IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 February 2024, accepted 1 April 2024, date of publication 8 April 2024, date of current version 16 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3385636

RESEARCH ARTICLE

ODTRA-Based Task Offload Optimization for Industrial Internet of Things: Improving Efficiency and Performance With Digital Twins and Metaheuristic Optimization

DHIVYA SWAMINATHAN[1], ARUL RAJAGOPALAN[2], VENKATRAM NIDUMOLU[3], (Senior Member, IEEE), ROOBAEA ALROOBAEA[4], HOSSAM KOTB[5], KAREEM M. ABORAS[5], AND ALI ELRASHIDI[6,7]

[1]School of Electrical Engineering, Vellore Institute of Technology, Chennai, Tamil Nadu 600127, India
[2]Centre for Smart Grid Technologies, School of Electrical Engineering, Vellore Institute of Technology, Chennai, Tamil Nadu 600127, India
[3]Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Vijayawada, Andhra Pradesh 522302, India
[4]Department of Computer Science, College of Computers and Information Technology, Taif University, Taif 21944, Saudi Arabia
[5]Department of Electrical Power and Machines, Faculty of Engineering, Alexandria University, Alexandria 21544, Egypt
[6]Electrical Engineering Department, University of Business and Technology, Jeddah 23435, Saudi Arabia
[7]Engineering Mathematics Department, Faculty of Engineering, Alexandria University, Alexandria 21544, Egypt

Corresponding authors: Dhivya Swaminathan (dhivya.s2019@vitstudent.ac.in), Arul Rajagopalan (arulphd@yahoo.co.in), and Ali Elrashidi (a.elrashidi@ubt.edu.sa)

ABSTRACT The Industrial Internet of Things (IIoT) is the recent innovation that had revolutionized the industries by enabling interconnected devices and systems to exchange intelligent data. However, implementing and operating such IIoT systems have various challenges. This article addresses those challenges pertained to task offloading in IIoT in which the resource-intensive tasks are transmitted and executed on remote cloud servers. To optimize the task offloading decisions this work propose the integration of Digital Twins, which are the computer-generated replicas of physical objects. By using the functionalities of Digital Twins along with real-time monitoring, and metaheuristic optimization algorithms this work presents a task offloading model for IIoT. Through this combined framework, the proposed model attempts to minimize the task execution time by considering the server capacity, bandwidth constraints, and device power consumption. The proposed Offloading with Digital Twins and Raindrop Algorithm (ODTRA) algorithm that is based on the water cycle metaphor and the Probabilistic Recursive Local (PRL) search algorithm had efficiently optimizes offloading performance which was proven through different experiment simulation and analysis.

INDEX TERMS IIoT, digital twins, task offloading, metaheuristic optimization, water cycle metaphor, decision-making system.

I. INTRODUCTION

The Industrial Internet of Things (IIoT), which is a fundamental component of the term Industry 4.0, has resulted in significant developments in different domains by helping interlinked devices and networks to exchange and communicate data freely. By making the effective use of devices such as sensors, actuators, and advanced data analysis, the IIoT efficiently connects digital techniques and physical processes in a single process, consequently improving operational efficiency, productivity as a whole, and decision-making services [1], [2], [3], [4], [5]. While IIoT offers enormous chances, it also faces significant challenges. The successful implementation and operation of IIoT systems

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

require overcoming many hurdles. Some primary challenges include [6], [7], [8], [9], and [10].

A. DATA VOLUME AND VELOCITY

IIoT environments generate massive volumes of data from interconnected devices, sensors, and systems. It poses challenges in terms of data storage, processing, and analysis. Advanced data management and analytics measures are vital to effectively controlling the high data volume, velocity, and variety [11], [12].

B. RESOURCE CONSTRAINTS

Edge Devices (ED) in IIoT ecosystems have limited computational power, memory, and energy resources. It poses challenges when performing complex computations and processing large volumes of data locally. Efficient resource consumption and Task Allocation (TA) methods are necessary to optimize system performance while minimizing resource consumption [13], [14].

C. REAL-TIME DECISION-MAKING

Timely decision-making is crucial in IIoT applications. Limited processing capabilities, network delays, and variable data availability make real-time decision-making challenging. Organizations need fast and accurate Decision-Making Systems (DMS) using real-time data and advanced analytics [15], [16].

