

Received 9 March 2024, accepted 24 March 2024, date of publication 27 March 2024, date of current version 2 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3382214

## RESEARCH ARTICLE

# Enhancing Smart Factories Through Intelligent Measurement Devices Altering Smart Factories via IoT Infusion

OMAR ALRUWAILI<sup>1</sup>, FAN WU<sup>2</sup>, WAEL MOBARAK<sup>3,4</sup>,  
AND AMMAR ARMGHAN<sup>5</sup>, (Senior Member, IEEE)

<sup>1</sup>Department of Computer Engineering and Networks, College of Computer and Information Science, Jouf University, Sakaka 72388, Saudi Arabia

<sup>2</sup>College of Information Science and Technology, Zhejiang Shuren University, Hangzhou 310015, China

<sup>3</sup>Civil Engineering Department, University of Business and Technology, Ar Rawdah, Jeddah 23435, Saudi Arabia

<sup>4</sup>Engineering Mathematics Department, Alexandria University, Alexandria 11432, Egypt

<sup>5</sup>Department of Electrical Engineering, College of Engineering, Jouf University, Sakaka 72388, Saudi Arabia

Corresponding author: Ammar Armghan (aarmghan@ju.edu.sa)

This work was supported by the Deanship of Scientific Research at Jouf University through the Fast-Track Research Funding Program.

**ABSTRACT** The technology's integration into factories has accelerated automation's growth, creating autonomous working conditions and cutting-edge capacity for production. Modern and smart factories provide consumers with time-saving solutions and reliable outcomes. The present paper presents the concept of Event-Dependent Process Planning (EDPP), which seeks to improve the time-effectiveness of smart factories. The suggested approach automatically arranges planned and queued activities according to previous results, matching them with customer demands. Before process planning, essential data are provided by intelligent measuring equipment in the factories. Recurrent learning ensures the integrated process planning is successful and aligned with customers' needs. The efficiency with which the planning method exceeded customer expectations in earlier years is used to instruct this learning process. Applications of the technique are made to the manufacturing automation process's delivery and production layers. Essential metrics like processing time, response ratio, delivery delay, and backlogs are evaluated in an experimental analysis to validate the suggested process strategy. The proposed EDPP achieves 11.38% less processing time, 5.43% high response ratio, 10.18% less delivery delay, and 3.8% less backlog rate.


**INDEX TERMS** IoT, process planning, queuing and scheduling, recurrent learning, smart factory, cutting-edge capacity.

## I. INTRODUCTION

Industrial Internet of Things (IIoT) is the assimilation of Industry 4.0 and the recent Internet of Things (IoT) paradigm. The IIoT improves smart factory production and planning to meet consumer demands [1]. IoT is employed for consumer interaction, distributed industrial process analysis, data migration, and visualisation in industrial automation. The processing includes data collection, analyzing, and exchange of information among machine-to-machine (M2M) or machine-to-human (M2H) [2]. This leverages heterogeneous entities' data and process management in the industrial environment. IIoT uses cloud and edge computing, mobile

technologies, 3-dimensional production, robotics, RFID technology, etc. [3]. IIoT operates in five layers: Field, control, supervisory, planning, and management. Different production and manufacturing industries rely on IoT networks for reliable data storage and exchange [4]. The planning layer is responsible for sensing the data and deciding the interaction with the environment. The operational and maintenance cost for high-quality data transmission, analysis, and storage is optimal through distributed IoT functions. Real-time decision-making and visualisation are adopted from the IoT paradigm for better smart industry environment performance control [5].

The Internet of Things is a framework that enables several resources to be connected simultaneously. It can communicate with distributed systems, cloud computing, autonomous

The associate editor coordinating the review of this manuscript and approving it for publication was Young Jin Chun .

storage, and wireless sensor networks (WSN) [6]. The communication devices vary from mobiles, sensors, wearable computing devices, large processing servers, and virtual machines. The scope of IoT is to monitor and control the devices for better automation in an industrial environment [7]. IoT consists of three tiers: Device, Edge gateway, and cloud, which perform different functions inside and outside the smart industry. Based on these three tiers, the smart factories establish reliable communication with diverse heterogeneous devices [8]. This increases communication infrastructure without any delay or loss through appropriate communication technologies. IoT is used in many applications, such as Wearable sensors, smart cities, factories, homes, and healthcare systems. It increases the capabilities of smart factories in the network through different levels of human-to-computer interaction [9].

The data from various smart factory components is provided for analysis to make the best judgments about task production and planning. The choice is reached about how to distribute the work to the machines, product delivery, evaluation of customer feedback, and behaviour modelling by the previous task [10], [11]. The process analyses the patterns and updates of the running tasks in different layers of the IIoT. Machine learning in smart factories is used to discover and improve decision-making at all the process planning and analysis levels [12]. Some conventional analysis relies on computation and task offloading, queuing, resource allocation, etc. It uses the preceding data to analyse the resultant output through decisions and process planning [13]. This paper aims to balance the scheduling and delivery process to enhance the response ratio. The queue and response time are maintained throughout the process for better results to overcome and reduce the delivery delay.

## II. RELATED WORKS

Tang et al. [14] presented an industrial cloud and edge intelligence using a reconfigurable manufacturing method. This method is used for a multi-agent system under three processes: agent interaction, agent behaviour, and negotiation mechanism. The proposed method uses mixed-flow production done on random orders.

López et al. [15] provided as it offers a foundation for creating Industry 4.0 systems, the Reference Architectural Model for Industries 4.0 (RAMI 4.0) lacks tangible management platforms. The I4.0 platform for manufacturing that this study suggests offers infrastructure services for I4.0 system management. Due to its industrial agent foundation, the platform facilitates collaboration and negotiation. The unique feature of the platform is that it offers configurable AASs, which shortens development timeframes.

Computational offloading is performed in IoT- Edge-Cloud Computing Environments using the Multi-hop Cooperative method modelled by Hong et al. [16]. The multi-hop cooperative messaging mechanism (MCMM) is integrated with game theory to enhance the quality of service (QoS).

The initial step is to decrease the computation time of the process. Then, to improve the path, the Nash equilibrium is proposed.

Dai et al. [17] evaluated the effort to solve task offloading and service caching difficulties; the paper presents a framework for fog computing that is helped by the cloud for enterprise management systems (EMS). The framework reduces task latency and energy use by shifting work to local, fog, and cloud processing. By noncooperative game theory, the authors provide a distributed work offloading methodology and employ the 0-1 knapsack technique for dynamic resource caching. Several trials have confirmed the efficiency of the suggested algorithms.

Huo et al. [18] implemented an Industry 4.0 context to gather fuzzy control systems in smart factories. Real-time data are collected with two stages of fuzzy control: Type 1 and Type 2. The first is used to determine and satisfy the re-balanced data lines, and the second is used to enhance the machine.

Implementing the task orchestration is proposed by a hidden Markov model for smart shop floors developed by Ding et al. [19]. Autonomous manufacturing is used to interact with the work-in-progress (WIP) at the time of production. The Markov model is used to resolve a particular process flow.

Liu et al. [20] propose the digital twin-based model for the flow-types smart manufacturing system. Configuration, motion, control, and optimisation (CMCO) are two key methodologies for customised and software-defined design. They encapsulate the digital twin method.

Smart job shops in Industry 4.0 are done on the graphical deduction for producing instructional service systems introduced by Wang et al. [21]. The author developed a framework for key-based methods to make the necessary set of services in the system. These three layers are used: manufacturing resource, technical support, and Application layer.

Liu et al. [22], proposed a Cloud-based Advanced Planning and Scheduling (CAPS) System for developing the automotive parts in the industry. CAPS is used for intelligent dynamic planning and scheduling. It is used for a production planner to upload the CAPS data in the web page.

Sekhar et al. [23], displayed the soft sensor technique based on machine learning for auto industry lean manufacturing level prediction. It uses a database of 46 auto component companies in Pune, India, that have implemented lean manufacturing and related flexibilities. The greatest prediction accuracy of 80% was attained by the trilayered neural network design, which was followed by ensemble RUSBoosted trees, fine, medium, along with coarse trees, narrow as well as wide neural networks with quadratic and cubic SVMs, and broad and narrow neural networks.

Installation of industrial product-service systems (IPSS) is implemented in cloud-based resource planning tools is developed by Mourtzis et al. [24]. IPSS is used to produce a reliable link between the user and the environment. It resolves the complex integration of heterogeneous stakeholders.

A Smart manufacturing implementation system (SMIS) is presented by Zhang et al. [25], for top-level planning architecture. The first step is to learn the factory information through SMIS planning. Then, smart manufacturing is used to select the production control in smart factories.

Hamed et al. [26], demonstrated the Strain sensors, such as FBG, SAW, and MRs, are discussed in the paper as options for next-generation smart tools and components. Even though FBGs have been thoroughly examined, further study is still required to resolve device-specific problems, lessen data processing load, and enhance features. Consistent strain sensitivity, multimodal sensing, device downsizing, and affordable high-volume production are among the difficulties.

Dehnavi et al. [27], proposed a Fog integrated smart factory for reliability aware resources in the industrial application in real-time scenarios. This work decreases communication bandwidth by introducing the local private cloud with fog nodes: branch and bound-based exhaustive search algorithm.

