scientific reports

OPEN

Peer-driven task scheduling and resource allocation for enhanced performance in industrial IoT systems

Ayman Alfahid[1], Chahira Lhioui[2], Somia Asklany[3]✉, Rim Hamdaoui[4], Monia Hamdi[5], Ghulam Abbas[6]✉, Amr Yousef[7,8] & Anis Sahbani[9]

Peer-to-Peer (P2P) systems in the smart industry, enhanced by the Industrial Internet of Things (IIoT), provide a robust framework for efficiently managing and processing distributed tasks. However, resource sharing and allocation in these systems, often conducted one-to-one, can lead to task stagnancy when multiple resources are required in sequence. This paper proposes a novel Peer-dependent Scheduling and Allocation Scheme (PSAS) that leverages predictive learning to optimize task scheduling and resource allocation to address this limitation. The scheme evaluates resource availability, task length, and deadlines to minimize stagnancy and maximize system throughput. Predictive learning in PSAS enhances decision-making by analyzing historical resource utilization and providing real-time recommendations for resource allocation. This approach ensures scalability, reduces delays in task completion, and enhances overall system reliability, marking a significant advancement in P2P-based IIoT systems. Key performance metrics, including task processing ratio, processing time, stagnancy factor, and wait time, demonstrate that PSAS outperforms existing methods by improving processing ratio by up to 10.62% and reducing stagnancy by 5.06%.

Keywords Industrial internet of things, Peer-to-peer, Resource allocation, Smart industry, Task scheduling, Peer-dependent scheduling, Allocation scheme

Abbreviations

n	Number of tasks in the system
m	Number of available resources (machines)
$T = \{T1, T2, ..., Tn\}$	Set of tasks to be scheduled
$R = \{R1, R2, ..., Rm\}$	Set of available resources
$L(T_k)$	Length of task (execution time or workload size)
$D(T_k)$	Density of task (computational intensity)
S	Task schedule generated by PSAS
Q	Queue of pending tasks awaiting execution
$W(T_k)$	Waiting time of task in the queue
S_{final}	Optimized task schedule after allocation
θ_w	Waiting time threshold for task scheduling

[1]Department of Information Systems, College of Computer and Information Sciences, Majmaah University, 11952 Majmaah, Saudi Arabia. [2]Department of Computer Science and Artificial Intelligence, College of Computing and Information Technology, University of Bisha, 67714 Bisha, Saudi Arabia. [3]Department of Computers and Information Technologies, College of Sciences and Arts Turaif, Northern Border University, Arar 91431, Saudi Arabia. [4]Department of Computer Science, College of Science and Human Studies-Dawadmi, Shaqra University, 11911 Shaqra, Riyadh, Saudi Arabia. [5]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, 11671 Riyadh, Saudi Arabia. [6]School of Electrical Engineering, Southeast University, Nanjing 210096, China. [7]Electrical Engineering Department, University of Business and Technology, Ar Rawdah, 23435 Jeddah, Saudi Arabia. [8]Engineering Mathematics Department, Alexandria University, Lotfy El-Sied St. Off Gamal Abd El-Naser, Alexandria 11432, Egypt. [9]Institute for Intelligent Systems and Robotics (ISIR), CNRS, Sorbonne University, 75006 Paris, France. ✉email: Somia.Asklany@nbu.edu.sa; lashariabbas@seu.edu.cn

θ_q	Queue capacity threshold
ρ	Resource utilization factor
$P(T_k)$	Prediction function for task allocation
$St(T_k)$	Stagnancy factor of task Tk (delay in execution)
$A(T_k)$	Acceptance ratio of task Tk (successful execution without delay)
μ	Mean execution time for task scheduling
x_g	Scheduling gain factor based on predictive learning
o_n	Optimization factor for resource allocation
ab	Available state of a resource
qt	Queued time for a task in the waiting list
vi	Verification indicator for resource availability
r'	Reallocation factor when a task is reassigned to another resource
$C(T_k)$	Computation time required for task Tk
Δt	Time variation for real-time scheduling adjustments

Background

The industrial Internet of Things (IIoT) transforms the upcoming industrial systems by sensing the devices and actuators with computing capabilities and networking. The energy demand for Industrial IoT applications is increasing[1]. Peer-to-Peer (P2P) connections between Industrial nodes can satisfy the energy demand and improve energy efficiency by decreasing the loss while transferring the information to improve green industrial IoT[2]. Industrial information is maintained in a peer-to-peer network through blockchain features such as nonrepudiation and non-tampering. Truly distributed peer-to-peer systems tolerate node failure in the blockchain[3]. A scalable and trust-less P2P model can overcome the lack of trust between the systems; this will distribute information securely and operate transparently. Interaction between systems in the blockchain is carried out with a pair of public and private keys[4]. Systems use their Private Key to sign the transaction and transmit in one-hop peers. P2P in a smart building with wind generators and solar panels form a microgrid that will harvest energy produced and trade energy between systems[5].

IoT approach is applied in various applications like smart cities, home automation, industrial applications, etc. Real-time data collected from industrial IoT provides huge profits for industries[6]. Traffic aware scheduling algorithm in Industrial IoT minimizes the time for completing the task, end to end delay, and improves network lifetime, etc.[7] Centralized task manager will perform scheduling by combining routing information with resources needed for communication. Time-domain scheduling will operate tasks in a TDMA (Time Division Multiple Access) manners. In the MAC layer, centralized and decentralized task solutions are included in the state of art scheduling algorithm[8]. Optimal scheduling is done by reducing the network duty cycle. 6TiSCH scheduling provides high reliability for industrial controlling and monitoring applications enabled by wireless communication, and radio resources are efficiently allocated for the wireless devices. Data collected for the same task to perform in various industries is scheduled by high priority to low priority applications[9].

A large number of industrial IoT nodes are automated by an altered branch in industrial IoT applications. The data are sent to various time slots because industrial IoT nodes have multiple functions to perform[10]. For service priorities and QoS guarantees, communication resource allocation problems are considered in industrial IoT. According to the industrial IoT node priorities, the allocation of resources has taken place[11]. A deep Q network algorithm optimizes time slot and channel allocation models to ensure that high-priority nodes access the network. Resource allocation is divided into centralized and decentralized resource allocation. In centralized resource allocation, maintaining and managing scheduling problems is done by the gateway management node[12]. Increasing the network throughput ensures the transmission of high priority data by an adaptive resource allocation method that adapts deep Q network learning. In device communication in industrial IoT, resource allocation is carried out by a matching approach and many-to-one matching of peer effects[13]. An industrial cyber-physical IoT system (CPIoTS) considers resource allocation problems to maximize energy efficacy and minimize the transmit rate through the QoS requirement of sensors and actuators[14].

In IIoT peer-to-peer (P2P) contexts, task stagnation, unused resources, and unpredictable workload variations make standard task scheduling and resource allocation ineffective. Because they can't adapt to real-time work needs and system restrictions, centralized scheduling and static resource allocation slow processing and lower system performance. Lack of predictive learning-based adaptive scheduling systems worsens resource allocation and stagnation, affecting reliability and scalability. The study introduces the Peer-dependent Scheduling and Allocation Scheme for IIoT work and resource planning. PSAS eliminates stagnancy using predictive learning to maximize system performance.

IIoT faces task scheduling and resource allocation challenges, especially in Peer-to-Peer (P2P) systems. Current approaches often fail to account for the dynamic nature of resource availability, task dependencies, and computational demands, leading to prolonged delays and underutilization of critical resources[15]. Current methods, such as centralized task management and machine learning-based optimization techniques, have limitations, such as scalability bottlenecks and single points of failure. Advancements in distributed scheduling, such as fuzzy-based priority scheduling and load quantization approaches, partially address resource allocation issues but fail to handle real-time task dependencies[16].

Problem statement

A critical gap exists in developing adaptive, predictive learning-based scheduling techniques that dynamically optimize resource allocation while ensuring low stagnancy and high system throughput. Addressing these challenges is essential for achieving higher operational efficiency, reducing downtime, and ensuring reliability in dynamic industrial environments.

Research movation and its novelty

The motivation behind this research is to integrate IoT devices' into industrial systems makes management and resource allocation more complicated. Time-sensitive processes must be optimized for improving IIoT performance and reliability. Alongwith this the consideration of security and interoperability challenges in this environments are also need to be analyzed. The innovative idea in this research is that the use of IIoT-specific independent network-slicing topologies learning for performing adaptive resource allocation. Thereby proposing an interoperable access architecture and real-time scheduling algorithms for industrial contexts shows novel research idea that motivates this research and improves IIoT system functionality and efficiency.

Due to these issues, the Peer-dependent Scheduling and Allocation Scheme (PSAS) is developed to fix existing scheduling methods. PSAS was created because a predictive learning model that can leverage prior resource use patterns to forecast job delays and distribute resources is needed. PSAS dynamically optimizes work scheduling using predictive learning to boost productivity and reduce stagnation. The second motivation is creating a distributed work scheduling method for big IIoT deployments. Unlike centralized scheduling, PSAS dynamically loads and redistributes workloads amongst peer nodes, ensuring load balance and resource efficiency. Work scheduling in complex industrial contexts becomes more flexible.

Key contributions

The main contributions of our study are:

- A predictive task scheduling mechanism is developed that evaluates resource availability and task dependencies, with predictive learning leveraged to reduce task stagnancy by 5.06% compared to existing methods.
- A dynamic resource allocation framework is to be created that adapts to real-time workload variations, with processing time improved by 16.38% over baseline methods.
- The Peer-dependent Scheduling and Allocation Scheme (PSAS) is to be evaluated using key metrics such as task processing ratio, wait time, and resource utilization, with a 10.29% improvement in resource efficiency demonstrated.
- Scalability is to be ensured with a lightweight and scalable framework suitable for large-scale IIoT deployments, with reliability ensured under high task density scenarios.

The proposed PSAS solves this research challenge called job stagnancy in P2P-based IIoT systems by employing dynamic scheduling algorithms that can prioritize resource distribution depending on peer capabilities and network conditions, thus improving responsiveness and reducing delays. IoT devices' integration into industrial systems makes management and resource allocation more challenging, and time-sensitive processes are optimized to improve IIoT performance and reliability. Security and interoperability issues in varied environments are analyzed. Thus the research idea of using an interoperable access architecture and real-time scheduling algorithms for industrial contexts shows novel methods overcomes the existing research challenges and improves IIoT system functionality and efficiency.

The rest of the research article is arranged in the following manner: Section "Related works" discusses the recent work on the proposed topic. Section "Problem formulation" discusses the problem Formulation, Section "Proposed peer-dependent scheduling and allocation scheme" gives a details of the Proposed Peer-dependent Scheduling and Allocation Schem. Section "Result and discussion" discusses the related hyperparameters for this network model and comparison analysis with existing models. Section "Conclusion" discusses conclusion and future research direction.

Related works

Xia et al.[17] proposed real-time scheduling for the IIoT. The performance of RDF (relative execution assited deadline first model) performs better than FP (fixed priority model) and EDF (earlier deadline first model). Performance reliability are improved by system schedulability with the help of MC (mixed-criticality) based RDF models.

A quantized approach to load scheduling is recommended by Bhatia et al.[18] in a fog computing environment for IoT applications. The node computing initialization is used to estimate the fog node computation capacity. The QCI-Neural network model is used to handle real-time heterogeneous tasks. The superiority of the proposed work is analyzed and compared with the state-of-the-art scheduling models.

Farhan et al.[19] presented that the fast growth communication technologies and energy networks has led to the Internet of Energy (IoE). Digital control devices send data via IT systems. The energy industry has grown, attaining the IoE milestone and developing the next generation. This study examines the integration of smart communication infrastructure, metering, and energy maintenance methods. Communication architecture, supply and demand, and protocol are examined. The study addresses integration, security, and energy management to optimize IoE technology.