D. DYNAMIC ENVIRONMENTS

IIoT systems operate in dynamic environments with device mobility, changing network conditions, and shifting resource availability. Maintaining constant connectivity, managing handovers, and maintaining dynamic network and resource conditions are essential [17], [18].

E. SECURITY AND PRIVACY

The interconnected nature of IIoT systems increases their vulnerability to security threats. Unauthorized access, data breaches, and malicious attacks can have severe consequences. Robust security measures such as authentication, encryption, access controls, and continuous monitoring are required to protect IIoT systems and ensure privacy [19], [20].

F. INTEROPERABILITY AND STANDARDIZATION

Lack of interoperability and standardization in the IIoT environment makes participating devices and systems from multiple vendors challenging. The design is necessary to continue advancing attempts toward standardized procedures, protocols, and freely accessible gateways in order to ensure an integrated exchange of information and communication among the different elements of the IIoT [21], [22].

The data from sensors that have connections to other devices defies the processing power of ED. Additionally, there is a probability that analyzing this data nearby will result in a shortage of resources, issues with delay, and a boost in Energy Consumption (EC) [23], [24], [25]. The principle of Task Offloading (TO), in which tasks that require considerable resources are assigned to remote server locations in the cloud and executed by those servers, has come to prominence as an achievable approach for overcoming these problems. The initial implementation of TO in IIoT presents several significant issues [13], [26], [27], [28], [29]. The very first phase in the technique, when deciding which tasks should be offloaded and which should be run locally, involves paying close attention to several variables, like the unique features of the tasks, the amounts of resources that are available, the state of the network, and environmental restrictions.

Offloading decisions are made even more complicated by the unpredictable nature of IIoT environments, which are defined by changing network bandwidth, the mobility of devices, and the availability of resources. In the IoT, it is of the utmost importance to have effective practices for Resource Allocation (RA), Task Scheduling (TS), and optimization [29], [30], [31], [32], [33].

In the framework of the IIoT, the deployment of Digital Twins (DT) is an approach with demonstrated promise for addressing the problems that have been highlighted by TO [31], [31], [34], [35], [36]. Real-time data for research, management, and development can be achieved by the use of computer-aided DT of physical objects or systems [34], [37], [38], [39], [40]. In addition to improved access and predictiveness, IIoT-DT additionally supports resource management. The actual application of DT in TO for immediate tracking and evaluation of IIoT systems allows proactive decision-making and dynamic TA, which in turn generates data on the effectiveness of the system, the efficient use of resources, and the state of the network [41], [42], [43], [44], [45].

DT analyses and tests the impacts of offloading decisions on the functionality of the system and its financial performance [46]. Metaheuristic approaches have the potential to significantly enhance IIoT-TO decision-making through the application of factors such as evolutionary processes, swarm intelligence, and optimization approaches [47]. These mathematical methods resolve complicated optimization challenges successfully and efficiently without ensuring the optimal global solution, but they frequently result in near-optimal solutions within a feasible time limit [48]. The present work introduces a Task Offloading Model (TOM) for IIoT, which makes use of DT and a Metaheuristic Optimization (MO) methodology intended for offloading [49]. The framework that is proposed has the ultimate goal of significantly reducing the total TET while simultaneously taking into consideration the computation capacity of servers, bandwidth limitations, and the EC rate of ED [50].

The proposed work's main contributions are listed below:

- **Development of the odtra algorithm:** A MO algorithm for TO in IIoT networks, which includes DT, has been proposed and designated the acronym ODTRA. By assuming server capacity, the volume of bandwidth,

and device EC, ODTRA attempts to minimize Task Execution Time (TET) to the highest level possible. For the TO method in the most efficient procedure, the algorithm generated the use of the Probabilistic Recursive Local Search (PRLS) method and integrated the Water Cycle Metaphor (WCM).