Raza et al. [28] displayed arguably the most significant innovations for tackling global issues like energy consumption and global warming: smart energy or SE. Using smart meters to gather data for business intelligence (BI) incorporates IoT technologies with energy infrastructure. The incorporation of BI with data from smart meters is examined in this study, along with research gaps and drawback path. The following table 1 shows the clear understanding of the related works.

### III. EVENT-DEPENDENT PROCESS PLANNING

IoT-aided smart factories incorporate machines and devices for communication. It enhances the time-effective solutions for smart factories by using event-dependent process planning (EDPP). This proposed work aims to balance the scheduling, i.e., en-queue and delivery processes, throughout the entire progression. Recurrent learning is used to balance the two processes. The input is obtained from the sensor machine in the field layer and sent to the control layer. Figure 1 presents the overview of scheduling and process planning in IIoT architecture.

The initial step is to identify the current event in the machine. The scope is to decrease the backlogs and progression time in this way the delivery delay is reduced. If the delivery delay decreases the response ratio increases. To achieve this, the scheduling method is queued by incoming data and it is formulated by using the following (1).

$$e = i + \sum_t^p \frac{(q' + v)}{n} \quad (1)$$

In (1), the sensor data collection process is denoted as  $e$ , and the incoming data are represented as  $i$  and the data are acquired in the required time  $t$ . The processing of data is termed as  $p$ , the data are queued as  $q'$ .  $v$  is termed as an event, and  $n$  represents the number of incoming data. By using this, the scheduling is done and the data is queued. The data is

queued based on the time; the first incoming data is queued and followed by the impending data.

After scheduling the event identification is done, this is necessary to see whether the process is under certain evaluation. The running processes are derived by using the following (2) as follows. In this proposed work it uses two layers first is the field layer second is the control layer, which is used to sense and plan the data for communication.

$$s_o = \begin{cases} f_0 = \frac{q' + i}{e} * \left( \prod_{p=0}^n [t_i + h_i] \right) \\ r_0 = (q' * e) + \frac{\sum t_i}{n} \\ g_0 = (p + e) * \prod_n^v t_i + \frac{i}{l} \end{cases} \quad (2)$$

By using (1), the sensed data are queued as the scheduling process, in (2), the scheduling data identify the events. The events are classified into three types they are as follows.

- Offloading
- Overloading
- Dropping

Based on these three classifications the events are analyzed by using (2).  $s_0$  is denoted as scheduling. In Figure 2, the process of the above events in the IIoT is presented.

The offloading is denoted as  $f_0$  and there is formulated in the first case as  $\frac{q'+i}{e} * \left( \prod_{p=0}^n [t_i + h_i] \right)$ . In this, the offloading is used to share the data with the other machine to increase the progression time. It is evaluated as the incoming data are queued. From that, the data are not able to process in the required time so in this case, the process is shared  $h_i$  to the other machine.

The second case is overloading for this the perusing data are queued for processing the event. It is evaluated by  $(q' * e) + \frac{\sum t_i}{n}$  here; the process is completed in the required time, so the perusing data are queued for further processing. The third classification is dropping,  $(p + e) * \prod_n^v t_i + \frac{i}{l}$  in this, if the data are not able to complete in the required time means there leaves  $l$  from the machine. All these three classifications take place in two layers.

In the field layer, the data are sensed from the machines through a sensor represented by using (1). Then, the sensed data are sent for further data processing by the rest of the layers. The sensing is done in every layer, but it varies according to the structures. The preliminary layer knows how many machines are processing and obtains the related information.

The second layers sense the information regarding the planning and processing of the machine. The control layer measures the device processing in the queued data. Based on the classification of the data the response time should increase. They are done by balancing the scheduling and delivery of a process, which is achieved by using recurrent machine learning. It checks for the event in the machine for processing.

The table 2 represents pseudocode that explains how to apply recurrent learning with event-dependent process planning (EDPP). First, factors are set, then events are identified

**TABLE 1. Comparison of existing works with proposed EDPP framework.**

Study	Methodology	Key Features	Contributions
Tang et al. [14]	Reconfigurable method for manufacturing; Multi-agent system	Industrial Cloud and Edge Intelligence; Mixed flow production on random orders	Offers a reconfigurable method for manufacturing with multi-agent system approach
López et al. [15]	I4.0 platform for manufacturing; Reference Architectural Model for Industries 4.0 (RAMI 4.0)	Infrastructure services for I4.0 system management; Configurable AASs	Addresses the lack of tangible management platforms in RAMI 4.0 with an I4.0 platform
Hong et al. [16]	Multi-hop Cooperative method; Game-theoretic integration	Computational offloading in IoT-Edge-Cloud Computing Environments; Quality of Service enhancement	Introduces a Multi-hop Cooperative method integrated with game-theoretic to enhance QoS in computation offloading
Dai et al. [17]	Fog computing framework; Noncooperative game theory	Task latency and energy reduction; Dynamic resource caching	Presents a fog computing framework for EMS that reduces task latency and energy use via noncooperative game theory and dynamic resource caching
Huo et al. [18]	Fuzzy control system implementation in smart factories; Real-time data gathering with Type 1 and Type 2 fuzzy control	Context of Industry 4.0; Enhances machine performance	Implements fuzzy control systems in smart factories to improve machine performance within Industry 4.0 context
Ding et al. [19]	Hidden Markov model for smart shop floors; Autonomous manufacturing; Markov model for process flow	Task orchestration; Interaction with work-in-progress; Resolution of process flow	Develops a hidden Markov model for smart shop floors to orchestrate tasks and resolve process flow
Liu et al. [20]	Digital twin-based model for flow-type smart manufacturing system; CMCO methodologies	Customized and software-defined design; Encapsulation of digital twin method	Proposes a digital twin-based model with CMCO methodologies for flow-type smart manufacturing systems
Wang et al. [21]	Graphical deduction for instructional service systems in smart job shops	Key-based methods for manufacturing resource, technical support, and Application layer	Introduces a framework for smart job shops using graphical deduction and key-based methods for manufacturing services
Liu et al. [22]	Cloud-based Advanced Planning and Scheduling (CAPS) System	Intelligent dynamic planning and scheduling	Develops a CAPS System for intelligent dynamic planning and scheduling in automotive parts production
Sekha et al. [23]	Soft sensor technique based on machine learning for auto industry lean manufacturing level prediction	Database analysis; Prediction accuracy	Utilizes machine learning-based soft sensor technique to predict lean manufacturing levels in the auto industry
Mourtzis et al. [24]	Installation of industrial product-service systems (IPSS) in cloud-based resource planning tools	Reliable link between user and environment; Integration of heterogeneous stakeholders	Implements IPSS to create a reliable link between users and the environment, resolving integration challenges
Zhang et al. [25]	Smart Manufacturing Implementation System (SMIS)	Top-level planning architecture; Selection of production control in smart factories	Presents an SMIS for top-level planning and production control selection in smart factories
Hame et al. [26]	Discussion on strain sensors for next-generation smart tools and components	Strain sensor options; Device-specific problems; Challenges	Discusses strain sensor options and challenges for next-generation smart tools and components
Dehnavi et al. [27]	Fog-integrated smart factory; Real-time reliability-aware resources	Decreased communication bandwidth; Branch and bound-based exhaustive search algorithm	Implements a fog-integrated smart factory to decrease communication bandwidth and ensure real-time reliability-aware resources
Raza et al. [28]	Smart energy (SE) with smart meters and IoT technologies	Energy consumption reduction; BI integration with smart meters	Discusses smart energy innovations for energy consumption reduction and BI integration with smart meters
Proposed Work	Event-Dependent Process Planning (EDPP)	Automating process planning; Aligning activities with customer demands	Introduces EDPP to enhance time-effectiveness in smart factories by automating process planning and aligning activities with customer demands.

from sensor data, scheduling is arranged, types of events are determined, recurrent learning is used to improve how they perform, choices are taken regarding dropping, overloading, or offloading, hidden layer generation, computation reduction, handle equalization, the parameter modification, and the process is repeated for continuous improvement. IoT-assisted smart factory aim to balance scheduling and delivery procedures to guarantee effective workflow and enhanced performance. Recurrent learning is used to adjust

to occurrences to enhance the system's overall efficiency. For continuous enhancement, the process is iterated.