Niragire and Kwena[20] examined that project risk management affects community assisted programs, specifically the Peer-Driven Change Project. Risk identification, analysis, control, and contingency planning were assessed. Project success was strongly correlated with risk control, identification, and contingency planning. Research also identified a lesser positive association between risk analysis and project success. The study proposed risk prioritization improvements but concluded that strong risk management methods are essential for project success. Standardized risk management frameworks, staff training, and real-time risk monitoring are recommended.

Yung et al.[21] investigated the potential of value engineering to aid in selecting and evaluating P2P lending platforms in Indonesia, a developing nation where conventional banking services are scarce. According to the research, low-value index platforms have more nonperforming loans (NPLs) and fewer favorable terms for stakeholders. In contrast, high-value index platforms provide competitive interest rates, lower fees, and better risk management. The results highlight the significance of a value engineering strategy in improving the sustainability of Indonesia's P2P lending ecosystem and optimizing platform selection, offering practical advice for platforms, regulators, and consumers involved in peer-to-peer lending.

Padmaja et al.[22] examined the effect of performance management on contentment among IT workers at a company located in Hyderabad. Ordinary Least Squares Regression Analysis and ANOVA tests are among the quantitative methods used in the research. Clarity of performance objectives, comprehension of career development goals, quality of feedback, recognition and awards, and overall employee satisfaction are all positively associated, demonstrating the substantial impact of performance management on employee satisfaction. Personalized career development plans, cross-departmental cooperation, open feedback forums, strategic performance workshops, recognition dashboards in HRMS portals, and recognition programs led by peers are some of the suggestions for process improvements made in the study.

Vashishth et al.[23] investigated the potential for industrial robotics systems to include blockchain technology and artificial intelligence (AI). Allocation of resources can be done dynamically according to data and system needs in real-time using AI-based methodologies. Blockchain technology ensures transparency and trust by documenting and validating resource transactions on a decentralized, secure network. Machine learning, evolutionary algorithms, and reinforcement learning are some artificial intelligence algorithms covered in this chapter. They are used in industrial robots for tasks such as optimizing resource allocation. Additionally, it delves into how blockchain technology, which offers a distributed ledger for recording and verifying transactions, can improve resource allocation and optimization.

Qin et al.[24] proposed a Mobility-aware Computation Offloading and Task Migration approach (MCOTM) to solve the limits of mobility-aware devices. The trajectory and resource prediction method reduces job turnaround time and energy usage. Lagrange interpolation equations determine mobile device trajectory and LSTM tracks IIoT time-varying resource parameters. The predictions help Deep Deterministic Policy Gradient (DDPG) make task relocation, and resource allocation decisions. According to experiments, the suggested MCOTM mitigates execution time by 42% and energy usage by 10% while maintaining a migration ratio of 50%.

Jie et al.[25] proposed game-theoretic resource allocation for an Industrial IoT environment based on the fog. The fog node provides the service to data users by providing resources from the cloud center. The resource allocation problem is modeled to maximize resource utilization and achieve Stackelberg equilibrium, and three algorithms are developed for the Nash equilibrium. The performance of the proposed method is evaluated and compared with computing schemes.

By utilizing an Actor-Critic Deep Reinforcement Learning (DRL) framework, Chen et al.[26] devise a fresh method for optimizing cloud resource management. This solution tackles the intricate and ever-changing task of allocating cloud resources by facilitating adaptive decision-making that improves energy efficiency and establishes optimal load balancing.

Zhang et al.[27] suggest collaboratively offloading diverse tasks to optimize satellite edge computing energy utilization. This method solves satellite network issues including low bandwidth and excessive latency by intelligently distributing processing workloads between satellites and edge devices. The recommended technique boosts system efficiency and satellite component longevity. If included in the related work section, this research would illuminate the importance of adaptable frameworks like PSAS in many computing environments and provide a broader view of energy-efficient task scheduling methods.

For massive edge computing systems, Zhang et al.[28] presented a scheduling method driven by AI that maximizes energy efficiency. This approach differs from PSAS in that it prioritizes energy consumption above task stagnation and scheduling dynamics in real-time.

Adaptive task migration algorithms were investigated in distributed fog computing systems by Chen et al.[29], with an emphasis on node workload balance. PSAS improves upon this idea by incorporating predictive learning into real-time task scheduling, which helps with stagnation problems. For IIoT systems enabled by 5G, Chen et al.[30] suggested hierarchical scheduling models with a focus on minimizing network latency. PSAS, on the other hand, is better suited to extremely dynamic industrial contexts since it addresses resource usage and scheduling stagnation.

Table 1 summarizes the research gap analyzed from the existing research works.

Real-time scheduling utilizing the Mixed-Criticality Relative Execution Deadline (MCRD) algorithm is one way researchers have improved Industrial IoT reliability. It also optimized node computing capacity with a fog-based IoT load scheduling QCI-Neural Network model. Internet of Energy (IoE) effects on smart infrastructure, energy management, and security were also examined, and risk management in community-based projects, value engineering in sustainable platform selection, and performance management's favourable impact on employee happiness were also stressed.

Task stagnation, poor resource usage, and dynamic workload variations make task scheduling and allocation in IIoT contexts, especially P2P designs, challenging. Current centralized and static scheduling methods can't handle real-time task interdependence. This generates bottlenecks, underutilization, and scalability issues. Without predictive learning methods, reactive scheduling allocates work without considering prior trends, causing delays and longer wait times. Traditional approaches' inflexibility causes unexpected technological failures, communication delays, and job priorities-changing disturbances. Another concern is that many scheduling methods are centrally controlled. This reduces their flexibility and creates failure points in large-scale installations. A decentralized, adaptive, and predictive system must optimize throughput and eliminate processing delays to manage resource constraints, changing workloads, and real-time uncertainty.

Inefficient task management, scalability difficulties, and lack of flexibility define existing IIoT work scheduling and resource allocation methods. Many task execution techniques use static scheduling frameworks, which cause idleness and resource underutilization. Most methods also use one-pass allocation techniques, which fail

Author(s)	Algorithm used	Contribution	Results achieved	Limitations
Xia et al.[17]	RDF, FP, EDF, MCRD	Proposed real-time scheduling with heterogeneous routing for IIoT	Improved reliability, real-time performance, schedulability	Limited scalability; lacks adaptability to fluctuating workloads
Bhatia et al.[18]	QCI-Neural network model	Quantized load scheduling in fog computing for IoT applications	Enhanced task handling, computation estimation	Performance degrades under high load; lacks predictive allocation
Farhan et al.[19]	IoE framework	Explored IoE integration with ICT for smart energy networks	Improved metering, pricing, energy management	Focus on energy networks; lacks IIoT scheduling focus
Niragire and Kwena[20]	Risk Management Framework	Analyzed project risk management in community-based programs	Strong correlation with project success metrics	Not applicable to IIoT scheduling; lacks technical resource focus
Yung et al.[21]	Value Engineering Strategy	Evaluated P2P lending platforms using value engineering	Improved sustainability, platform selection strategies	Limited relevance to IIoT; lacks dynamic resource scheduling
Padmaja et al.[22]	OLS Regression, ANOVA	Studied performance management effects on IT employee satisfaction	Positive correlation with satisfaction metrics	Focus on HR metrics; no direct IIoT application
Vashishth et al.[23]	AI Algorithms, Blockchain Integration	Investigated AI and blockchain for resource optimization in industrial robotics	Enhanced transparency, dynamic resource allocation	High overhead with blockchain; limited real-time scalability
Qin et al.[24]	MCOTM, Lagrange Interpolation, LSTM, DDPG	Mobility-aware computation offloading and task migration in IIoT	Reduced turnaround time (42%), energy usage (10%)	High migration rate (50%); complex in resource-constrained environments
Jie et al.[25]	Game-Theoretic Models, Stackelberg & Nash Equilibrium	Resource allocation in fog-based IIoT environments	Optimized utilization, improved scheduling equilibrium	Affected by dynamic workloads; lacks predictive allocation models

Table 1. Research gap analysis of related works.

Component	Mathematical formulation	Description
Decision variables	$x_{ij} \in \{0{,}1\}$ $y_{ij} \in \{0{,}1\}$ $C_i \in \mathbb{Z}^+$	$x_{ij} = 1$ if task T_i is assigned to processor P_j $y_{ij} = 1\text{if task } T_j$ depends on task T_i
Objective function	$min \sum_{i=1}^{n} C_i \; or \min max_i C_i$	Minimize total completion time or makespan
Constraint 1: task assignment	$\sum_{j=1}^{m} x_{ij} = 1 \forall_i \epsilon \{1, \ldots, n\}$	Each task is assigned to exactly one processor
Constraint 2: peer dependency	$C_i \geq C_i + t_i - M\,(1 - y_{ij}) \, \forall (i, j) \in D$	Task T_j must execute after dependent task T_i; M is a large constant
Constraint 3: resource limitation	$\sum_{i=1}^{n} r_i x_{ij} \leq R_j \forall_j \epsilon \{1, \ldots, m\}$	Total resource usage must not exceed capacity of each processor
Constraint 4: execution time bounds	$C_i \geq t_i \forall_i$	Each task's completion time must be at least its execution time
Constraint 5: binary nature	$x_{ij}, y_{ij} \in \{0{,}1\}$	Binary decisions for assignment and dependencies

Table 2. ILP-based problem statement for peer-dependent scheduling and allocation scheme.

to dynamically reallocate delayed or stopped work since they assign tasks once without monitoring. Scalability limitations and single points of failure make centralized scheduling unsuitable for large-scale, distributed IIoT systems. Current optimizations based on machine learning favor scheduling gains over long-term predictive resource allocation, limiting their ability to proactively resolve delays, defects, and workload fluctuations. A decentralized, predictive, and flexible IIoT scheduling mechanism is needed to overcome these limits. This technique should balance loads, manage jobs proactively, and modify resources in real time.

Problem formulation

The Peer-dependent Scheduling and Allocation Problem is a scheduling process that optimizes performance by allocating tasks with peer dependencies across processors. The formulation identifies decision variables, objective function, and constraints, providing a theoretical foundation for the proposed algorithmic solution and ensuring clarity in the problem structure.

Table 2 outlines the Integer Linear Programming (ILP) formulation of the Peer-dependent Scheduling and Allocation Problem. The objective function aims to minimize the overall completion time (makespan) across all tasks. Constraints enforce valid task assignment, respect peer dependencies, limit resource usage per processor, and ensure correct timing of task executions. Binary variables represent assignment and dependency decisions. This formal model satisfies the request for a structured problem definition using ILP and serves as a foundation for evaluating the efficiency and correctness of the proposed solution.

Proposed peer-dependent scheduling and allocation scheme

IIoT, using a peer-to-peer system, identifies the task and improves the processing rate reliably by performing a schedule. The PSAS method was developed in this work to decrease the stagnancy and waiting time of the smart industry. In Fig. 1, the PSAS process flow in IIoT is presented.

It focuses on predictive learning that deploys task scheduling and resource allocation. The task's length is identified using predictive learning, and the stagnate task is detected and overcome using the PSAS approach.