- **Integration of DT:** The use of DT is implemented in the Task Offloading Model (TOM) in order to improve TA decisions more in environments connecting IIoT. The system is able to make smarter TO decisions because it can take advantage of real-time monitoring and predictive modeling. Proactive RA and TO have been rendered attainable by this fusion, which results in enhanced system efficiency and use of the system.
- **Optimization of offloading performance:** By examining the proper domain and applying the PRLS algorithm, ODTRA is able to optimize the efficiency of offloading. This iterative method is responsible for finding state-of-the-art results, which in turn optimizes system efficiency by drastically minimizing latency, improving energy efficiency, and sustaining the quality of service standards.
- **Potential for practical implementation:** A practical example of the feasible deployment of IIoT scenarios in the real world is presented by the ODTRA algorithm. Considering several restrictions and efficiency, RA provides a viable solution for OTO performance. The integration of DT further enhances its applicability by enabling real-time monitoring and up-to-date decision-making during offloading.

The article is systematized as follows: Section II analyses background studies in IIoT-TO; Section III presents the methods employed in the proposed TOM; and Section IV describes the detailed methodology and steps in implementing the ODTRA algorithm for TO in IIoT. Section V presents the proposed model's experimental setup, simulation results, and performance evaluation. Finally, Section VI concludes the article, summarizing the key contributions and highlighting the significance of the proposed TOM using DT and the ODTRA algorithm in IIoT environments.

II. RELATED WORKS

Researchers, including [51], have explained and discussed the advancements and challenges in DT. They explored several sectors, such as healthcare, industry, and smart cities. They highlighted technologies used in creating DT, including IoT, IIoT, REST, SOAP, cloud computing, Machine Learning (ML): (a) Supervised Learning (SL) and (b) Unsupervised Learning (UL), and Deep Learning (DL) [52]), diverse databases (MongoDB, Redis, MySQLi), and data analytics with recognized smart manufacturing as an advanced model with cognitive abilities, proposing a DT-based model that employed intelligent methods for autonomous manufacturing [53]. Reference [54] discussed the emergence of Industry 4.0, which promotes modernizing traditional manufacturing through technology-driven approaches. Recent technologies innovations such as IIoT, big data analytics, Augmented Reality (AR)/Virtual Reality (VR), and Artificial Intelligence (AI) [55] are pivotal in advancing smart manufacturing. The idea of DT involves creating digital replicas of physical objects [56]. In the Smart Manufacturing Systems (SMS) context, a remote semi-physical debugging method employing DT is proposed to address system issues [57]. This method is validated through a case study focusing on smartphone assembly line debugging [58]. Authors [59] introduce a multidimensional torsional model for driveline components in DT, enabling monitoring of remaining service life. The authors optimize and adjust data collection in virtual models to improve DT performance [60]. The virtual data obtained from DT displays consistency with actual data, facilitating predictive maintenance [61]. Their work [62] studied the DT's role in optimizing EC in industrial manufacturing, with experimental results demonstrating improved efficiency and reduced EC.

Reference [63] investigated integrating DT technology in sustainable manufacturing to enhance quality, productivity, and flexibility in intelligent manufacturing. They aimed to leverage these technologies for the overall enhancement of manufacturing operations [64]. A related study [65] stressed the challenges of difficult operating conditions, leading to frequent mishaps and accidents and high maintenance costs. The authors [66] proposed implementing DT system applications with automated processes to address these challenges to enhance security, automation, monitoring, and control in different domains–the comprehensively analyzed technologies and tools for improving DT technologies [67]. By employing 5D models, they tried to present their findings and facilitate better comprehension of the results [68]. In their study [69], their colleagues examined the fusion of DT, ML, and IoT for model-based system engineering with DT technology. They aimed to seamlessly integrate these technologies within DT systems, ensuring efficient application [70], [71]. Authors [72] designed an open-source framework to address integration challenges in the IIoT. The framework is a web application for communication protocol [73]. It provides developers with flexibility in protocol selection, testing, security enhancement, storage analysis, and issue resolution, offering valuable solutions for IIoT integration [67].

Today, there are numerous fields where DT is employed along with edge computing, as demonstrated in studies conducted by [74]. These studies explore the deployment of DT to enhance edge computing tasks such as offloading and RA [75]. Reference [76] emphasized that creating DT for physical entities can improve edge computing systems' task management and resource consumption. To address the challenge of minimizing End-to-End Delay (EED) in DT-assisted Mobile Edge Computing (MEC) for industrial automation, [77] proposed a solution that leverages the DT of MEC servers. In industrial backgrounds [78], this

approach demands the real-time monitoring of physical counterparts to assess the computational capabilities of MEC servers. Doing so optimizes EED, enhances overall performance, and improves efficiency. Reference [79] introduced a TOM using a DT framework that focuses on selecting consistent MEC servers based on channel state information and blockchain technology. The goal is to make OTO decisions in MEC environments, ensuring efficient resource consumption. In their study published by [80], they devised a Federated Learning (FL) method empowered by blockchain technology, which applies digital replicas to capture terminal devices' operational conditions and actions. The use of this method ensures a secure environment of learning and the confidentiality of data, which helps ensure an improvement in the level of security and privacy in FL environments [81].