Smart measuring devices in advanced factories gather data on production, quality, energy usage, maintenance, and supply chain. This information is used in Event-Dependent Process Planning (EDPP) to improve production scheduling, identify obstacles, allocate resources effectively, and forecast and avoid delays. EDPP changes process queues as needed, distributes workloads evenly, and deals with challenges in



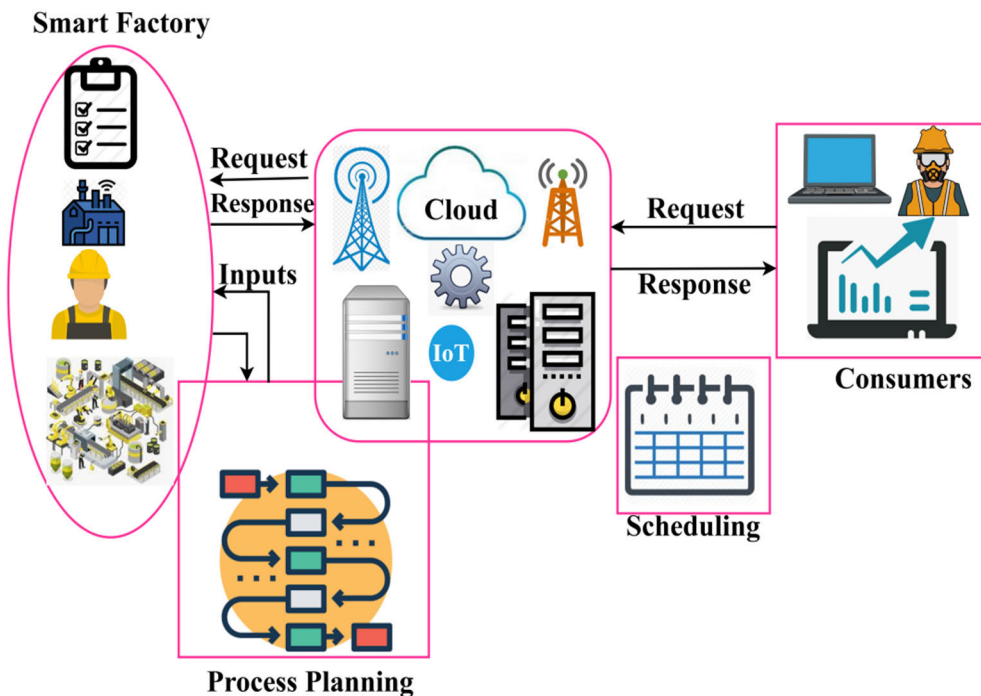


FIGURE 1. Process Planning in IIoT.

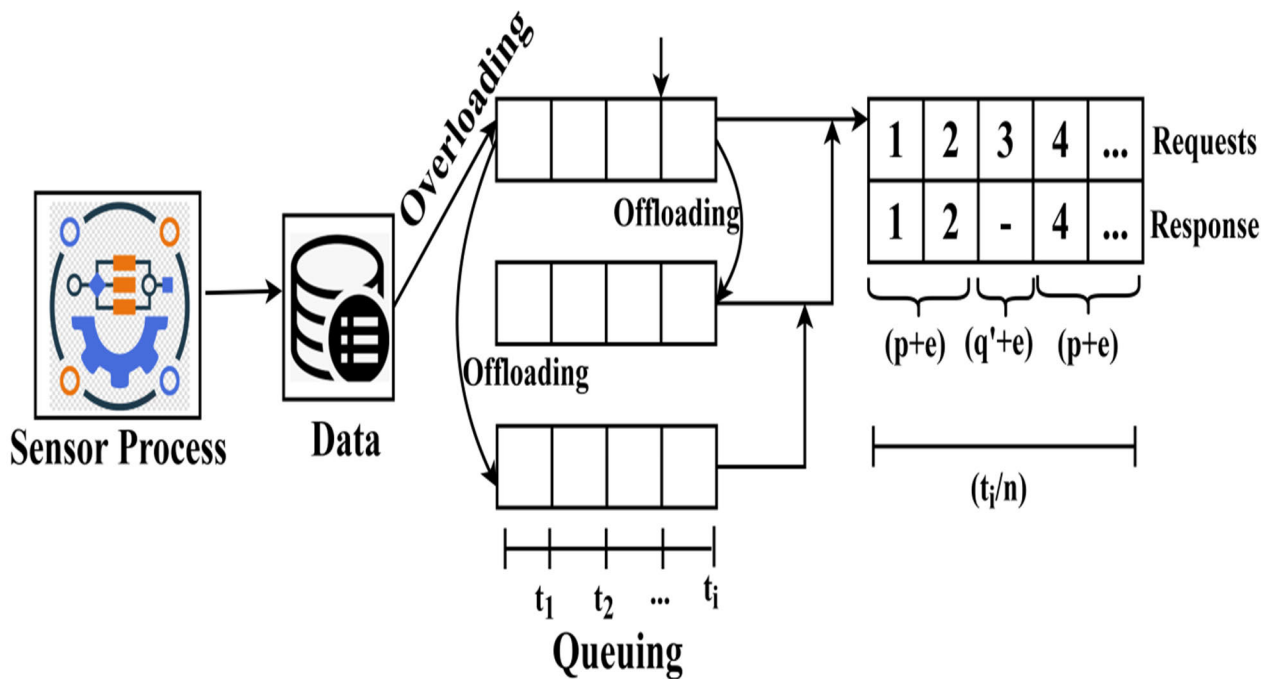


FIGURE 2. Event Process in IIoT.

advance to improve efficiency and product quality. Using real-time data from smart measuring devices, EDPP enhances decision-making, lowers processing time, minimizes supply delays, and ensures efficient production processes in advanced factories.

**A. EDPP- RECURRENT LEARNING**

Recurrent learning is used to verify the balance between the customer and the process in the machine. It monitors the event and increases the response ratio. The control must be done if the queuing increases the response ratio decreases and

TABLE 2. Pseudocode for event-dependent process planning.

Pseudocode for Event-Dependent Process Planning	
# Recurrent Learning for Event-Dependent Process Planning	
# Steps:	
1.	Initialize parameters and variables with initialize_parameters()
2.	Utilizing sensor data to recognize events event_identification(data) = event
3.	Organizing in response to receiving data scheduling(data) = s_result
4.	Determining events (dropping, overloading, and offloading) classify(result) = type
5.	Using recurring learning to identify events and enhance performance rec_learning(data, type)
6.	Deciding either to drop, overload, or offload dec_result = m_decision(data, type)
7.	The recurrence network generates layers that are hidden by using the formula layers(data, type) <=
	$s_o = \begin{cases} f_0 = \frac{q' + i}{e} * \left( \prod_{p=0}^n [t_i + h_i] \right) \\ r_0 = (q' * e) + \frac{\sum t_i}{n} \\ g_0 = (p + e) * \prod_{i=1}^v t_i + \frac{i}{l} \end{cases}$
8.	Reducing the method computations for a more effective workflow drop_method = compute_dropping_method(type, data)
9.	Utilizing recurrent learning equalize_processes(data, type), balance
10.	Modify parameters to improve performance set_parameters()
11.	Carry out the procedure once again for ongoing enhancement

vice versa. To avoid this following machine learning is used. The offloading, overloading and dropping are identified and enhance the performance of the machine. The EDPP is used to plan the event while processing and it is based on (1). The following (3) is used to identify the offloading.

$$\theta(f_0) = e + i * \begin{cases} \sum_{t_i}^v [p + q'] * \frac{h}{n} = 0 \\ \sum_{t_i}^v [p + q'] * \frac{h}{n} \neq 0 \end{cases} \quad (3)$$

By using (2), the scheduling is performed from the sensed data, from that the offloading is detected in (3). In these conditions the first state denotes, if the queued process is still working at the same time, if the new process enters the queue means it shares the data to the subsequent machine.  $\sum_{t_i}^v [p + q'] * \frac{h}{n} = 0$ , states the resultant is equal to 0, which means the machine is ideal so allocate the data to increases the performances.

The second case is  $\sum_{t_i}^v [p + q'] * \frac{h}{n} \neq 0$ , here it is not equal to 0, so it states that the machine is busy processing the

data. Here, in offloading the first case satisfies the condition. If the event is identified, then it gives high performance at the particular stage in the machine. In this way, every machine is assigned a process in offloading by sharing methods. The second condition is used as the training data in recurrent learning. The overloading is detected by using the following (4) as follows.

$$\theta(r_0) = p - t_v \begin{cases} \prod_e^{s_0} [q' * e_i] < t \\ \prod_e^{s_0} [q' * e_i] > t \end{cases} \quad (4)$$

From (3), the offloading is calculated, in (4), the overloading is derived in two stages. Figure 3 illustrates the learning process for classifying offloading events.

The first stage is  $\prod_e^{s_0} [q' * e_i] < t$ , if the process enters the machine and already the machine is processing the data means the new process is en-queued. In this way, if the process is not completed in the assigned time means it denotes that the process is taking too long time to finish. In this case, the response ratio increases, and hence, time management is

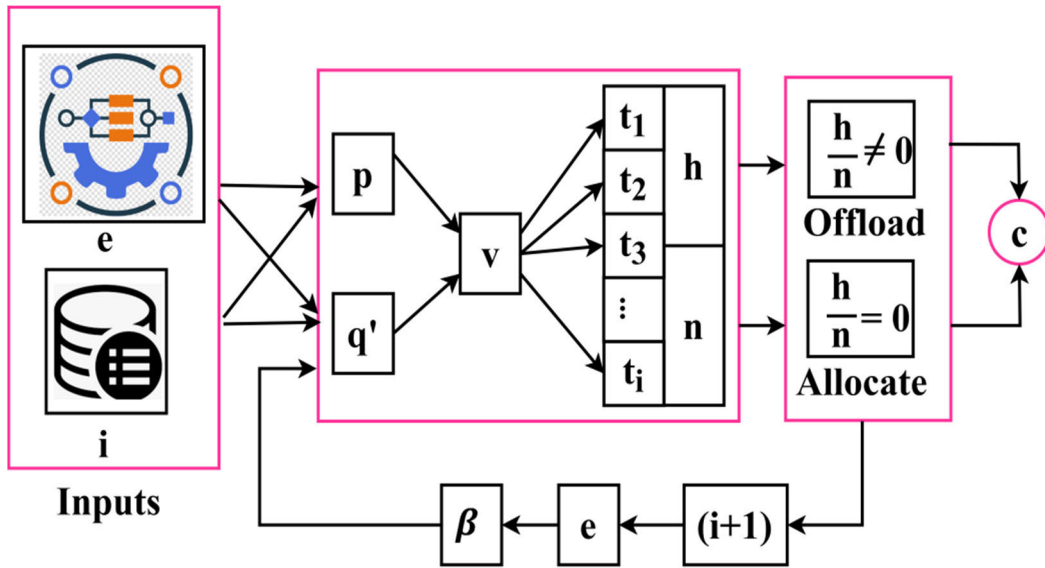


FIGURE 3. Offloading-First Stage.