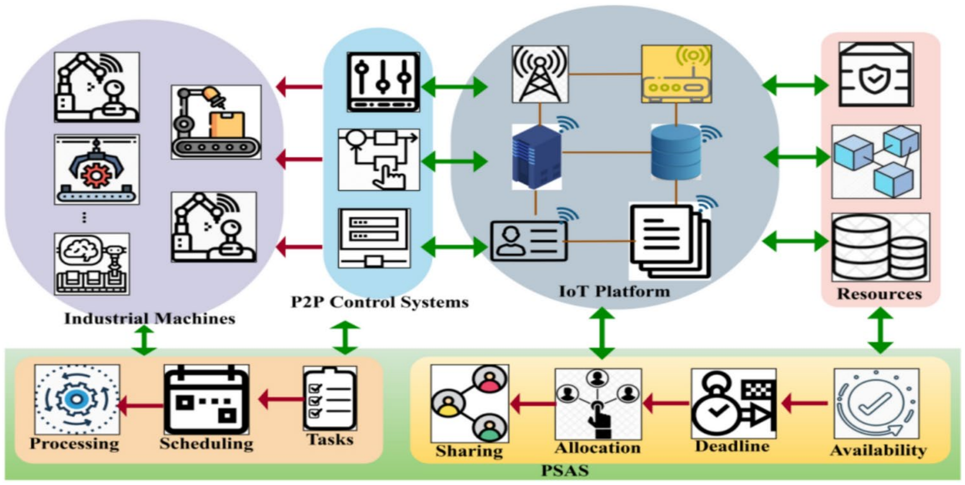

Fig. 1. PSAS process flow.

In a smart industry, the resources are detected by monitoring either the busy or ideal state, formulated in the following equation.

$$\rho = \left(\frac{\mu/o_0}{t_k}\right) * \prod_{x_g}^{a_b} \left[s_0\left(\vartheta\right) + d_a\left(c_e\right) - \left(q_t + e_i\right)\right] * \sum_{o_n} x_g - t_k \tag{1}$$

In the above Eq. (1), monitoring is performed for the varying resources in the smart industry to identify whether the resources are busy or ideal. Here, the resources are defined as the machines in the IIoT environment periodic monitoring is done to allocate the task to the resources. The resources are represented as o_0, the number of resources is denoted as o_n, the task is termed as t_k, the busy and ideal is referred to as s_0 and d_a. The monitoring is denoted as ρ, which deploys the preceding state of the task assigned to resources, and its computational levels are analyzed. The prior task is termed as c_e; it defines the queued process for the resources; in this case, the additional task is not allocated to the particular resources.

It is carried out by defining the length of the task, i.e., how long the assigned task runs in the application; by utilizing this, the pursuing functions are allocated desirably. The pursuing task is denoted as u'; the queued process is represented as q_t and the length of the task is referred to as x_g. In this process, scheduling is done for the varying resources by deploying the task's length and computing time. The scheduling is termed as ϑ, the detection is referred to as μ, and the allocation of the task is denoted as a_b. The waiting time e_i is also calculated to address the processing time for the P2P systems in the smart industry. The following sections analyze the P2P system interaction by defining task scheduling and resource allocation.

Optimization techniques

The proposed hybrid approach effectively integrates genetic algorithms (GA) and enhanced particle swarm optimization (PSO) to address the complexities of scheduling in multi-machine systems.

In Fig. 2a, A hybrid technique using genetic algorithms (GA) and enhanced particle swarm optimization (PSO) addresses scheduling issues. GA simulates natural selection and crossover mechanisms to develop diverse, high-quality starting solutions. Enhanced PSO optimizes GA schedules by adjusting parameters to reduce task delays and machine idle time. Resources are efficiently allocated, and workload is balanced across machines to decrease task delays and idle time.

In multi-machine scheduling contexts, GA's exploratory capabilities and PSO's exploitative efficiency enable scalability and adaptability. Optimizing scheduling in dynamic, resource-intensive situations like smart manufacturing improves IIoT system throughput and resource utilization, meeting research aims. By overcoming method limitations, GA and PSO improve scheduling in complicated, multi-machine systems.

Task scheduling

It is a set of protocols to be followed while assigning the task to the system that deploys the previous working state by performing a prediction process. Here, it verifies whether the task is assigned to suitable resources. The 'if and otherwise' condition is used in the following equation. It is associated with the state of the smart industry system, and the computation is evaluated from the IoT environment. There are two types of schedulers: long-term and short-term scheduling that deploy the resources state of action in IIoT.

$$\nabla = \begin{cases} 1, & if \quad \rho + \left(\left(t_k * a_b\right) + \frac{s_0 - d_a}{\prod_{x_g + y'} o_0}\right) \\ 0, & Otherwise \end{cases} \tag{2a}$$

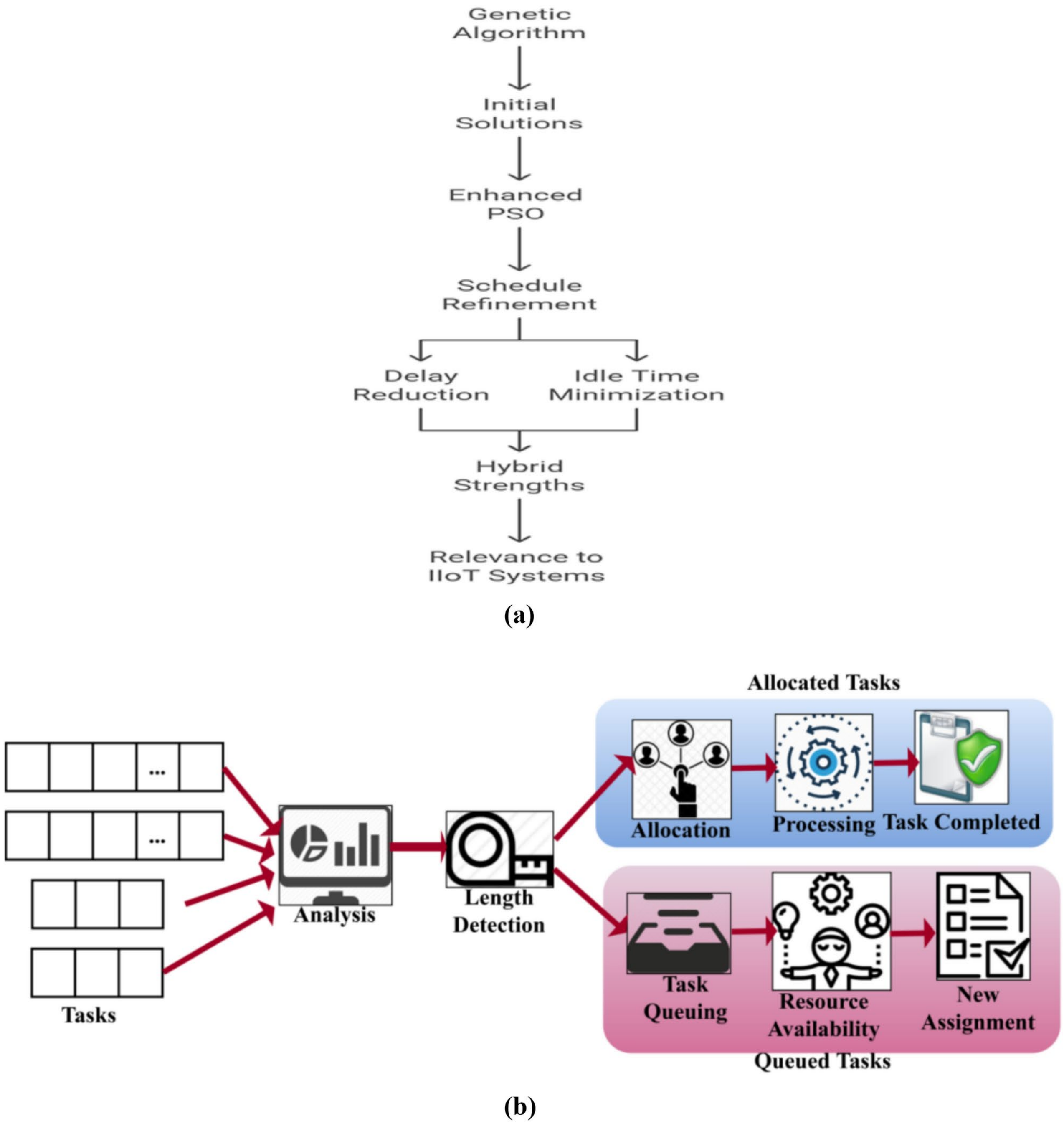

Fig. 2. (**a**) Optimization techniques. (**b**) Task scheduling process.

In Eq. (2a), The analysis is carried out by evaluating the 'if and otherwise' condition. If the condition is satisfied, then the tasks are allocated; else, the condition is computed. The analysis is termed periodic monitoring, performed for the varying task allocation. Here, the initial stage is to identify the system's state as either busy or ideal; by deploying this, the task is reliably assigned to the system. The task's length and density are calculated to schedule the resources. Here, it monitors the computation time and provides the resultant.

The busy and ideal state is monitored periodically and allocated the task to suitable resources, and it is represented as $\frac{s_0-d_a}{\prod_{x_g+y'} o_0}$, the density of the task is denoted as y'. The density is defined as how long the task withstands in the smart industry and is measured, which is used to allocate the upcoming job by deploying the density. For every instance, the length and density are calculated periodically; the scheduling is provided efficiently from that. In Eq. (2a), if the condition runs by evaluating the task's length and density, it is identified whether it is busy or ideal. From that, the processing is carried out; otherwise, the condition is executed. If the otherwise condition is executed, the processing ratio decreases, and predictive learning is introduced to improve this.

Task scheduling-predictive learning

Task scheduling is performed for the varying resources that deploy the task assigned to the system and monitor the processing. The scheduling is carried out using the predictive learning method associated with the previous state of resources, and the result is provided. Thus, predictive learning uses the knowledge that affects performing the scheduling in a planned manner in the IIoT environment. The task scheduling process is illustrated in Fig. 2b.

It is used to predict the pursuing task allocated with the history of resources. For this, it uses some knowledge-based retrieval. The following equation is used to identify the task with predefined knowledge.

$$h_0 = \left(\frac{s_0 + t_k}{\sum_{e_i} \rho}\right) * \left(\mu + \frac{x_g}{o_n}\right) + ((a_b * t_k) - v_i + c_e) + \beta - u' \tag{2b}$$

In Eq. (2b), The task scheduling is done by utilizing the prediction model associated with the knowledge of the preceding state of the resources. The detection is carried out for the smart industry's resources. It calculates the length of the task, and it is denoted as $\left(\mu + \frac{x_g}{o_n}\right)$. Here, the state of the system is identified, and it is denoted as h_0, if it is busy, the task is not allocated to the system. Here, the waiting time is calculated for every task assignment, and it is represented as $\left(\frac{s_0+t_k}{\sum_{e_i} \rho}\right)$.

Predictive learning is used to map the preceding task's working state with the upcoming task allocated to the system; here, it measures the length and density. The allocation of the task is evaluated the available state of the resources, and it is termed as v_i, and map with the preceding task working, and it is denoted as $((a_b * t_k) - v_i + c_e)$. The prediction is referred to as β, to define the pursuing state of the system reliably; by deploying this, task scheduling is done. The length and density of the task vary for every assignment. The following equation is used to calculate this.

$$h_0\left(t_k\right) = \left(\frac{1}{o_n}\right) * \prod_{v_i}^{n_0} (\rho + q_t) * [(s_0 - d_a) + (a_b\left(d_a\right)/\beta)] - (\nabla - k_e) \tag{3}$$

In Eq. (3), task length and density is defined by relating it with the preceding state of processing that deploys predictive learning. Task scheduling is estimated for varying resources, periodic monitoring is carried out, and the system's available and unavailable state is identified. The unavailable is defined as n_0, by deploying this task is provided to the ideal system, and it is represented as $[(s_0 - d_a) + (a_b\left(d_a\right)/\beta)]$. The prediction is evaluated to allocate the task to the resources. It also monitors the working state of the system on time, and it is termed as k_e that shows better scheduling. The length and density are identified by evaluating the task allocation's predefined knowledge; by utilizing this, the tasks are allocated desirably.