Another study [82] exploited DT in order to develop a virtual representation of MEC networks, which permitted intelligent decisions to be made regarding lane changes in connected vehicles. This virtual replica assists in evaluating and orchestrating lane-changing strategies, helping more intelligent DMS [83]. Their study [84] proposed a system for Mobile Users (MUs) to minimize EED when TO→edge servers nearby. The method used DT to evaluate the edge server's state, enabling efficient offloading decisions while considering the constraint of long-term migration cost.

In work by [85], a DT system assisted MUs in selecting high-quality MEC servers. The system managed real-time network status to enable the MUs to offload the tasks to MEC servers by achieving reduced EC and latency. Reference [86] explored the use of DT in supporting aerial-assisted Internet of Vehicles (IoV). Based on network resource demands, two incentive mechanisms maximized vehicle support and energy efficiency [87]. The DT was crucial in optimizing network performance and RA in aerial-assisted IoV networks. DT technology monitors Electronic Devices (EDs) and real-time errors between the DT and physical entities, improving task efficiency and reducing system error [88]. Combining edge computing and this approach enhances network performance and resource optimization. Air-assisted IoV's dynamic DT was proposed in another study [89]. An Unmanned Aerial Vehicle (UAV) with a DT system improved energy efficiency and resource scheduling in air-assisted IoV networks [90].

The Accelerated Particle Swarm Optimization (APSO) algorithm, introduced by [91], is a dynamic programming-based PSO scheme designed to optimize computing time and minimize service costs in edge computing environments by efficiently TS and ensuring load balancing. Offloading communication and computing in-vehicle edge computing were developed. To save time and be cost-effective, offload decision-making and RA. Crowd Search-based Local Search (CSLS) and Particle Swarm Optimization based on Computational Offloading (PSOCO) improved edge resource scheduling system performance. A Biogeographic Optimization (BO) algorithm optimized multiple factors and enhanced system performance [92].

Researchers have recently improved the Water Cycle Algorithm (WCA), a Water-based Metaheuristic Optimization (WMO) algorithm, to find optimal solutions across domains [93]. Using flexible formulations and global search, the WCA solved the Job-Shop Scheduling Problem (JSSP) and Multi-Processor Scheduling Problem (MPSP) better than other algorithms [94]. WCA, a mathematical model, optimizes the Reliability-Redundancy Allocation Problem (RRAP) better than previous methods. Remanufacturing Rescheduling Problem (RRP) and LS operator integration model to improve Discrete Water Cycle Algorithm (DWCA) efficiency are the focus of the study. This approach solves real-world remanufacturing cases, proving the DWCA algorithm's value in optimization problems [95].

Reference [96] present an intelligent TSO that leverages AI integration in EC. This model is robust in its adaptability, security features, and ability to achieve low processing delays, making it highly effective in dynamic environments. However, it may not fully address the challenges in heterogeneous edge computing environments. Reference [97] propose an AI-enhanced offloading framework for IIoT, focusing on maximizing service accuracy within edge computing. This framework effectively balances the trade-off between delay and service accuracy. Nonetheless, it might not completely cater to the diverse computational capabilities present in complex IIoT scenarios.

Reference [98] introduce an optimal TSO in Mobile Edge Computing (MEC) and Device-to-Device (D2D) communication frameworks, applying Q-learning for minimizing EC and executing EED. While it shows promising energy efficiency and EED reduction results, its primary focus on MEC and D2D might limit its general applicability. Reference [99] develop a joint TO and RA system, targeting accuracy-aware ML-based IIoT applications in edge-cloud network architectures. This model is adept at minimizing the long-term average system cost, considering the inference accuracy of ML models. However, the complexity of its implementation and the potential requirement for extensive computational resources are notable limitations.