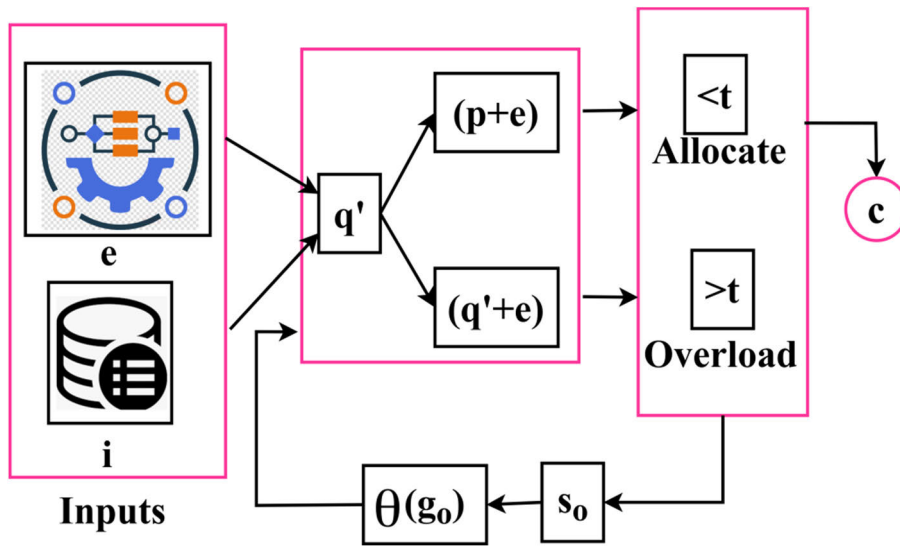


FIGURE 4. Overloading Analysis-Second Stage.

necessary to improve the overloading process. The second stage of overloading analysis in a recurrent manner is presented in Figure 4.

The second stage is  $[\prod_e^{s_0} [q' * e_i] > t$ , here, the queued process is completed in the assigned time. Where the time is lesser than the process it states that machines should perform further evaluation. The third classification is dropping; it affects the success rate. In this stage, it should process either offloading or overloading. The following is used to formulate the dropping method in EDPP.

$$\theta(g_0) = \begin{cases} 1, q' + \prod_e^{s_0} [p - l] \\ 0, otherwise \end{cases} \quad (5)$$

In (4), the overloading measures the sensed data in the assigned time. (5) is used to drop data from the queue. In recurrent learning, the analysis refers to the previous data from the hidden state. To improve the success, rate the dropping must be a focus on either offloading or overloading method. The following (6) derives the dropping to increase the success rate using recurrent learning based on existing and preceding data.

$$c = j(o_t, \theta [f_0, r_o, g_o]) + \mathcal{H} \quad (6)$$

(5) is used for dropping data in the queue on two conditions. In (6), the existing and preceding data are computed as  $c$  and  $o$ . The function is termed as  $j$  to decide whether the dropping is done on overloading or offloading. It uses the

hidden layer for retrieving the preceding data and it is denoted as  $\mathcal{H}$ . By using the above (6), it reduces the backlogs, which is said to be a stagnate data in the queue. The process is not performed for a long time but remains in the queue and it is derived by using (7).

$$\beta = \frac{\theta [s_0] + \prod (q' * t_i) + (n * v)}{c} \tag{7}$$

By using (7), the backlogs are maintained by fixing the time for every process in the queue. It reduces the backlogs  $\beta$  and maintains the queue and response ration. By using (6), the existing and preceding data are evaluated. If the existing data are obtained in the assigned time means the queue process is maintained properly. If it takes more time means the delivery delay attains, in this, it addresses the backlogs in the machine. The EDPP takes the appropriate step to improve the process. The queuing and backlog should be maintained After the backlogs, the decision making is done for balancing the scheduler and delivery process by utilizing three classifications.

The decision is made on how much time does the data tends to process in the machine. It is calculated by using fixed time for doing this operation. The following (8) is used to formulate the decision-making process in recurrent learning. It uses (6) and (7) for evaluating the result.

$$\alpha = \beta + \begin{cases} (o_r, \theta[f_0]) + \frac{h}{n} < \rho \\ (o_r, \theta[r_0]) * \prod_e^{s_0} [q'] = \rho \\ (o_r, \theta[g_0]) + [p - l] > \rho \end{cases} \tag{8}$$

In (6), the existing and preceding data are computed, and the decision is done by using (8). In this first condition is  $(o_r, \theta[f_0]) + \frac{h}{n} < \rho$ , here the offloading is computed in the assigned time. The overloading is the second condition  $(o_r, \theta[r_0]) * \prod_e^{s_0} [q'] = \rho$ , in this, the time taken is equal to the processing of the data. So, it is not considered for processing, the third condition is the dropping is greater than the assigned time. Form this first importance is given to the

offloading process because the computation is done on the assigned time.

By evaluation this above (8) the dropping must decide which method to be performed to get the success rate. The dropping might be overloading or offloading classification. The weights and biases are improved by using the activation function. The following (9) is used for calculating the activation function.

$$\mu = c_t + \langle \tanh[w_o] * \alpha \rangle + \langle [w_o(\theta)]t \rangle \tag{9}$$

The above (9) is used for calculating the activation function for the recurrent network. The activation function is represented as  $\mu$ , and the weight of the neuron is denoted as  $w_0$  and the recurrent neuron is termed as  $\partial$ . The formula is evaluated in the tanh activation function, here the weight of neuron and detection of classification methods are determined. It is determined based on the fixed time  $\alpha$ .

The Recurrent is done by training the data in the network based on the input and output data. It is maintained by using hidden layers, here, it is associated with two hidden layers. There are used for increases the response ratio which is the scope of this work. The following (10) and (11) are used as the hidden layers in this network.

$$\mathcal{H}_1 = \begin{cases} \Delta_1 = [(s_0 + e) * q'] + w_o \\ \Delta_2 = \Delta_1 + q' * (p_r - n) + \theta \\ \vdots \\ \Delta_m = \Delta_2 * e + i - \Delta_{m-1} \end{cases} \tag{10}$$

$$\mathcal{H}_2 = \begin{cases} \Delta_1 = w_o[s_o + e - n] \\ \Delta_2 = \Delta_1 * q' + \theta \\ \vdots \\ \Delta_m = \Delta_2 + \alpha - \Delta_{m-1} \end{cases} \tag{11}$$

The above two (10) and (11), the hidden layer is calculated. The variables  $\mathcal{H}_1$  and  $\mathcal{H}_2$  represent the two hidden layers. Similarly,  $\Delta$  it denotes the layers and  $m$  denotes the number

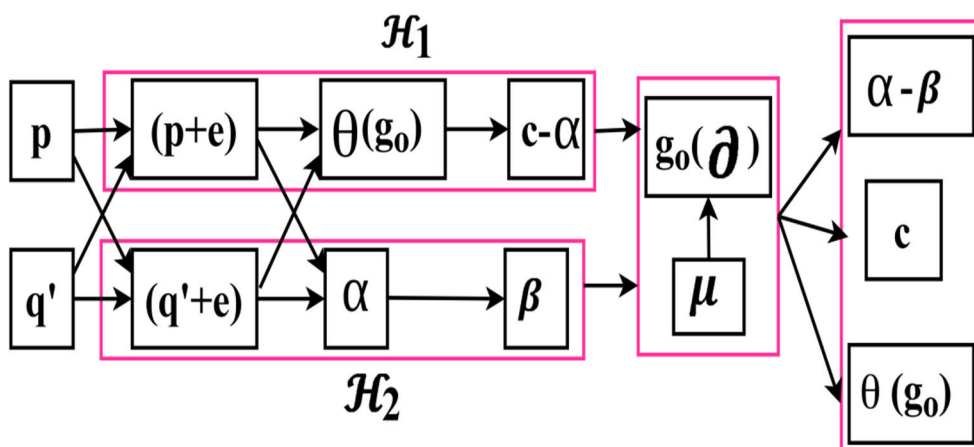


FIGURE 5. Hidden Layer Processing for IoT Offloading Decisions.



