Cai, *Energy-efficient secure transmission for wireless powered Internet of things with multiple power beacons*, IEEE Access, 6 (2018), pp. 75086–75098.
[12] S. Kim, C. Kim, and J. Kim, *Reliable smart energy IoT-cloud service operation with container orchestration*, in Proc. 19th Asia-Pacific Netw. Oper. Manage. Symp. (APNOMS), Sep. 2017, pp. 378–381.
[13] S. Zoppi, A.V. Bemten, H. M. Gürsu, M.Vilgelm, J. Guck, and W. Kellerer, *Achieving hybrid wired/wireless industrial networks with WDetServ: Reliability-based scheduling for delay guarantees*, IEEE Trans. Ind. Inform., 14 (2018), no. 5, pp. 2307–2319.
[14] Y. Peng, W. Qiao, L. Qu, and J. Wang, *Sensor fault detection and isolation for a wireless sensor network-based remote wind turbine condition monitoring system*, IEEE Trans. Ind. Appl., 54 (2018), no. 2, pp. 1072–1079.
[15] W. Sun, W. Lu, Q. Li, L. Chen, D. Mu, and X. Yuan, *WNN-LQE: Wavelet neural-network-based link quality estimation for smart grid WSNs*, IEEE Access, 5 (2017), pp. 12788–12797.
[16] J. Yuan and X. Li, *A reliable and lightweight trust computing mechanism for IoT edge devices based on multi-source feedback information fusion*, IEEE Access, 6 (2018), pp. 23626–23638.
[17] H. Park, H. Kim, K. T. Kim, S.-T. Kim, and P. Mah, *Frame-type-aware static time slotted channel hopping scheduling scheme for large-scale smart metering networks*, IEEE Access, 7 (2018), pp. 2200–2209.
[18] S.-C. Wang, S.-C. Tseng, K.-Q. Yan, and Y.-T. Tsai, *Reaching agreement in an integrated fog cloud IoT*, IEEE Access, 6 (2018), pp. 64515–64524.
[19] D. Purkovic, M. Honsch, and T. R. M. K. Meyer, *An energy efficient communication protocol for low power, energy harvesting sensor modules*, IEEE Sensors J., 19 (2019), no. 2, pp. 701–714.
[20] J. Luo et al., *Container-based fog computing architecture and energy-balancing scheduling algorithm for energy IoT*, Future Generation Computer Systems, 97 (2019), pp. 50–60.
[21] H. Q. Al-Shammari, A. Lawey, T. El-Gorashi, and J. M. Elmirghani, *Energy efficient service embedding in IoT networks*, in Proc. 27th Wireless Opt. Commun. Conf. (WOCC), Apr./May 2018, pp. 1–5.
[22] I. S. M. Isa, M. O. I. Musa, T. E. H. El-Gorashi, A. Q. Lawey, and J. M. H. Elmirghani, *Energy efficiency of fog computing health monitoring applications*, in Proc. 20th Int. Conf. Transparent Opt. Netw. (ICTON), Jul. 2018, pp. 1–5.
[23] P. Gazori, D. Rahbari, M. Nickray, *Saving time and cost on the scheduling of fog-based IoT applications using deep reinforcement learning approach*, Future Generation Computer Systems, 110 (2020), pp. 1098–1115.
[24] A. Rajith, S. Soki, and M. Hiroshi, *Real-time optimized HVAC control system on top of an IoT framework*, in Proc. Third Int. Conf. Fog Mobile Edge Comput. (FMEC), Apr. 2018, pp. 181–186.
[25] A. Laszka, S. Eisele, A. Dubey, G. Karsai, and K. Kvaternik, *TRANSAX: A blockchain-based decentralized forward-trading energy exchanged for transactive microgrids*, in Proc. IEEE 24th Int. Conf. Parallel Distrib. Syst. (ICPADS), Dec. 2018, pp. 918–927.
[26] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar, and A. Yachir, *Energy-centered and QoS-aware services selection for Internet of Things*, IEEE Trans. Autom. Sci. Eng., 13 (2016), no. 3, pp. 1256–1269.
[27] K. Ramana, et al., *Leaf disease classification in smart agriculture using deep neural network architecture and IoT*, Journal of Circuits, Systems and Computers, 31 (2022), no. 15, 2240004.
[28] M. R. Kumar, B. R. Devi, K. Rangaswamy, M. Sangeetha and K. V. R. Kumar, *IoT-Edge Computing for Efficient and Effective Information Process on Industrial Automation*, 2023 International Conference on Networking and Communications (ICNWC), Chennai, India, 2023, pp. 1–6.
[29] Potunarayana, sreedharbhukya, chandrashekarjatoth, p. Premchand, *Quality-aware Energy Efficient Scheduling Model for Fog Enabled IoT Network*, Computers and Electrical Engineering, 2021.
[30] V. K. A. Kumar, M. R. Kumar, N. Shribala, N. Singh, V. K. Gunjan, K. N.-e.-a. Siddiquee, and M. Arif, *Dynamic Wavelength Scheduling by Multiobjectives in OBS Networks*, Journal of Mathematics, 2022, Article ID 3806018.
[31] J. Luo, et al., *Container-based fog computing architecture and energy-balancing scheduling algorithm for energy IoT*, Future Generation Computer Systems, 97 (2019), pp. 50–60.
[32] S. M. Vadlamaani, P. K. Bharti, and M. R. Kumar, *Generalized Statistical Indicators For Cloud Computing Fault Tolerance*, International Journal of Intelligent Systems and Applications in Engineering, 10 (2022), no. 1s, pp. 269–281.
[33] M. Bhatia, S. K. Sood, and S. Kaur, *Quantumized approach of load scheduling in fog computing environment for IoT applications*, Computing, (2020), pp. 1–19.
[34] A. M. Fahim, et al., *An efficient enhanced k-means clustering algorithm*, Journal of Zhejiang University-Science A, 7 (2006), no. 10, pp. 1626–1633.

Edited by: Anil Kumar Budati
Special issue on: Soft Computing and Artificial Intelligence for wire/wireless Human-Machine Interface
Received: Sep 24, 2023
Accepted: Dec 23, 2023