Tables 1 to 4 comprised the holistic overview of the scientific research based on categories.

III. METHODS

A. WATER CYCLE ALGORITHM (WCA)

The WCA [100] utilizes a metaphorical representation of raindrops and bodies of water to conceptualize its optimization process. The initial population, raindrops, represents optimization problem solutions. The best individual is analogous to a sea, while the promising raindrops are depicted as rivers. The remaining raindrops converge into streams, merging into rivers, ultimately finding their way to the sea. Figure 1 presents the block chart for the WCA. In terms of the WCA, a raindrop or a single solution to the

TABLE 1. Literature summary of DT technology in smart manufacturing and performance enhancement.

Category		Ref	Main Contribution	Strengths	Weaknesses	Relation to Proposed Work
DT Technology in Smart Manufacturing and Performance Enhancement	Applications in Various Sectors	[16]	Advancements and challenges in DT	Comprehensive sector analysis	Lack of depth in specific areas	Foundational knowledge for DT application in IIoT
		[17]	Framework for smart manufacturing	Smart manufacturing methods	Focused on manufacturing	Basis for DT in IIoT
	Advancements in Smart Manufacturing	[18]	Discussion on Industry 4.0	Highlights technology-driven approaches	Limited to traditional manufacturing	Contextualizes technological environment for ODTRA
		[19]	A case study in smartphone assembly	Practical application of digital twins	Specific to smartphone assembly	Demonstrates real-world application relevancy for ODTRA
		[20]	Energy optimization in manufacturing	Focus on energy efficiency	Narrow focus on energy features	Highlights energy optimization potential in ODTRA
	Performance Optimization and Enhancement	[21]	DT in sustainable manufacturing	Enhances overall operations	Limited to sustainable practices	Aligns with ODTRA's efficiency and sustainability
		[22]	DT for automated processes	Improves security and automation	Specific to certain conditions	Supports ODTRA's aim for enhanced automation
		[23]	Analysis of DT technologies	Broad technology review	Lacks specific application details	Provides technological context for ODTRA
		[24]	Fusion of DT, ML, and IoT in engineering	Integrates advanced technologies	Focus on model-based engineering	Illustrates integrated tech potential in ODTRA

optimization problem is denoted as a $1 \times N_{\text{var}}$ dimensional array, represented as Equation (1):

$$\text{Raindrop} = \left[X_1, X_2, \ldots, X_{N_{\text{var}}}\right] \tag{1}$$

In the optimization problem, N_{var} represents the problem's dimension. The cost function (Cost $_i$) for each raindrop is calculated to determine its effectiveness using the following Equation (2):

$$\text{Cost}_i = \int \left(X_1^i, X_2^i, \ldots, X_{N_{\text{var}}}^i\right), i = 1, 2, 3, \ldots, N_{\text{pop}} \tag{2}$$

where the population count of the raindrops is represented by N_{pop} . The direction of flow for the raindrops, whether they go into the sea or the rivers, is determined by the following Equation (3), considering the strength of the flow:

$$NS_n = \text{round}\left\{\left|\frac{\text{Cost}_n}{\sum_{i=1}^{N_{Sr}} \text{Cost}_i}\right| \times N_{\text{Raindrops}}\right\}, n = 1, 2, \ldots, N_{sr} \tag{3}$$

where N_{sr} represents the sum of rivers and a single sea, and $N_{\text{Raindrops}}$ indicates the population of raindrops that either directly flow into the rivers or enter the sea. Streams are produced due to rainfall, which later converge to form rivers or directly flow into the ocean. The sea is the ultimate destination for all rivers and streams, representing the best optimal point. The following Equations (4) and (5) determine the updated positions of the rivers and streams:

$$X_{\text{River}}^{i+} = X_{\text{River}}^i + \text{rand} \times C \times \left(X_{\text{Sea}}^i - X_{\text{River}}^i\right) \tag{4}$$

$$X_{\text{Stream}}^{i+} = X_{\text{Stream}}^i + \text{rand} \times C \times \left(X_{\text{River}}^i - X_{\text{Stream}}^i\right) \tag{5}$$