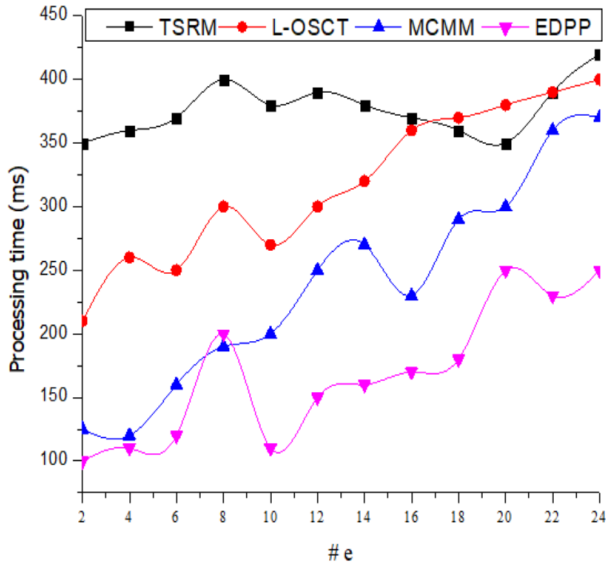


FIGURE 6. Processing Time Comparisons.

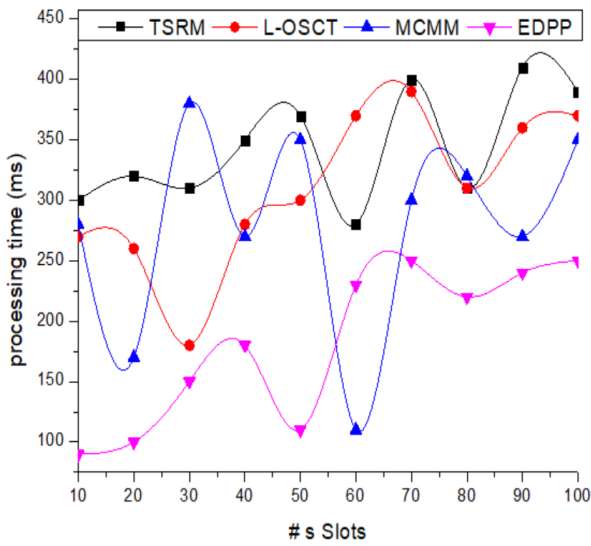


FIGURE 7. Processing Time Comparisons.

of hidden layers. Based on these hidden layers the data are being trained. It is used for the recurrent network to improve the response ratio. By using this, the dropping classification is calculated to enhance the success rate. The hidden layer process based on different conditions is illustrated in Figure 5.

The hidden layer process in the recurrent network as when the offloading is performed the error occurs is sent back to the first hidden layer as the training set. If the second process is overloading means, the output error data are again sent to the hidden layer. The aim of designing the hidden layers is used to get the appropriate result by matching the preceding data in the layer. In this way, the recurrent learning is processed based on the preceding data the output of the first hidden layer is given as the input to the second hidden layer.

The following (12) is used for obtaining the dropping method for an efficient method. The recurrent machine learning is used to achieve the efficient result and improve the process in the scheduled queue, by decreasing the processing delay (delivery) in the network. The objective is satisfied by using the backlogs by using (7). It is updated for every incoming process enters the queue.

$$g_0(\partial) = \begin{cases} \left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] * \mu + \mathcal{H}_1 + \mathcal{H}_2 = \alpha_\rho \\ \mu \left[ \frac{\sum t_i}{n} + q' * e_i \right] \mathcal{H}_1 + \mathcal{H}_2 \neq \alpha_\rho \end{cases} \quad (12)$$

From (10) and (11), the hidden layers are observed for the recurrent layers. By using this, the above (12),  $\alpha_\rho$  represents the fixed time for making a decision. This equation is used to decide the dropping can be either overloading or offloading methods. It uses two conditions the first level is  $\left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] * \mu + \mathcal{H}_1 + \mathcal{H}_2 = \alpha_\rho$  here the offloading and overloading are used for processing the data. In a queue, the data are evaluated by the fixed time if it does not process in the fixed time and weight of the data. It means the first condition satisfies the processing time and data weight.

The second condition is  $\mu \left[ \frac{\sum t_i}{n} + q' * e_i \right] \mathcal{H}_1 + \mathcal{H}_2 \neq \alpha_\rho$  here, they are having the time, and the weight of the data is done by using (9). This condition does not satisfy the case. The results denote the dropping includes the offloading methods because the above equation satisfies the first condition. The offloading performs better because if it has more process to complete it shares the process to the other machine.

In this way, the offloading is reliable compared to the overloading. After the dropping decision is made the balancing is done for the scheduling and delivery process. It is done by acquiring two hidden layers in (10) and (11). The following (13) is used to evaluate balancing the two processes by using recurrent learning.

$$\partial = \begin{cases} \left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] + \mathcal{H}_1 + \mathcal{H}_2 * q' < R \\ \left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] + \mathcal{H}_1 + \mathcal{H}_2 * q' > R \end{cases} \quad (13)$$

From (12), the decision is made for the dropping method, using this the recurrent learning obtains the better result by using (13). In (13) it indicates two levels, the initial condition is  $\left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] + \mathcal{H}_1 + \mathcal{H}_2 * q' < R$ , the response ratio is denoted as R. The first level satisfies the condition as the process done on the machine achieves the more response time.

The second level is  $\left[ \prod_{p=0}^n [t_i + h_i] + \frac{h}{n} \right] + \mathcal{H}_1 + \mathcal{H}_2 * q' > R$ , here there the process takes more response ratio. It does not satisfy the conditions. In this case, again they process (8) and (9). By doing this the queue and the response ratio are balanced by deriving (2). The proposed

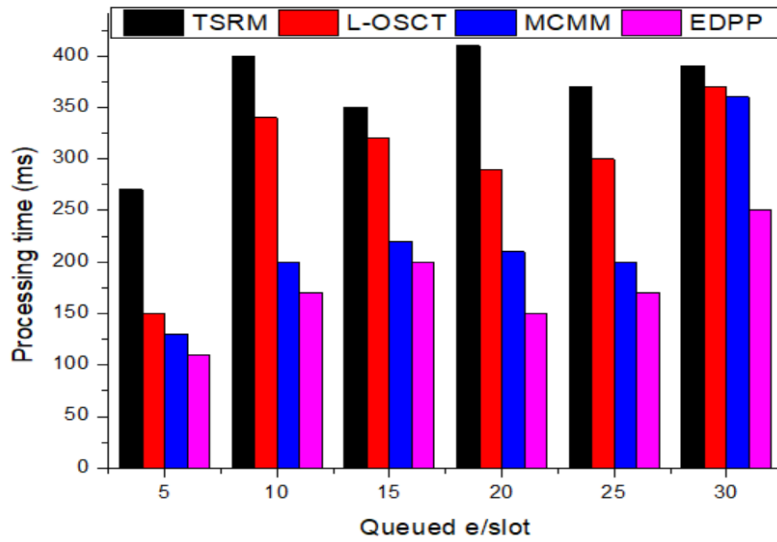


FIGURE 8. Processing Time Comparisons.

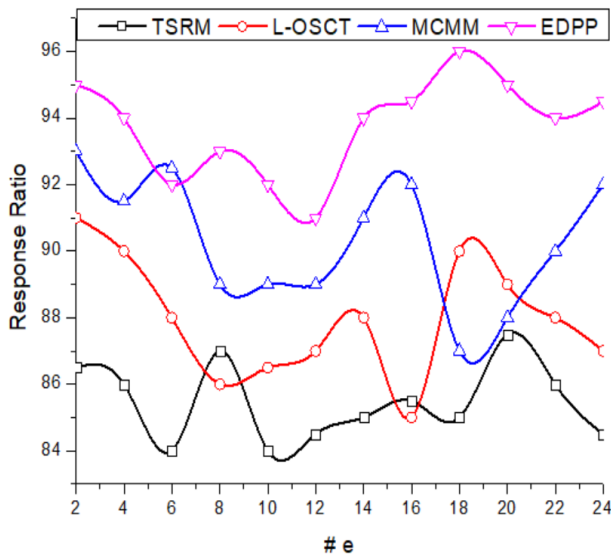


FIGURE 9. Response Ratio Comparisons.

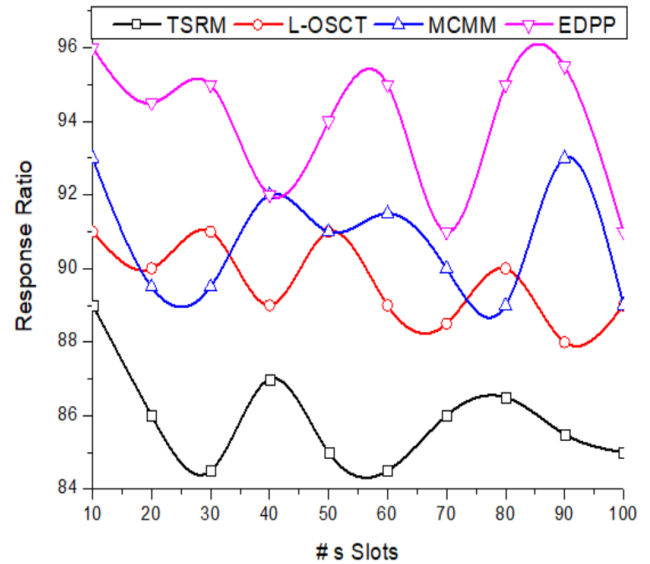


FIGURE 10. Response Ratio Comparisons.

objective is satisfied by using (13) and obtains an increase in response time. In this, both the scheduling and delivery process are balanced through the process completes. Thus, the IoT in smart factories performs using field and control layers and improves the process planning from the measuring devices.