If the length and density are defined, then the scheduling is estimated reliably, and in this, queuing is monitored; if the process is in the queue, it waits for the ideal state. By processing this, the system maintains a busy state, no processes are queued, and a task performs better in the IIoT. Thus, the task's length and density are identified by equating the above equation to perform the scheduling. The verification is used to share the task with the other ideal resources. An availability and unavailability check is performed and formulated as below.

$$\partial = \prod_{o_n}^{\rho} v_i\left(o_0\right) + t_k\left(d_a\right) + a_b - c_e + \left(\frac{r' + o_0\left(d_a\right)/\vartheta}{x_g + y'}\right) \tag{4a}$$

In Eq. (4a), The verification phase is done by evaluating the above Eq. (4a) to check the availability and unavailability of resources and provide the task accordingly. If the resource is available, the task is allocated to the desired system, and the computation is promptly monitored, which enhances the processing time. Thus, the task is allocated with the predefined knowledge to analyze the upcoming task. The length and density are calculated to assign the new task to the system for this previous history of processing is necessary to perform the scheduling, and it is denoted as $\left(\frac{r'+o_0(d_a)/\vartheta}{x_g+y'}\right)$.

The verification is ∂ used to analyze the availability and unavailability of the resources, and then the sharing r' is processed among the resources. Only the ideal systems are provided with the task; thus, the assignment is performed post this verification phase. In this scheduling method, predictive learning allocates the task to the smart industry's resources to decrease stagnancy. The following equation is used to map the preceding task working with the pursuing task and detects the stagnancy to address this methodology.

$$\delta = \sum_{o_n} (s_0 - d_a) + \nabla * \left(\frac{o_0 + v_i}{r'}\right) * \prod_{\vartheta} \left(k_e + \frac{e_i}{q_t}\right) + \rho - \left(x_g + y'\right) \tag{4b}$$

In Eq. (4b), the mapping is represented as δ; in this, it detects the busy and ideal state of the system by evaluating the predictive learning method. It is associated with the knowledge of the previous state of resources and provides the task to the available resources in IIoT. Thus, the mapping is estimated by acquiring the upcoming task associated with the prediction method. It results in the sharing of the task, and it is denoted as $\nabla * \left(\frac{o_0+v_i}{r'}\right)$. Figure 3 portrays the task mapping for the devices using predictive learning.

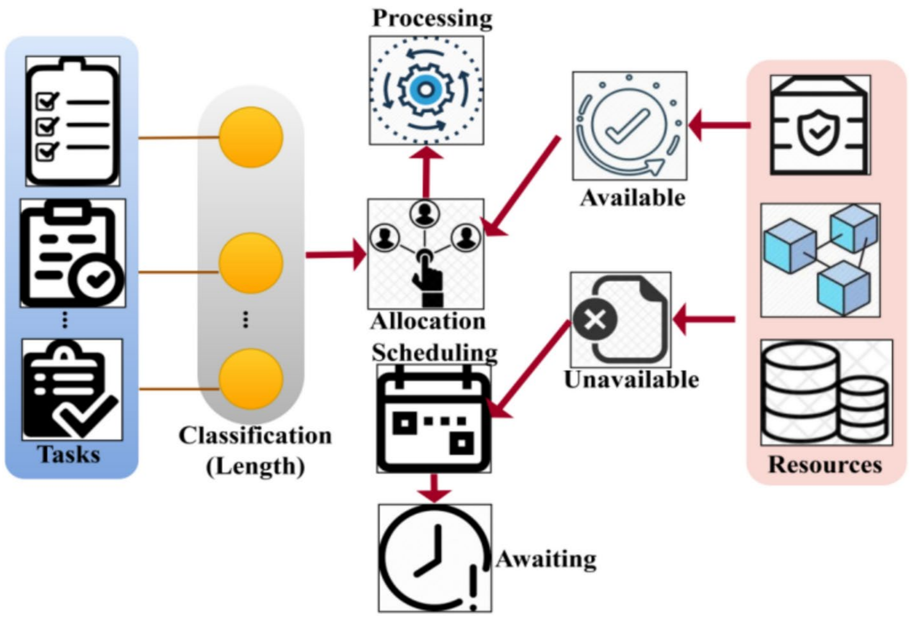

Fig. 3. Task mapping using predictive learning.

Mapping is performed by relating the preceding state of data with the pursuing state, resulting in reliable scheduling that increases the processing ratio. In this proposed work, the mapping is done for every instance to perform the efficient scheduling associated with the queued task. Here, the stagnancy is not properly identified. The following Eq. (5) is used to detect stagnancy and verify shared resources and mapping.

$$\delta\left(r'\right) = \left(t_k + \rho/q_t\right) * \sum_{k_e} \left(e_i + o_0\right) * \left(\frac{s_0 + d_a}{\vartheta}\right) + \left(h_0 - \beta\right) * a_b \left(d_a\right) \tag{5}$$

In Eq. (5), the task mapping to the ideal system is estimated by scheduling; this periodic monitoring of resources is done to analyze the availability. If the resources are available, the scheduled task is given to the system and monitors the working state that deploys to detect stagnancy. Stagnancy is defined as the task not being completed on time; to address this, resource sharing is performed for the available task that is ideal or not. Here, the queued process is monitored, and it is denoted as $(t_k + \rho/q_t)$.

In this equation, the identification is performed by utilizing the prediction model. This shares the ideal resources, and it is denoted as $(h_0 - \beta) * a_b\,(d_a)$. Thus, the stagnancy is detected by equating the above equation with periodic monitoring of resources in the IIoT environment. Task scheduling in predictive learning is used to share the resources with the ideal system and address the waiting time of the process that relies on the queue. It is formulated in the following equation.

$$\vartheta = \left(\frac{\rho + o_n}{s_0 - d_a}\right) * a_b + \prod_{\delta} \left(c_e - u'\right) * \left(t_k * o_0/m' * \nabla\right) + \sum r'\left(o_0 - a_b\right) - q_t \tag{6}$$

In Eq. (6), the scheduling is performed by deploying the predictive learning method associated with the system's busy or ideal state; this method uses periodic monitoring to identify stagnancy. If stagnancy is detected, the processing ratio is improved. For this analysis, the previous step of the task allocation process must be mapped. This mapping identifies the task allocation to the respective resources and monitors the stagnancy, and it is represented as m'.

The task schedule is evaluated by the availability of resources in the innovative industry, which provides resources for the ideal system. The task is allocated to the respective resources after the verification phase equated in Eq. (4a). Thus, the prediction process is carried out reliably. Thus, the task schedule is estimated, decreasing the queued process, leading to stagnancy and reducing the processing ratio. The resource allocation is evaluated after the task scheduling to provide the task to the necessary system; this predictive learning is used and discussed in the following section.

```
INPUT:
    Task List: T = [$t_1, t_2, \ldots, t_k$]
    Initial System State: $h_0$ (e.g., "busy", "idle")
    Resource List: R = [$r_1, r_2, \ldots, r_n$]
    Parameters: $\mu, x_g, o_n, \varrho, q_t, d_a, a_b, s_0, v_i, r'$
    Thresholds: Waiting Time Threshold ($\theta_w$), Queue Capacity ($\theta_q$)

OUTPUT:
    Final Task Schedule ($S_{final}$)
    Performance Metrics: Task Completion Time, Queue Length, Resource Utilization

BEGIN
1. Initialize system state : $h_0 \leftarrow$ "idle" and resources R.
2. FOR each task $t_k$ in T:
        a. Calculate task length: $L = \mu + (x_g / o_n)$
        b. Calculate task density: $D = \varrho + q_t$
3.     IF : $h_0$ == "busy":
            Skip task $t_k$ and CONTINUE.
4.     ELSE:
            Calculate waiting time ($W_t$) for task $t_k$.
            IF $W_t > \theta_w$:
                Add task $t_k$ to queue Q.
            ELSE:
                Predict task allocation using the equation:
                $h_0 = \left(\frac{s_0 + t_k}{\sum^{e_i} \rho}\right) * \left(\mu + \frac{x_g}{o_n}\right) + ((a_b * t_k) - v_i + c_e) + \beta - u'$    // Eq.3
            Allocate tk to resource $R_j$ in R.
6.     Monitor task queue Q periodically for resource availability.
7.     IF stagnancy detected:
            Adjust task allocation using:
            $\delta = \sum_{o_n}(s_0 - d_a) + \nabla * \left(\frac{o_0 + v_i}{r'}\right) * \prod_{\vartheta}\left(k_e + \frac{e_i}{q_t}\right) + \rho - \left(x_g + y'\right)$  //Eq.4b
8.     Update task assignment for timely processing.
9.     Verify task allocation using:
        $\delta(r_{\prime}) = \left(t_k + \rho / q_t\right) * \sum_{k_e}(e_i + o_0) * \left(\frac{s_0 + d_a}{\vartheta}\right) + (h_0 - \beta) * a_b(d_a)$  //Eq.5
10. Finalize task schedule $S_{final}$ based on resource utilization.
11. Output $S_{final}$ and associated performance metrics.

END
```

Pseudocode 1. Task scheduling with predictive learning

The pseudocode 1 describes a task scheduling framework that uses predictive learning to optimize resource utilization and task completion in dynamic systems. It uses inputs like task list and system state and parameters like task length, density, waiting time threshold, and queue capacity. The algorithm initializes the system, calculates task length and density, and checks for busyness. Tasks exceeding the threshold are queued, and resources are allocated based on predictive equations. The algorithm ensures timely processing, verifies allocations, and finalizes a task schedule.

Suppose an IIoT-enabled smart factory includes five machines (M1, M2, M3, M4, M5) that perform ten real-time tasks (T1–T10) with varied execution times and dependencies. T5 will be idle while M2 works on T3 since conventional scheduling assigns jobs sequentially. Instead, PSAS predicts job needs and reroutes T5 to M4, a machine with idle capacity, to reduce wait time and stagnation. PSAS dynamically reroutes task execution based on prior task completion times to ensure optimal task allocation and minimal delay, even when a job requires several resources (e.g., T7 needs M1 and M3 consecutively).

Resource allocation

Resource allocation is sharing the available resources with the ideal system and verifying whether it works properly or not, and it is monitored periodically. It manages the task, assigns them to the available resources, and calculates the resource's waiting time in the queue to decrease the processing ratio. In Fig. 4, the resource allocation process is illustrated.

The allocation of resources is used to promptly evaluate the number of tasks in the ideal system and estimate its length and density. The resources are analyzed by equating the following equation to share the task with the system.

$$\nabla\left(o_0\right)=\left(r'+t_k\right)*\left(\frac{q_t-m'}{\rho/k_e}\right)+\prod_{s_0}\left(t_k-h_0\right)*\left(\frac{c_e-u'}{\beta}\right)+\delta \tag{7a}$$

In Eq. (7a), a resource analysis is carried out to share the task with the desired system; periodic allocation monitoring addresses the stagnancy in this process. The queued tasks are analyzed to decrease the stagnancy, and it is denoted as $(r'+t_k)*\left(\frac{q_t-m'}{\rho/k_e}\right)$ Here, the length of the task is deployed. If the task length and density are calculated, task-sharing analysis is performed reliably, improving the processing ratio. Predictive learning allocates resources to the smart industry's varying resources that describe previous knowledge.