#### IV. PERFORMANCE ANALYSIS

In this section, the performance of the proposed EDPP is analyzed. The smart factory environment is simulated using the Contiki Cooja simulator with a pack of 32 machine sensors. An aggregator set of 8 devices is capable of organizing the sensor data based on time. The sensor data requires augmentations of queuing and process scheduling depending

on the available queuing slots of 10-100. Each queuing slot is capable of gaining 2-24 processes such that a maximum of 30 processes is queued at a single slot for processing. The analyzing and dissemination system handles a maximum of 40 processes and the same in the maximum slot interval of 8 mins is responded. The offloading limit is set as 12 processes and the maximum time for response is 18 s. Using this experimental setup, the proposed EDPP is analyzed for the metrics processing time, response ratio, delivery delay, and backlogs. To perform a comparative analysis, the above metrics are considered for the methods TSRM [17], L-OSCT [23], and MCM [16], which are discussed in the related works section.

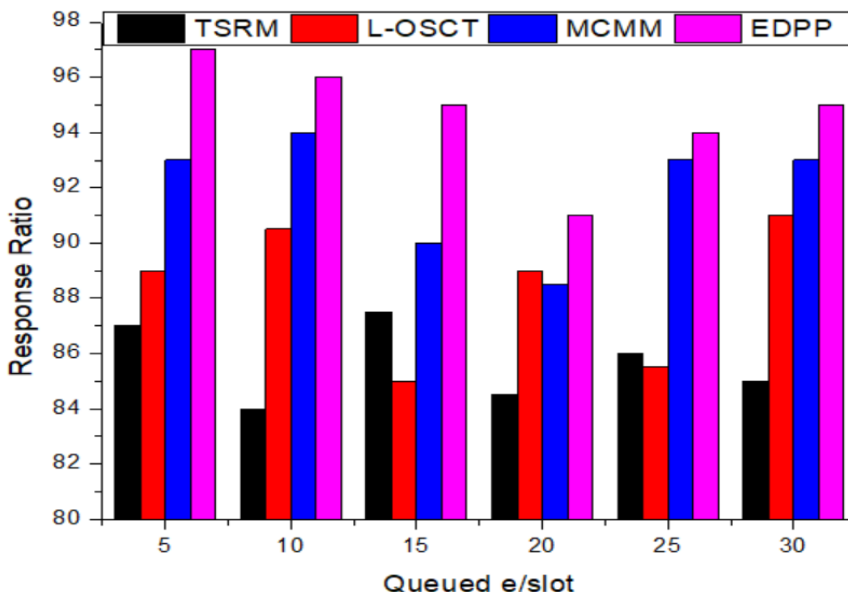


FIGURE 11. Response Ratio Comparisons.

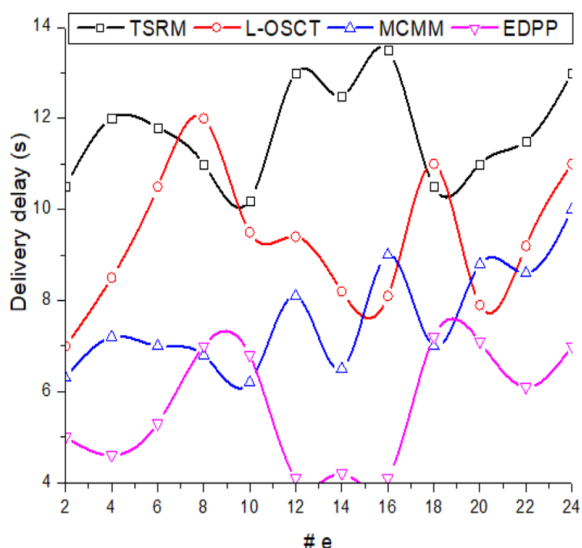


FIGURE 12. Delivery Delay Comparisons.

**A. PROCESSING TIME**

The acquired sensor data in each processing event is assigned to the appropriate scheduling instances based on  $s_o$ . This classification determines the overloading/ offloading scenario by identifying appropriate slots for processing data. The different classification of  $f_o$  and  $r_o$  assigns free slots for the active requests. Similarly, the available free slots adapt to the incoming  $e$  such that no overloading/ drop is experienced. Unlike the other methods, scheduling is not sequential/ invariant for different concurrent processes of offloading and overloading. Therefore, concurrency in this method increases the chances of higher processing. The significant reason for wait time-less process assignment and estimation helps to reduce the

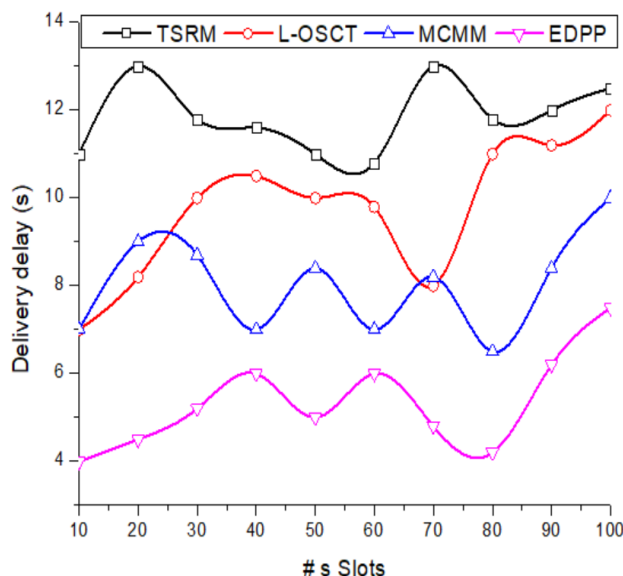


FIGURE 13. Delivery Delay Comparisons.

number of failed analyses for the varying inputs (Figure 6) and varying slots (Figure7). In case of varying queued  $e$ / slot, the conditional analysis of  $< t$  or  $> t$  [ (4)] and  $\theta(f_o) = 0$  or  $\neq 0$  [ (3)] for offloading is performed recurrently. This recurrent analysis is performed for both the  $f_o$  and  $r_o$  in a concurrent manner. Therefore, the processing time is shared and hence is less for even the enqueued process (Figure 8).

**B. RESPONSE RATIO**

Figures 9,10, and 11 represents the comparative analysis of response rate in concern to varying processes, slots, and

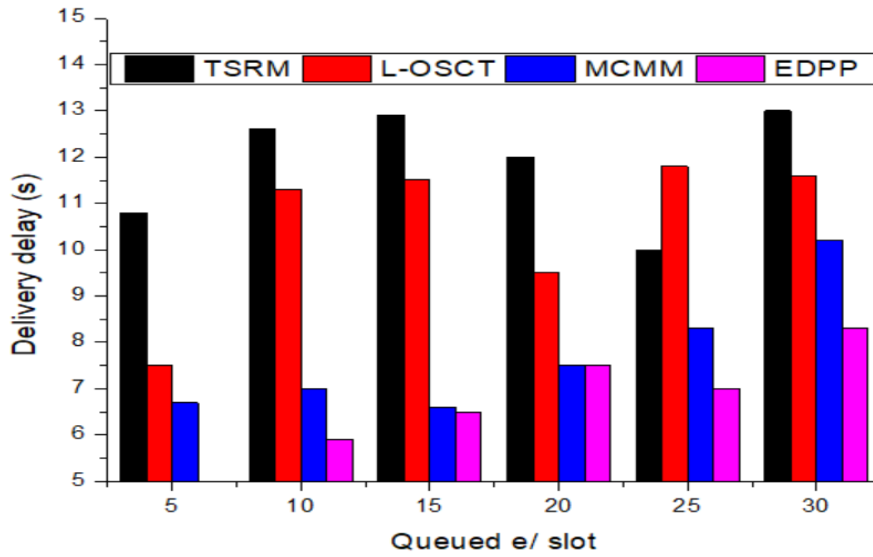


FIGURE 14. Delivery Delay Comparisons.

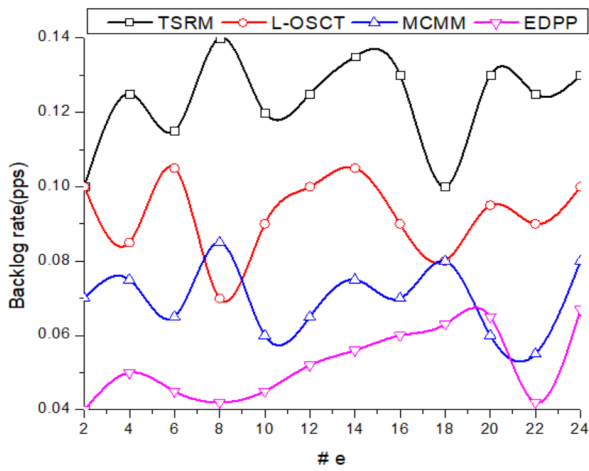


FIGURE 15. Backlog Comparisons.

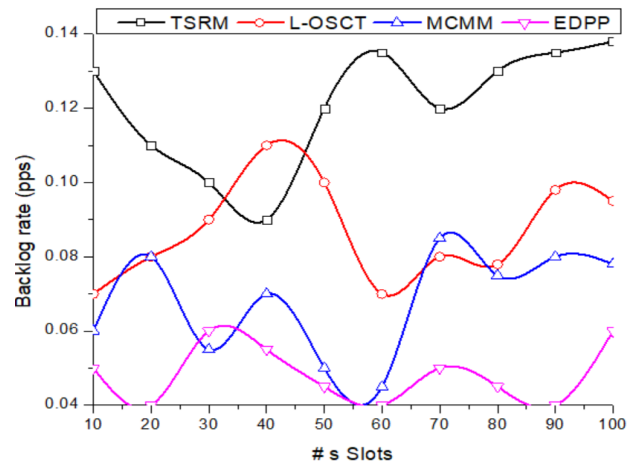


FIGURE 16. Backlog Comparisons.

process in either  $f_o$  and  $r_o$ . This classification is useful in reducing  $g_o$  based on refining  $\mathcal{H}$  based output  $c$ . The condition is analyzed for  $\theta[f_o] + \frac{h}{n} < \rho$  and  $\prod[q'] = \rho$  to extract more feasible  $c$  from the existing solutions of  $\theta[f_o]$  such that a smaller number of responses is backed off. The mediate offloading requirements and overloaded queuing is identified using the hidden layer outputs estimated using  $s$  (10) and (11). For the overloaded processes, the solution is filtered as  $g_o(\theta)$  and  $\theta$  validates if the responses to request ratio. The response ration from the learning process decides offloading or allocation or overloaded instances of the queuing and scheduling of different acquired processes. Therefore, both  $\theta[g_o]$  and  $\theta[f_o]$  along with  $g_o(\theta)$  is capable of providing a maximum response to the acquired requests. Both the offloaded and overloaded instances are capable of verifying  $\theta$  and  $g(\theta)$  in achieving a high response rate.

C. DELIVERY DELAY

The maximum time for response in the simulation is set as 18s. With is the analysis is performed to verify if EDPP is delay conscious. The delay of the proposed method is reduced by independently analyzing  $\mathcal{H}_1$  and  $\mathcal{H}_2$  using  $(p + e)$  and  $(q' + e)$  constraints. The number of process events and the varying slots is synchronized to ensure  $g_o(\theta)$  is achieving in both  $(c - \alpha)$  and  $\alpha$ (alone) validations. The overloaded instances are extracted from the conditions satisfying  $(q' + e) > t$  and  $\frac{h}{n} \neq 0$  where the slots are mapped to the existing requests other than the new (queued) requests. Therefore, considering the time of  $t_i$  for  $n$ , the delay observed in  $(t_i/n)$  instance is less, compared to the other methods. This is unanimous for varying process events (Figure12), slots (Figure 13), and process/ slot (Figure 14). In all these instances, the recurrent analysis aims to differentiate

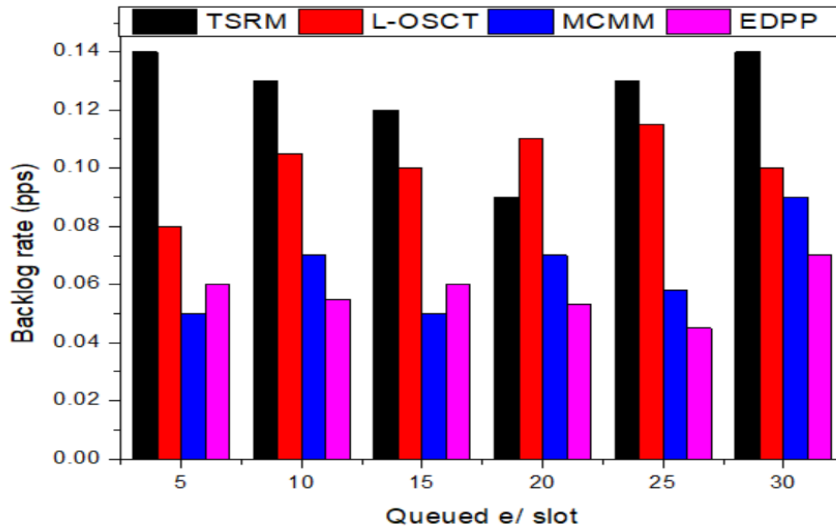


FIGURE 17. Backlog Comparisons.

TABLE 3. Comparative analysis for varying e, varying s slots, varying queued e/slot.

Metrics	Varying e				Varying s Slots				Varying Queued e/ Slot			
	TSRM	L-OSCT	MCMM	EDPP	TSRM	L-OSCT	MCMM	EDPP	TSRM	L-OSCT	MCMM	EDPP
Processing Time (ms)	410.98	389.47	367	251.76	409.17	378.71	367.65	254.26	398.38	380.49	354.35	248.83
Response Ratio	85.44	87.57	92.17	93.43	84.96	88.04	90.83	92.77	85.25	90.75	92.36	94.88
Delivery Delay (s)	12.63	10.76	9.97	7.24	12.93	11.7	9.93	7.56	12.61	11.18	9.81	7.78
Backlog Rate (packets per second-pps)	0.135	0.098	0.084	0.063	0.138	0.098	0.079	0.062	0.14	0.097	0.072	0.065

overloaded and dropping processes. Such differentiation helps to allocate available queuing and scheduling slots based on their availability and the capability of the processing systems. Therefore, the prolonged waiting/ paused process analysis is prevented by identifying the event of the process and its appropriate slot, reducing delay.

#### D. BACKLOG

The proposed EDPP achieves fewer backlogs compared to the other methods for varying e slots and queuing/ slot [17, Refer to Figures 15, 16]. There are two prime reasons for suppressing backlogs viz: initial classification of  $g_o, f_o$  and  $r_o$  based on the sequence of e and available time slots, and offloading. The initial classification relies on the occurrence of  $r_o$  in any of e process, such the offloading is performed. Contrarily, if an extracted c is the solution, then  $\partial$  and  $g(\partial)$  are the validating conditions to ensure if the request/ response is balanced. This conditional verification is performed in a recurrent manner for  $\mathcal{H}_1$  and  $\mathcal{H}_2$  using different conditions. The mediate mapping conditions based on  $(p + e)$  and  $(q' + e)$  help to identify all the possible allocations where the incoming e is assigned to the free slot. The free slots with the

conditions  $\frac{h}{n} = 0$  and  $q' > t$  are selected for assigning the overloaded requests and therefore,  $\beta$  from  $c \in (p + e)$  is the  $r_o$  backlog. In the case of  $g_o$  and  $\partial$  verification,  $(\alpha - \beta)$  is the final output of c achieving fewer backlogs. In Tables 3, the comparative analysis results in concern to the varying e, slots, and queuing e/ slot is presented.

In concern to the varying e, the proposed EDPP achieves 11.77% less processing time, 5.04% high response ratio, 11.63% less delivery delay, and 4.27% less backlog rate. In concern to the varying e, the proposed EDPP achieves 11.32% less processing time, 4.83% high response ratio, 11.46% less delivery delay, and 4.3% less backlog rate. In concern to the varying e, the proposed EDPP achieves 11.38% less processing time, 5.43% high response ratio, 10.18% less delivery delay, and 3.8% less backlog rate.i

The setup for testing the suggested Event-Dependent Process Planning (EDPP) technique included creating a virtual smart manufacturing setting using the Contiki Cooja simulator with 32 machine detectors and 8 aggregator devices. Data collection methods involved collecting data on the rate of production, quality measures, energy usage, maintenance plans, and supply chain activities from the advanced measuring



devices in the simulated intelligent factory. Statistical methods like comparing data, analyzing past records, and evaluating measurements were used to evaluate how well the EDPP plan performed. The investigations assessed processing time, reaction ratio, delivery delay, as well as backlog rates to confirm the usefulness of the suggested EDPP method in enhancing operational efficiency and decreasing delays in smart factory operations.

The evaluation of the suggested Event-Dependent Process Planning (EDPP) approach against TSRM, L-OSCT, and MCM in smart production situations might be based on their importance, recognized norms, data accessibility, methodological resemblances, and research extent. These approaches might be selected because of their particular emphasis on smart manufacturing settings, past industry acknowledgment, or common traits with the suggested EDPP methodology. The selection could also take into account the accessibility of data for contrast and how well it aligns with the study aims.

## V. CONCLUSION

This article discusses the event-dependent process planning that is focused on improving the performance of the industrial internet of things. In this method, the data process from the machine sensors is classified for overloading or offloading at an early stage to prevent process backlogs. The diverse queuing and scheduling conditions are recurrently analyzed to ensure the request and responses are balanced for the analyzed process. Recurrent processing based on a different slot and process allocation helps to reduce the processing time by a concurrent allocation of requests and its associated data. The pre-classification of process instances and the allocated slots refines more feasible outputs through appropriate offloading guided by the recurrent analysis, improving the response rate. The experimental analysis verifies the consistency of the proposed method by improving the response rate and reducing processing time, delay, and backlog.