Resource allocation-predictive learning

In predictive learning, the resource allocation is done by planning the system's availability in the IIoT and assigns the task by detecting the queued task. If the queued task is addressed, the stagnancy is decreased in this processing, and the resource allocation is given to the ideal system that completes the task on time. The analysis of task completion is estimated by mapping with the previous state of action, and it describes the length and density of the task. The duration of the queue task leads to stagnancy. It is checked by equating the following equation that deploys the available resource relies upon or not by performing detection.

$$\mu=\left(o_0+r'\right)*\left(\frac{\nabla+m'}{a_b}\right)+\sum_{\delta}h_0\left(s_0\right)*\left(\frac{t_k/\partial}{e_i+m'}\right)+\left(x_g*y'\right)-k_e \tag{7b}$$

The detection is done equated in the above Eq. (7b). In this identification, resources are allocated, and the analysis is performed, and it is represented as $(o_0+r')*\left(\frac{\nabla+m'}{a_b}\right)$. The mapping is performed for the preceding and pursuing resources, and the task is allocated desirably by deploying predictive machine learning. The detection process is used to evaluate the resource availability and queued process that takes more time to complete the task that is denoted as $\left(\frac{t_k/\partial}{e_i+m'}\right)$. The following equation evaluates a task's identification and computation time to reduce processing time.

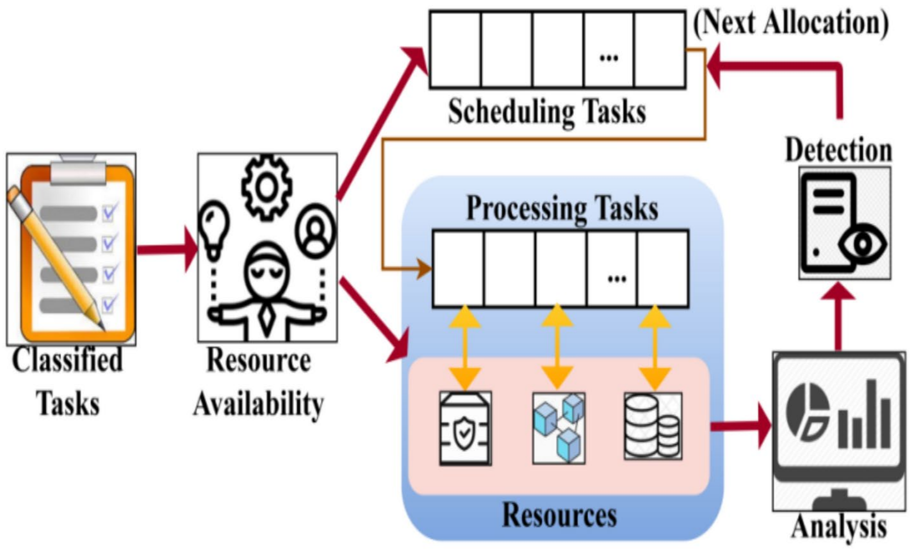

Fig. 4. Resource allocation process.

$$\begin{aligned} h_0\left(k_e\right) &= \prod\nolimits_{\delta}\left(\beta * q_t\right) + \left(\rho * \frac{m'}{c_e}\right) * \left(r' * o'\right) \\ &= \left(\frac{r' + t_k/o_n}{\rho}\right) * \left(o' - \nabla\right) - \left(\beta - m'\right) \end{aligned} \tag{8a}$$

In Eq. (8a), the processing time is computed by deploying the resource allocation to the ideal system; this predictive learning is used to predict the state of upcoming resources working. The detection process is evaluated for the varying resources in a P2P manner and provides a better task allocation. The preliminary step is to analyze the state of the resources and whether the process is queued; based on this, the task is assigned to the neighbor resources, termed as o'. Here, the analysis is carried out for the neighboring system, which performs the allocated task and states the predictive learning.

Predictive learning maps the preceding with the pursuing task and finds whether the particular system is ready to execute the task. In this method, the processing time is decreased and shows a better processing ratio, and the task monitoring is evaluated by equating. $\left(\frac{r'+t_k/o_n}{\rho}\right)$. Thus, the processing time is decreased by formulating the above Eq. (8a); the resource availability check is estimated for varying times and computed in the below equation.

$$\mathrm{o}_0\left(\mathrm{v_i}\right) = \left(\frac{1}{\mathrm{k_e}\left(\mathrm{x_g} + \mathrm{y}'\right)}\right) {*}\mathrm{q_t} - \left[\left(\mathrm{h_0} {*} \mathrm{d_a}\right) + \left(\mathrm{t_k}/\mathrm{a_b}\right)\right] * \left(\partial - \mathrm{k_e}\right) + \mathrm{c_e} * \mathrm{r}'\left(\mathrm{o}'\right) \tag{8b}$$

In Eq. (8b), The resource availability check is estimated for the number of systems in the IIoT; both the busy and ideal resources are available. Here, it verifies the system's state and provides the result that deploys the allocation process to the resources in a P2P manner. The task's length and density are identified, and the resources to the neighboring system improve the better performance. This verification of resource availability is executed for every state of the allocation process, making the resource allocation the ideal system. Computing improves the processing ratio and decreases the waiting time of the process. The task is enlisted in the queue and is responsible for sharing the resource with the other system. This process is evaluated in the following Eq. (9a).

$$\rho\left(l_i\right) = \left(m' * h_0\right) + \left(\frac{r'}{o_0}\right) * o' + \left(\frac{\nabla}{\sum v_i * c_e}\right) * \sum_{m'}\left(\delta + \beta\right) - k_e \tag{9a}$$

In Eq. (9a), The queued process in the IIoT application is monitored, and it is represented as l_i. It deploys the predictive learning approach that maps the preceding task state with the pursuing state and produces the output. It relates to the sharing of resources with the neighboring resources, and by assigning this task to the other system in the IIoT, the processing is done efficiently. The enlist holds the previous state of the task and monitors it to identify stagnancy. Thus, the task allocation enlist is performed reliably, and the P2P in IIoT processing is analyzed by deriving the following equation.

$$\nabla\left(\mathrm{o}'\right) = \prod_{\mathrm{q_t}}\left[\left(\mathrm{e_i} * \mathrm{a_b}\right) + \left(\frac{\mathrm{d_a} * \mu}{\beta}\right)\right] * \delta - \mathrm{a_b}\left(\mathrm{h_0}\right) - \mathrm{r}' + \arg\max \mathrm{r}'\left(\mathrm{d_a} + \mathrm{h_0}\right) - \mathrm{k_e} \tag{9b}$$

In Eq. (9b), the neighboring resources are analyzed and computed; here, the queued task is monitored and provides task sharing to the ideal system. The ideal resources are identified, and the prediction with the previous state provides efficient resource allocation that verifies the smart industry's available resources. The identification phase involves the sharing of resources on time, and it is termed as $\mathrm{r}'\left(\mathrm{d_a} + \mathrm{h_0}\right) - \mathrm{k_e}$. Thus, the P2P in IIoT is processed in the above equation. The integration of task scheduling and resource allocation is then performed to address the objective formulated in the section below.

```
Input: Resources ($o_0$,$r_{\prime}$), queued tasks ($t_k$), system state ($e_i$), task length (β), task density (ϱ)
Output: Optimized task allocation, reduced stagnancy
1. Initialize resources and tasks.
2. For each task:
        a. Check task readiness.
      b. Calculate stagnancy: $\mu = (o_0 + r_{\prime}) * \left(\frac{\nabla + m'}{a_b}\right) + \sum_{\delta} h_0(s_0) * \left(\frac{t_k/\partial}{e_i + m'}\right) + \left(x_g * y'\right) - k_e$
      c. Allocate task if stagnancy exists.
      d. Compute task time using: $h_0(k_e) = \left.\begin{matrix} \prod_{\delta}(\beta * q_t) + \left(\rho * \frac{m_{\prime}}{c_e}\right) * (r_{\prime} * o_{\prime}) \\ = \left(\frac{r_{\prime} + t_k/o_n}{\rho}\right) * (o_{\prime} - \nabla) - (\beta - m') \end{matrix}\right\}$
           e. Update resources and queue.
3. For each neighbor:
      a.   Calculate   task   processing   time$o_0(v_i) = \left(\frac{1}{k_e(x_g + y_{\prime})}\right) * q_t - \left[(h_0 * d_a) + \left({}^{t_k}/_{a_b}\right)\right]$
*                         $(\partial - k_e) + c_e * r_{\prime}(o_{\prime})$
      b. Share task if possible.
4. Check if task allocation is optimal.
5. Adjust allocation and reassign tasks if needed.
6. Output optimized allocation.
```

Pseudocode 2. Resource allocation—predictive learning

In pseudocode 2, Resource Allocation-Predictive Learning optimizes IIoT job allocation. Initializing resources and queued tasks reduces stagnancy and speeds processing. The algorithm checks each task for processing readiness and stagnancy. Tasks are distributed to neighbouring resources to optimize processing if stagnancy exists. The system continuously monitors neighbouring system resource availability to calculate task processing time and determine resource sharing. Checking if resources are optimally deployed and reallocating work to ideal systems refines the allocation process. Predictive learning improves resource allocation and decision-making. Optimized output reduces stagnancy and boosts system efficiency.

Task scheduling and resource allocation

These methods are performed by deploying predictive learning that states the previous state of task allocation to the ideal system and improves the performance and processing ratio. The integration process is done by performing the task scheduling for the number of resources; from that, the resource allocation is done by deploying predictive learning. It retrieves the task from the previous state and provides the mapping to decrease the waiting time and stagnancy factor. The following equation evaluates the integration of task scheduling and resource allocation with predictive learning.

$$\vartheta\left(a_b\right) = \int_{o_n}^{r'} t_k - d_a + \left(\frac{\nabla * \rho}{\sum_{e_i} \partial}\right) + h_0\left(o_0\left(s_0 - d_a\right)\right) + r'\left(k_e\right) - \beta \tag{10}$$

Equation 10 integrates task scheduling and resource allocation by evaluating the predictive learning method. Here, the preliminary step is used to identify the state of action of resources and provide the available resources. In this state, the processing time is analyzed and improved; the stagnancy is decreased because only the task is provided to the available ideal resources in the IIoT. Thus, the proposed work's objective is addressed by integrating the task scheduling and resource allocation process that deploys predictive learning.

The scheduling count varies for acceptance and failure of the task allocation process and shows the value low to wavy. Suppose the scheduling count increases and the failure decreases that deploy the prediction for the varying task. The scheduling is done by deploying the time factor associated with allocating the task to the resources (Fig. 5). The scheduling count is estimated for varying mapped tasks with the previous state of action and provides the task allocation. If the mapped resources increase, the sharing also increases for varying tasks in the smart industry. The mapped and shared figures show low to high values for varying scheduling counts (Fig. 6).

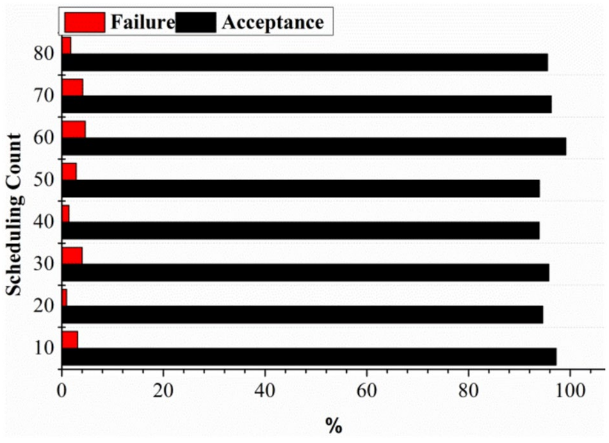

Fig. 5. Failure and acceptance %.