Future research areas involve investigating more advanced optimization methods, incorporating machine learning for forecasting, improving security in multi-hop flexible approaches, emphasizing scalability and managing resources, creating real-time monitoring with IoT, enhancing interoperability, studying energy-efficient computing, and carrying out validation research. These pathways seek to improve work delegation in IoT-Edge-Cloud settings, boost system efficiency, guarantee data protection, and encourage sustainability in intelligent manufacturing. By focusing on these areas, academics may help improve and make more safe industrial automation methods, pushing forward the development of IoT technology in manufacturing processes.

## REFERENCES

[1] D. Sun, R. Huang, Y. Chen, Y. Wang, J. Zeng, M. Yuan, T.-C. Pong, and H. Qu, "PlanningVis: A visual analytics approach to production planning in smart factories," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 579–589, Jan. 2020.

[2] C. Wang, G. Zhou, and Z. Zhu, "Service perspective based production control system for smart job shop under industry 4.0," *Robot. Comput.-Integr. Manuf.*, vol. 65, Oct. 2020, Art. no. 101954.

[3] S. Aheleroff, X. Xu, and Y. Lu, "IoT-enabled smart appliances under industry 4.0: A case study," *Adv. Eng. Inform.*, vol. 43, Jan. 2020, Art. no. 101043.

[4] J. F. Buhl, "A dual-arm collaborative robot system for the smart factories of the future," *Proc. Manuf.*, vol. 38, pp. 333–340, 2019.

[5] A. Munín-Doce, V. Díaz-Casás, P. Trueba, S. Ferrero-González, and M. Vilar-Montesinos, "Industrial Internet of Things in the production environment of a shipyard 4.0," *Int. J. Adv. Manuf. Technol.*, vol. 108, nos. 1–2, pp. 47–59, 2020.

[6] S. Marzia, AlejandroVital-Soto, and A. Azab, "Automated process planning and dynamic scheduling for smart manufacturing: A systematic literature review," *Manuf. Lett.*, vol. 35, pp. 861–872, Aug. 2023.

[7] H. Besharati-Foumani, M. Lohtander, and J. Varis, "Intelligent process planning for smart manufacturing systems: A state-of-the-art review," *Proc. Manuf.*, vol. 38, pp. 156–162, 2019.

[8] Y. Yi, Y. Yan, X. Liu, Z. Ni, J. Feng, and J. Liu, "Digital twin-based smart assembly process design and application framework for complex products and its case study," *J. Manuf. Syst.*, vol. 58, pp. 94–107, Jan. 2021.

[9] L. Kiefer, P. Voit, C. Richter, and G. Reinhart, "Attribute-based identification processes for autonomous manufacturing systems—An approach for the integration in factory planning methods," *Proc. CIRP*, vol. 79, pp. 204–209, 2019.

[10] Y.-J. Lin, S.-H. Wei, and C.-Y. Huang, "Intelligent manufacturing control systems: The core of smart factory," *Proc. Manuf.*, vol. 39, pp. 389–397, 2019.

[11] J. Zhang, J. Zhang, M. Zhong, J. Zheng, and L. Yao, "A GOA-MSVM based strategy to achieve high fault identification accuracy for rotating machinery under different load conditions," *Measurement*, vol. 163, Oct. 2020, Art. no. 108067.

[12] R. E. Andersen, "Self-learning processes in smart factories: Deep reinforcement learning for process control of robot brine injection," *Proc. Manuf.*, vol. 38, pp. 171–177, 2019.

[13] E. Kim, D.-H. Huh, and S. Kim, "Knowledge-based power monitoring and fault prediction system for smart factories," *Pers. Ubiquitous Comput.*, vol. 26, pp. 307–318, Dec. 2019.

[14] H. Tang, D. Li, J. Wan, M. Imran, and M. Shoaib, "A reconfigurable method for intelligent manufacturing based on industrial cloud and edge intelligence," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4248–4259, May 2020.

[15] J. Wang, F. Hu, G. Abbas, M. Albekairi, and N. Rashid, "Enhancing image categorization with the quantized object recognition model in surveillance systems," *Expert Syst. Appl.*, vol. 238, 2024, Art. no. 122240, doi: 10.1016/j.eswa.2023.122240.

[16] Z. Hong, W. Chen, H. Huang, S. Guo, and Z. Zheng, "Multi-hop cooperative computation offloading for industrial IoT-edge-cloud computing environments," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2759–2774, Dec. 2019.

[17] X. Dai, Z. Xiao, H. Jiang, M. Alazab, J. C. S. Lui, G. Min, S. Dustdar, and J. Liu, "Task offloading for cloud-assisted fog computing with dynamic service caching in enterprise management systems," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 662–672, Jan. 2023.

[18] J. Huo, F. T. S. Chan, C. K. M. Lee, J. O. Strandhagen, and B. Niu, "Smart control of the assembly process with a fuzzy control system in the context of Industry 4.0," *Adv. Eng. Inform.*, vol. 43, Jun. 2020, Art. no. 101031.

[19] K. Ding, J. Lei, F. T. S. Chan, J. Hui, F. Zhang, and Y. Wang, "Hidden Markov model-based autonomous manufacturing task orchestration in smart shop floors," *Robot. Comput. Integr. Manuf.*, vol. 61, May 2020, Art. no. 101845.

[20] Q. Liu, "Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system," *J. Manuf. Syst.*, vol. 58, pp. 52–64, Aug. 2021.

[21] C. Wang, X. Chen, J. Wu, and Z. Zhu, "Graphical deduction model based production instruction service system for smart job shops in industry 4.0," *Proc. CIRP*, vol. 83, pp. 739–742, 2019.

[22] J.-L. Liu, L.-C. Wang, and P.-C. Chu, "Development of a cloud-based advanced planning and scheduling system for automotive parts manufacturing industry," *Proc. Manuf.*, vol. 38, pp. 1532–1539, 2019.

[23] R. Sekhar, N. Solke, and P. Shah, "Lean manufacturing soft sensors for automotive industries," *Appl. Syst. Innov.*, vol. 6, no. 1, 2023, doi: 10.3390/asi6010022.

- [24] D. Mourtzis, E. Zervas, N. Boli, and P. Pittaro, "A cloud-based resource planning tool for the production and installation of industrial product service systems (IPSS)," *Int. J. Adv. Manuf. Technol.*, vol. 106, nos. 11–12, pp. 4945–4963, 2020.
- [25] X. Zhang, X. Ming, and Y. Qu, "Top-level scenario planning and overall framework of smart manufacturing implementation system (SMIS) for enterprise," *Int. J. Adv. Manuf. Technol.*, vol. 104, nos. 9–12, pp. 3835–3848, 2019.
- [26] Z. Xu, J. Wang, F. Hu, G. Abbas, E. Touti, M. Albekairi, and O. I. El-Hamrawy, "Improved camouflaged detection in the large-scale images and videos with minimum boundary contrast in detection technique," *Expert Syst. Appl.*, vol. 249, 2024, Art. no. 123558, doi: [10.1016/j.eswa.2024.123558](https://doi.org/10.1016/j.eswa.2024.123558).
- [27] S. Dehnavi, H. R. Faragardi, M. Kargahi, and T. Fahringer, "A reliability-aware resource provisioning scheme for real-time industrial applications in a fog-integrated smart factory," *Microprocess. Microsyst.*, vol. 70, pp. 1–14, Oct. 2019.
- [28] M. H. Raza, Y. M. Rind, I. Javed, M. Zubair, M. Q. Mehmood, and Y. Massoud, "Smart meters for smart energy: A review of business intelligence applications," *IEEE Access*, vol. 11, pp. 120001–120022, 2023.

**OMAR ALRUWAILI** received the Ph.D. degree in computer engineering from Florida Institute of Technology. He is currently an Assistant Professor and the Chair of the Computer Engineering and Networks Department, College of Computer Science and Information, Jouf University. He is actively teaching and research to publishing papers in his filed-on computer engineering and networks, wireless sensor networks, and the Internet of Things.

**FAN WU** was born in Jinhua, Zhejiang, China, in 1978. He received the M.S. degree in electronics and communication from Zhejiang University of Technology, in 2008. Since 2017, he has been an Associate Professor with the School of Information Technology, Zhejiang Shuren University. He has published three articles in SCI/EI and other journals, obtained three invention patents and more than 20 utility model patents, and more than 30 software copyrights. He has long been engaged in deep learning, 3-D reconstruction, digital image processing, and other fields of research.

**WAEEL MOBARAK** was born in Egypt, in 1979. He received the Ph.D. degree in applied engineering mathematics from the Faculty of Engineering, Alexandria University, Alexandria, Egypt, in 2013. He joined Alexandria University again as an Assistant Professor, in 2015. Currently, he is delegated as an Assistant Professor with the College of Engineering, University of Business and Technology, Jeddah, Saudi Arabia. His research interests include dynamics, mathematical modeling, and engineering management.

**AMMAR ARMGHAN** (Senior Member, IEEE) received the bachelor's degree from COMSATS University, in 2006, the M.S. degree in electronics and communication engineering from the University of Nottingham, in 2010, and the Ph.D. degree from Wuhan National Laboratory of Optoelectronic, Huazhong University of Science and Technology, Wuhan, China. He is currently an Associate Professor of electrical engineering with Jouf University. He has published more than 100 high-impact factor articles in the last three years. His research interests include machine learning, complementary metamaterial-based microwave and terahertz devices, biosensors, optics, and photonics. He is a reviewer of several reputed journals.

...