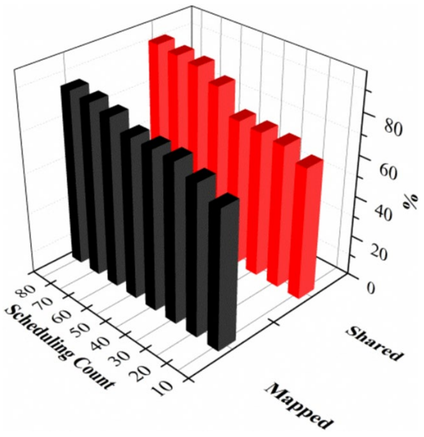

Fig. 6. Mapped and shared %.

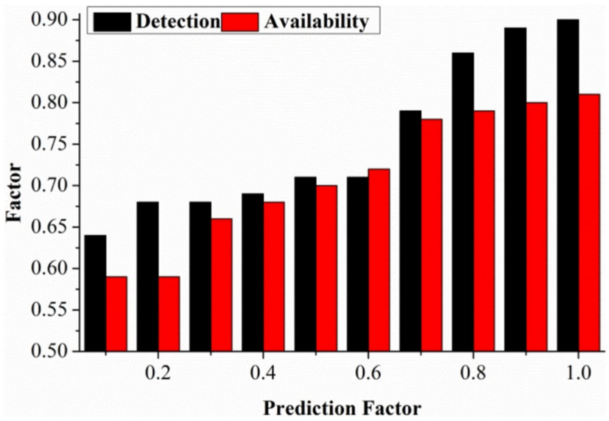

Fig. 7. Detection and availability factor.

The predictive factor is calculated for the varying factors associated with the detection and availability. Suppose the detection shows a higher range and the availability increases by performing a prediction. The detection of the busy and ideal state is used to relate to the prediction model of a task in the previous state of action (Fig. 7). The queuing percentage is evaluated to detect the stagnancy factor in this proposed work associated with the acceptance and processing (Fig. 8). Compare to accept the processing time shows the lesser value and

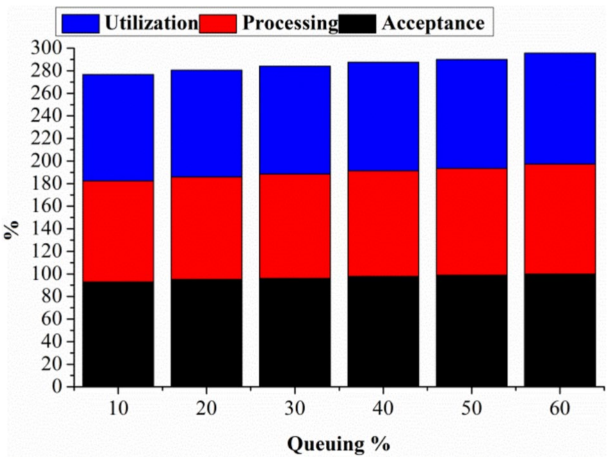

Fig. 8. Utilization, processing, and acceptance %.

Parameter	Value/range	Description
Number of tasks	1500	Total tasks processed in the simulation
Task size	50–500 MB	Varying task lengths for performance analysis
Processing devices	50	Number of machines handling the tasks
Scheduling instances	80	Total scheduling events conducted
Queuing rate	10 Tasks/instance	Number of tasks processed per instance
Performance metrics	Processing ratio, processing time, stagnancy factor, wait time, resource utilization, task acceptance ratio	
Comparative algorithms	MCRD[17], QA-HTS[18], GT-RA[25]	

Table 3. Parameter setting environment ranges.

the utilization increases by performing prediction. Here, resource acceptance, processing, and utilization show low to high value.

PSAS computation complexity depends on task scheduling, resource allocation, and stagnancy detection. Predictive learning scheduling analyzes prior task executions, adding sorting and prediction overhead, making it O (n log n). Resource allocation using greedy selection to analyze m resources for n activities is O(mn) challenging in the worst situation. Each scheduling cycle takes O(n) time for queue monitoring and reassignment in stagnancy detection. Thus, PSAS is efficient for large-scale IIoT settings and provides real-time adaptation with O (n log n + mn) complexity.

Result and discussion

This section discusses the performance of the proposed PSAS using simulation models. In this simulation, 1500 tasks of 50–500 Mb length are processed using 50 processing devices/machines. The simulation uses 80 scheduling instances with a queuing rate of 10 tasks/instance. The performance is verified using the metrics task processing ratio, processing time, stagnancy factor, wait time, resource utilization, and task acceptance ratio. The tasks and task lengths varied to analyze the above metrics. For a comparative study, the existing MCRD[17], QA-HTS [18], and GT-RA[25] are accounted for. Table 3 provides the hyperparameter setting range applied in this research.

More parameter setting citations to support the chosen values would strengthen the experimental setup's credibility. The selection of 1500 jobs, task sizes from 50 to 500 MB, 50 processing devices, and 80 scheduling instances was based on earlier research on scheduling efficiency in large-scale IIoT systems. For realistic workload distribution, the queue rate is 10 tasks per instance, the industry standard for real-time task scheduling systems. Include relevant articles that describe these experimental conditions to give the specified parameters greater weight. To understand how PSAS outperforms other approaches, a fast comparison is needed. The MCRD algorithm ranks jobs by execution date for real-time scheduling, but it can't manage changing workloads. QA-HTS distributes fog computing duties using quantized scheduling, although it performs poorly under large workloads. Game-Theoretic Resource Allocation (GT-RA) fails to handle work stagnation and real-time variations. This model optimizes resource utilization utilizing Nash and Stackelberg equilibria, however it lacks predictive learning. PSAS is more efficient at processing data, less sluggish, and uses resources better than these models, demonstrating its benefits in scalability, adaptability, and fault tolerance in IIoT-based P2P systems.

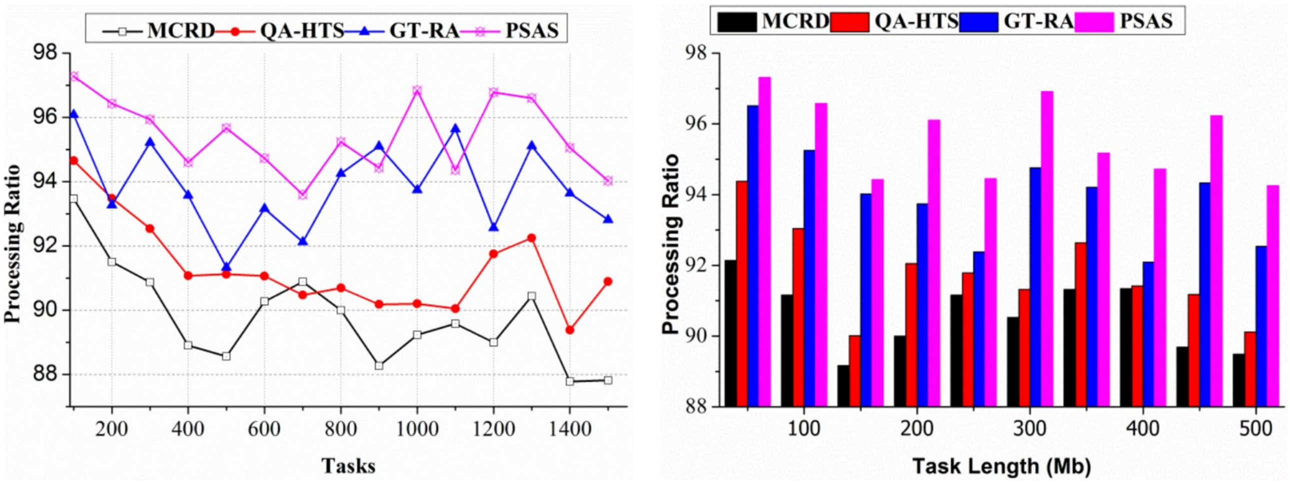

Fig. 9. Task processing ratio for tasks and task length.

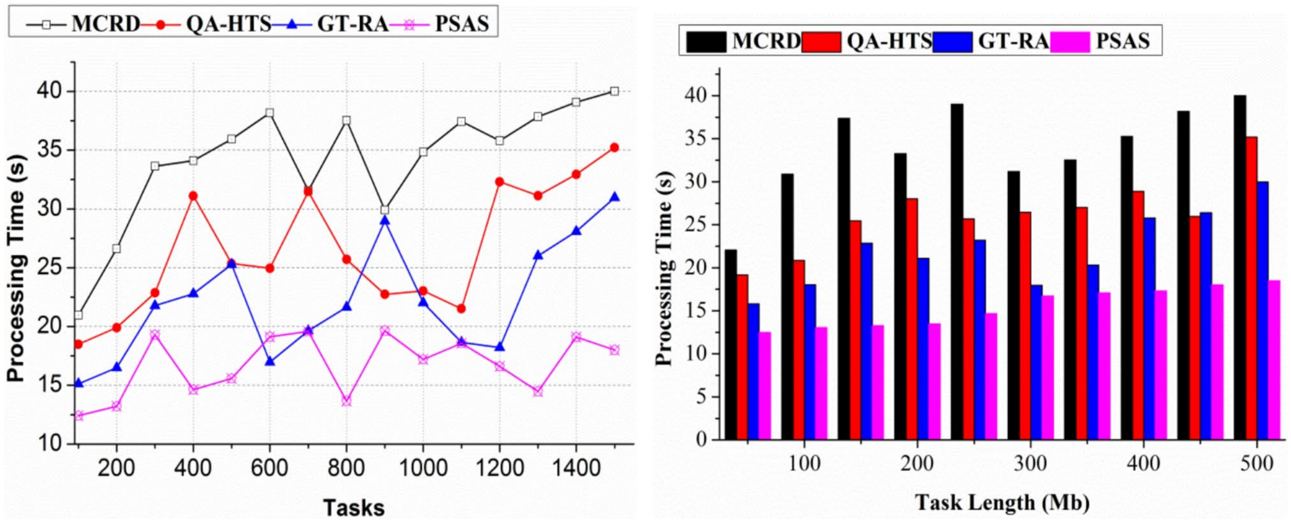

Fig. 10. Processing time for tasks and task length.

Task processing ratio

In Fig. 9, the task processing ratio for the proposed work increases for varying tasks and task lengths, which is represented as $\left(\frac{\mu/o_0}{t_k}\right)$. Here, the resources in the IIoT are detected and provide the task by deploying scheduling. The prediction model analyses the preceding information and provides the task allocation. Here, the stagnant information is detected by performing periodic monitoring to evaluate the state of the resources. In this task processing ratio, the analysis is used to assess the allocation process, and it is denoted as $\frac{s_0-d_a}{\prod_{x_g+y'} o_0}$.

The task length and density are identified by evaluating the verification phase. In this process, task scheduling is done for the varying resources in the smart industry. The stagnant task is identified in the queue by mapping with the previous state of the task. Resource sharing is done for the allocated system, which is monitored to find the task's changes. The sharing is carried out if the detection of available and unavailable resources is identified. The preliminary step is to determine whether the system state is either busy or ideal by deploying this; the task is provided. Here, periodic monitoring is done for the system that utilizes the task length and density in the smart industry associated with the number of resources.

Processing time

The processing time decreases for the varying task and task length, analyzing the available state of the system in the IIoT (Fig. 10).

The scheduling task is used to find the stagnancy, compare it with the previously available task, and provide the result. The processing time is used to evaluate the allocated task, and it detects the length and density of the task. The busy state of the system is identified by equating. $\left(\frac{s_0+t_k}{\sum_{e_i}\rho}\right) * \left(\mu + \frac{x_g}{o_n}\right)$ Here, the length and density of the task are detected. The stagnancy is detected, performed by deploying the prediction phase, and mapping is carried out. The mapping is done by evaluating the task's allocation to the system and monitoring the performance. The queued tasks are addressed to find the smart industry's stagnancy and analyze the waiting time. It is associated with the varying resources that deploy the length and density of the task. The processing time is used to compute the task scheduling and resource allocation integration process. In this process, the time is monitored by deploying the task's allocation to the ideal resources. By deploying this, the system performs the allocated task and provides the result on time, and it is computed as $(\nabla - k_e) * (s_0 - d_a)$. Thus, the proposed work's processing time shows a lesser value range than the other three methods.

Stagnancy factor

The stagnancy factor decreases for the proposed work compared to the other three existing methods that vary for tasks and length. The stagnancy is monitored in the queued tasks. It determines the number of tasks being processed, represented as $v_i\,(o_0) + t_k\,(d_a) + a_b$. Here, the task is allocated to the ideal resources in the IIoT, and the processing time is monitored. The stagnancy is monitored by periodic monitoring of tasks and the allocation factor that deploys the mapping. The mapping is done for the varying resources and provides the result on time associated with the detection phase. In this method, task scheduling is done by deploying predictive learning and maps the information. If the system seems to be in the Free State, it is allocated to the particular system. Here, prediction learning is used to evaluate the length and density of the task, and it is denoted as $\prod_{\vartheta}\left(k_e + \frac{e_i}{qt}\right) + \rho - (x_g + y')$. Resources are shared for the ideal system related to the processing time. The density is measured for the smart industry's varying resources by performing the scheduling and resource allocation factor. Thus, the stagnancy is monitored in the queue and addressed by evaluating the scheduled task's predictive learning (Fig. 11).

Wait time

The wait time is calculated to analyze the task length in the IIoT used to share the resources. The task is allocated to the neighboring system if the system is busy. In this, the verification is performed to detect the system's state. The detection of the system state is done by formulating $\rho * k_e + \frac{d_a - s_0}{\beta}$ Here, the prediction is used to evaluate the resources. The system's available state is used to deploy the scheduling for the varying resources, and mapping is performed here. The mapping is used to analyze the stagnancy state, and the waiting time is decreased. The waiting time is reduced by reliably performing the system scheduling. Both the available and unavailable state of the system are identified. From that, it detects the length and density of the task. Only the ideal resources are allocated with the task that deploys the scheduling and allocation, and it is equated $\left(\frac{s_0+d_a}{\vartheta}\right) + (h_0 - \beta)$. Predictive learning is used for the task scheduling approach that utilizes the resources from the industry. The mapping is carried out for the preceding and pursuing state of the system and addresses the stagnancy for P2P. This queued task is monitored and shared with the neighboring resources by evaluating the scheduling and resource allocation (Fig. 12).

Resource utilization

The proposed work's resource utilization factor indicates a better analysis of the available system and shares the task. In this process, the task is allocated to the neighboring resources associated with the state of action. Task scheduling is evaluated by detecting the busy and ideal state for every instance; the task and state differ. The detection process of the state is done by equating. $(t_k * o_0/m' * \nabla)$ Here, the stagnancy is analyzed.

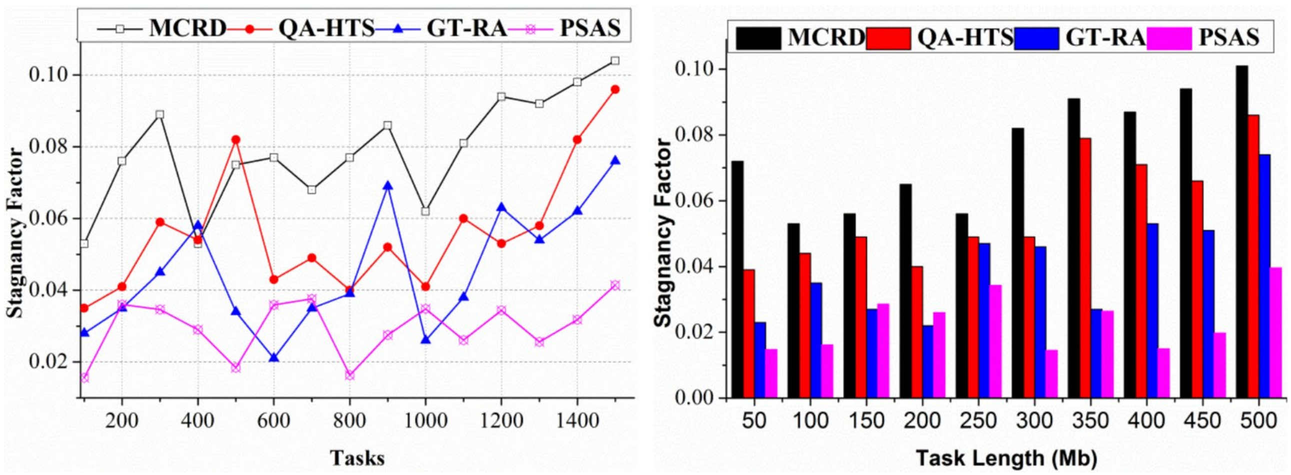

Fig. 11. Stagnancy factor for tasks and task length.

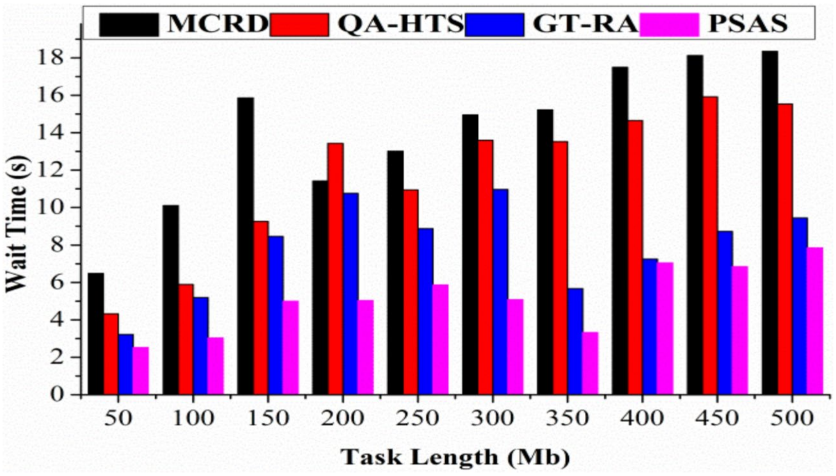

Fig. 12. Wait time for task length.

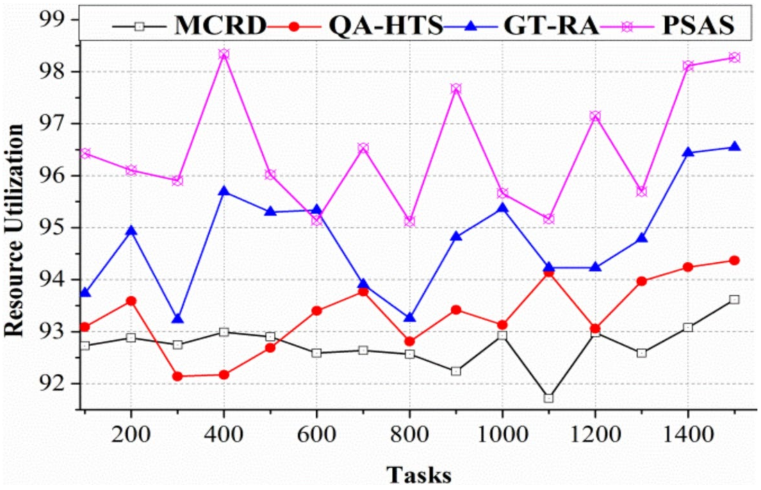

Fig. 13. Resource utilization for tasks.

The analysis finds the task's length and density and addresses the processing time and decreases. The waiting time is reduced for the varying resources, and the scheduling is performed by periodic monitoring. Predictive learning evaluates the system's state that deploys the queued process. Here, the system's available state is analyzed, and resource utilization in this proposed work decreases—the resource utilization increases for varying tasks assigned to the ideal system. The task assignment detects the system's state, maps the previous task length, and improves the upcoming task allocation. If the system's previous state is detected, the pursuing state will deploy better processing. It shows better resource utilization for the task allocation evaluated by deploying the scheduling (Fig. 13).

Task acceptance

In Fig. 14, the task acceptance for varying tasks in the smart industry deploys to identify the system's busy and ideal state. The task is assigned to systems and monitors the performance, and it is denoted as $(t_k - h_0) * \left(\frac{c_e - u'}{\beta}\right)$. The preceding state of the system is identified, and the task is allocated reliably. For the varying tasks, the scheduling is done, the resource allocation is performed, the stagnancy is addressed, and the length and density of the task are detected. The acceptance of the task is carried out for the ideal system associated with the prediction phase. Here, the acceptance ratio for the proposed work increases for varying tasks that deploys the availability detection.

The verification is carried out to analyze the queue process and decreases the stagnancy, and it is denoted as $\left(\frac{t_k/\partial}{e_i + m'}\right) - \rho$. The monitoring of task acceptance is used to deploy the scheduling and resource allocation phase. The mapping is used to analyze the system's processing state and improve the performance efficiently. Here, the verification is used to evaluate the detection of resources, and the integration is performed periodically. Here, the

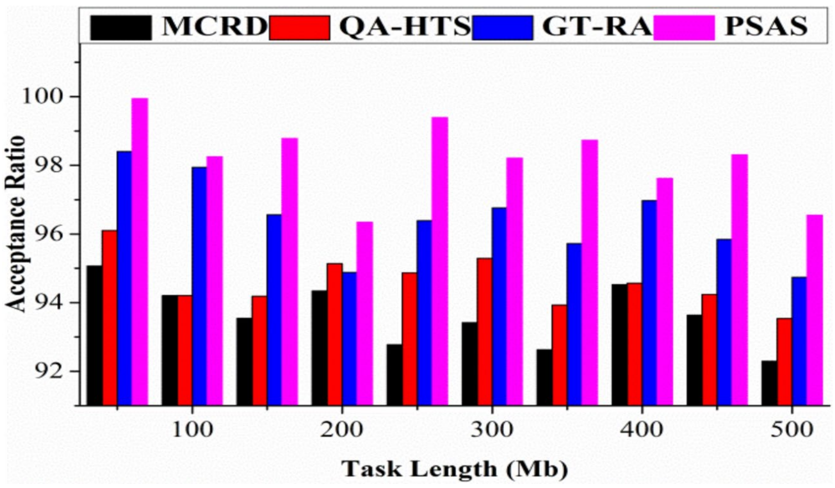

Fig. 14. Acceptance ratio for task length.

Metrics	MCRD	QA-HTS	GT-RA	PSAS	Improvement
Processing ratio	87.82	90.89	92.81	94.025	10.56% High
Processing time (s)	40.01	35.23	30.97	18.004	16.38% Less
Stagnancy factor	0.104	0.096	0.076	0.0414	5.06% Less
Resource utilization	93.62	94.37	96.55	98.275	10.29% High

Table 4. Comparison summary for tasks.

Metrics	MCRD	QA-HTS	GT-RA	PSAS	Improvement
Processing ratio	89.49	90.12	92.54	94.257	10.62% High
Processing time (s)	40.01	35.2	29.96	18.489	15.75% Less
Stagnancy factor	0.101	0.086	0.074	0.0396	4.74% Less
Acceptance ratio	92.3	93.54	94.74	96.551	9.07% High
Wait time (s)	18.35	15.54	9.45	7.848	15.22% Less

Table 5. Comparison summary for task length.

task acceptance ratio increases by integrating scheduling and resource allocation. In Tables 4 and 5, the summary of the comparison is presented with the improvements.

Peer-dependent Scheduling and Allocation Scheme (PSAS) outperforms current techniques by integrating adaptive resource allocation, decentralized scheduling, and predictive learning. This avoids the limitations of centralized and static systems. PSAS ensures optimal task allocation even in dynamic conditions, unlike MCRD's deadline-based job prioritization, which has scalability and workload issues. Unlike QA-HTS, which uses a quantized scheduling approach but lacks real-time predictive capabilities, PSAS actively detects stagnation and workload imbalances, reducing processing delays and queue wait times. PSAS enhances system throughput and task acceptance rates by using a hybrid scheduling method that continually monitors resource states, unlike GT-RA, which optimizes resource allocation using game-theoretic models but does not dynamically react to changing task demands. PSAS is totally decentralized, meaning peer nodes may communicate and redistribute workloads independently, improving load balancing and fault tolerance, unlike centralized methods that produce scalability bottlenecks and single points of failure. PSAS reduces processing time (16.38%), stagnancy (5.06%), and task processing ratio (10.62%) better than current scheduling algorithms for large-scale IIoT systems.

Real-world IIoT environment uncertainties include workload changes, network latency, machine faults, and resource availability. PSAS uses predictive learning to adapt to system circumstances and decrease uncertainty by altering scheduling decisions based on previous patterns and present availability. PSAS dynamically redistributes work to keep operations operating during delays or interruptions, unlike static allocation. Scalability is essential for large-scale deployments because computational cost increases with workers (n) and resources (m). PSAS uses an adaptive queuing mechanism to prioritize urgent jobs and evenly distribute workload across all nodes to increase linearly. PSAS can manage hundreds of resources and thousands of tasks without becoming stuck, making it ideal for smart manufacturing and large-scale industrial automation.

PSAS, a predictive learning-based scheduling and allocation strategy for P2P-based IIoT jobs, is presented in this work. This paper makes numerous significant contributions. First, a predictive task scheduling algorithm considers resource availability, task dependencies, and historical execution trends. Second, a dynamic resource allocation architecture may respond to real-time workload fluctuations and cut processing time by 16.38%. A decentralized scheduling strategy leverages peer-driven job execution to eliminate single points of failure in centralized systems. Final performance measures demonstrate that PSAS increases task processing ratio by 10.62%, stagnancy by 5.06%, and resource usage by 10.29%. PSAS anticipates work execution limits, unlike reactive systems that postpone job execution. PSAS tracks system states and re-allocates resources as needed, unlike allocation models. PSAS's distributed structure improves load balancing and fault tolerance over centralized alternatives, which scale poorly. All of these enhancements make PSAS the best IIoT scheduling solution—scalable, versatile, and rapid.

Conclusion
A peer-dependent scheduling and allocation scheme is proposed in this paper to aid stagnancy-less task processing in a peer-to-peer-assisted industrial IoT environment. The proposed scheme gains knowledge of the task length for mapping them without delay. In this process, the available resources are detected and are sequentially allocated for the first-come, first-serve tasks. The queued time-requiring tasks are assigned based on the scheduling prediction instances. For this purpose, predictive learning is employed; the learning process recommends allocating tasks based on completion time. The deadline and completion time of the tasks are prominent in ensuring peer-system based task processing. The problem of stagnancy due to varying tasks and lengths is addressed using predictive learning and availability based resource allocation. The task scheduling and resource allocation processes are concurrent, providing seamless processing with less wait time. The proposed scheme reduces processing time and stagnancy factors while improving the task acceptance rate.

Future work will include incorporating advanced machine learning models, extending the scheme to dynamic environments, developing energy-efficient scheduling methods, testing it in edge-cloud scenarios, introducing robust fault-tolerance mechanisms, optimizing multiple objectives, and cross-industry validation. These enhancements aim to improve the scheduling and allocation framework's robustness, scalability, and efficiency for dynamic and complex IIoT systems in various industrial domains.

Future studies might include communication delays, hardware failures, and dynamic work priorities to increase practical reliability. PSAS must change its scheduling in real time to network latency, which may affect task execution timings. In case of hardware issues like machine failures that disrupt operations, PSAS's ability to reallocate blocked tasks in real time without lowering system throughput is crucial. Industrial work priorities may change due to urgent orders or emergency repairs. Testing PSAS in these settings would demonstrate its adaptability. It is feasible to model these uncertainties in real industrial case studies to further assess PSAS's fault tolerance, responsiveness, and reliability for large-scale IIoT installations.

Data availability
The authors will make the data available upon request to the corresponding Author.

Received: 17 February 2025; Accepted: 15 April 2025
Published online: 25 April 2025

References
1. Lin, L. et al. Computing power networking meets blockchain: A reputation-enhanced trading framework for decentralized IoT cloud services. *IEEE Internet Things J.* **11**, 17082–17096 (2024).
2. Aouedi, O., Vu, T. H., Sacco, A., Nguyen, D. C., Piamrat, K., Marchetto, G. & Pham, Q. V. A survey on intelligent Internet of Things: Applications, security, privacy, and future directions. *IEEE Commun. Surv. Tutor.* (2024).
3. Alabadi, M., Habbal, A. & Wei, X. Industrial internet of things: Requirements, architecture, challenges, and future research directions. *IEEE Access* **10**, 66374–66400 (2022).
4. Almusawi, A. & Pugazhenthi, S. Innovative resource distribution through multi-agent supply chain scheduling leveraging honey bee optimization techniques. *PatternIQ Min.* **1**(3), 48–62 (2024).
5. Almusawi, A. & Pugazhenthi, S. Innovative resource distribution through multi-agent supply chain scheduling leveraging honey bee optimization techniques. *PatternIQ Min.* **3**(1), 48–62 (2024).
6. Böhm, S. & Wirtz, G. Cloud-edge orchestration for smart cities: A review of Kubernetes-based orchestration architectures. *EAI Endorsed Trans. Smart Cities* **6**(18), e2–e2 (2022).
7. Silva, F., Grilo, C., Martinho, R. & Rijo, R. Scheduling molds manufacturing processes through process mining. *Procedia Comput. Sci.* **239**, 2348–2358 (2024).
8. Hu, Y. et al. Industrial internet of things intelligence empowering smart manufacturing: A literature review. *IEEE Internet Things J.* **11**, 19143–19167 (2024).
9. Fang, Z., Yang, X. & Xin, Z. Economic transformations in urban industries during energy transition: Novel tradeoff technique for balancing energy consumption. *Sustain. Cities Soc.* **105**, 105220 (2024).
10. Saleem, M. U., Usman, M. R., Yaqub, M. A., Liotta, A. & Asim, A. Smarter grid in the 5G era: Integrating the internet of things with a cyber-physical system. *IEEE Access.* **12**, 34002–34018 (2024).
11. Bing, L., Gu, Y., Aulin, T. & Wang, J. Design of auto configurable random access NOMA for URLLC industrial IoT networking. *IEEE Trans. Industr. Inf.* **20**(1), 190–200 (2023).
12. Al-Masri, E. et al. Energy-efficient cooperative resource allocation and task scheduling for Internet of Things environments. *Internet Things* **23**, 100832 (2023).
13. Hu, J. et al. Blockchain-enhanced fair and efficient energy trading in industrial internet of things. *Mob. Inf. Syst.* **2021**(1), 7397926 (2021).
14. Ibrar, M. et al. Adaptive capacity task offloading in multi-hop D2D-based social industrial IoT. *IEEE Trans. Netw. Sci. Eng.* **10**(5), 2843–2852 (2022).
15. Wang, Y., You, X. & Zhao, X. Blockchain and game theory enable high-efficiency data sharing in the IIoT. *Comput. Electr. Eng.* **118**, 109449 (2024).

16. Rehman, Z., Tariq, N., Moqurrab, S. A., Yoo, J. & Srivastava, G. Machine learning and internet of things applications in enterprise architectures: Solutions, challenges, and open issues. *Expert. Syst.* **41**(1), e13467 (2024).
17. Xia, C., Jin, X., Xu, C., Wang, Y. & Zeng, P. Real-time scheduling under heterogeneous routing for industrial Internet of Things. *Comput. Electr. Eng.* **86**, 106740 (2020).
18. Bhatia, M., Sood, S. K. & Kaur, S. Quantumized approach of load scheduling in fog computing environment for IoT applications. *Computing* **102**, 1097 (2020).
19. Farhan, M., Reza, T. N., Badal, F. R., Islam, M. R., Muyeen, S. M., Tasneem, Z. et al. Towards next generation Internet of Energy system: Framework and trends. *Energy AI* (2023)
20. Niragire, D. & Kwena, R. Effect of project risk management on success of community-based projects: A case of peer driven change project in Burera district, Rwanda. *Afr. Q. Soc. Sci. Rev.* **1**(4), 176–194 (2024).
21. Yung, S., Langi, A. Z., Arman, A. A. & Simatupang, T. M. Choosing and evaluating P2P lending with value engineering as a decision support system: An Indonesian case study. *Information* **15**(9), 544 (2024).
22. Padmaja, R. Analyzing the impact of performance management process on employee satisfaction: A case study of a growing IT organization in Hyderabad. *J. Tech. Educ.* **304**.
23. Vashishth, T. K., Sharma, V., Sharma, K. K., Kumar, B., Chaudhary, S., & Panwar, R. Intelligent resource allocation and optimization for industrial robotics using AI and blockchain. In *AI and Blockchain Applications in Industrial Robotics* 82–110. IGI Global (2024).
24. Qin, W. et al. MCOTM: Mobility-aware computation offloading and task migration for edge computing in industrial IoT. *Futur. Gener. Comput. Syst.* **151**, 232–241 (2024).
25. Jie, Y., Guo, C., Choo, K. K. R., Liu, C. Z. & Li, M. Game-theoretic resource allocation for fog-based industrial internet of things environment. *IEEE Internet Things J.* **7**(4), 3041–3052 (2020).
26. Chen, Z., Hu, J., Min, G., Luo, C. & El-Ghazawi, T. Adaptive and efficient resource allocation in cloud datacenters using actor-critic deep reinforcement learning. *IEEE Trans. Parallel Distrib. Syst.* **33**(8), 1911–1923. https://doi.org/10.1109/TPDS.2021.3132422 (2021).
27. Zhang, C. & Yang, J. An energy-efficient collaborative offloading scheme with heterogeneous tasks for satellite edge computing. *IEEE Trans. Netw. Sci. Eng.* **11**(6), 6396–6407. https://doi.org/10.1109/TNSE.2024.3476968 (2024).
28. Chen, Z., Zhang, J., Min, G., Ning, Z. & Li, J. Traffic-aware lightweight hierarchical offloading toward adaptive slicing-enabled SAGIN. *IEEE J. Sel. Areas Commun.* **42**(12), 3536–3550. https://doi.org/10.1109/JSAC.2024.3459020 (2024).
29. Chen, Z., Huang, S., Min, G., Ning, Z., Li, J. & Zhang, Y. Mobility-aware seamless service migration and resource allocation in multi-edge IoV systems. *IEEE Trans. Mob. Comput.* (2025).
30. Chen, Z., Jiang, Q., Chen, L., Chen, X., Li, J. & Min, G. MC-2PF: A multi-edge cooperative universal framework for load prediction with personalized federated deep learning. *IEEE Trans. Mob. Comput.* https://doi.org/10.1109/TMC.2025.3528404

Acknowledgements
The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-IRC-2025-2932-01". The author extends the appreciation to the Deanship of Postgraduate Studies and Scientific Research at Majmaah University for funding this research work through the project number (ICR-2025-1723). The authors are thankful to the Deanship of Graduate Studies and Scientific Research at University of Bisha for supporting this work through the Fast-Track Research Support Program. This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R125), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Author contributions
All authors reviewed the manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to S.A. or G.A.

Reprints and permissions information is available at www.nature.com/reprints.