Here, the value of C lies within the range of 1 to 2, and rand corresponds to a random number uniformly distributed between 0 and 1. The stream and the river could interchange their roles if the solution represented by the stream is better than the rivers. Similarly, the sea and the rivers can also swap their roles. The inclusion of the parameter $d_{\max}$. It plays a crucial role in deciding whether to encourage or restrict an expanded search around the sea, representing the optimal solution. Its value is updated according to the Equation (6):

$$d_{\max}^{i+1} = d_{\max}^i - \frac{d_{\max}^i}{\text{Iter\_Max}} \tag{6}$$

Following evaporation, rain occurs, giving rise to new streams. These streams consist of newly formed raindrops located at various positions, which are defined as Equation (7):

$$X_{\text{Stream}}^{\text{new}} = LB + \text{rand} \times (UB - LB) \tag{7}$$

Here, LB and UB represent the lower and upper bounds, respectively, defining the range within which the positions of the new raindrops are randomly determined. The "*rand*" term means a uniformly distributed random number between 0 and 1, used to calculate the offset from the lower bound to obtain the position of each new raindrop in the stream. To enhance the algorithm's performance and convergence in

TABLE 2. Literature summary of DT and edge computing integration.

Category	Study Reference	Main Contribution	Strengths	Weaknesses	Relation to Proposed Work
DT and Edge Computing Integration	[26]–[29]	Deployment of DT in edge computing	Enhances offloading and RA	Focus only on specific edge computing scenarios	Provides groundwork for digital twin integration in ODTRA
	[13]	Improving task management in edge systems via DT	Enhanced task management and resource usage	Specific to physical entities' management	Demonstrates the utility of DT in ODTRA task management
	[30]	DT-assisted MEC for industrial automation	Optimizes latency and performance	Focuses mainly on industrial automation	Directly informs the ODTRA approach to latency optimization
	[31]	TOM using DT framework in MEC environments	Efficient resource utilization	Relies on blockchain and specific channel state information	Offers a model for efficient OTO decisions in ODTRA
	[32]	Federated Learning (FL) with blockchain and DT	Enhances learning security and data privacy	Specific to FL scenarios	Influences security and privacy considerations in ODTRA
	[29]	Virtual representation of MEC networks for intelligent decisions	Aids in orchestrating complex strategies	Limited to connected vehicle scenarios	Provides insights into smart decision-making for ODTRA
	[33]	Minimizing latency in edge server offloading	Efficient offloading decisions	Focuses on long-term migration cost	Proposal approaches for latency reduction in ODTRA
	[31]	High-quality MEC server selection via DT	Manages real-time network status	Specific to MUs' requirements	Supports ODTRA in optimizing server selection and TO
	[34]	Use of DT in aerial-assisted IoV	Maximizes vehicle satisfaction and efficiency	Focused on aerial-assisted IoV networks	Can be leveraged for network performance optimization in ODTRA
	[35]	Network performance and RA in IoV	Optimizes network resources	Specific to IoV contexts	Aligns with network optimization goals in ODTRA
	[36]	Monitoring EDs and reducing system error with DT	Improves task efficiency	Focuses on error monitoring	Enhances ODTRA's accuracy and efficiency
	[31]	Dynamic DT in air-assisted IoV networks	Improves energy efficiency and resource scheduling	Specific to UAV systems	Can inform resource scheduling approaches in ODTRA
	[34]	UAV with DT system in IoV networks	Enhances energy efficiency	Limited to UAV-based systems	Provides a model for energy-efficient resource management in ODTRA

TABLE 3. Literature summary of MO in edge computing.

Category	Study Reference	Main Contribution	Strengths	Weaknesses	Relation to Proposed Work
MO in Edge Computing	[37]	Accelerated Particle Swarm Optimization (APSO) algorithm	Efficient TS and balancing	Specific to edge computing environments	Could inform the optimization aspect of ODTRA for TS
	[38]	Offloading in-vehicle edge computing	Time and cost-saving in offloading	Limited to vehicle edge computing	Provides a model for efficient offloading in ODTRA
	[39]	Resource allocation and decision-making optimization	Enhances offloading efficiency	Focused on DMS	Offers strategies for decision-making in ODTRA
	[40]	CSLS and PSOCO for edge resource scheduling	Improves system performance	Specific to edge resource scheduling	Could be adapted for resource scheduling in ODTRA
	[41]	Biogeographic Optimization (BO) algorithm	Optimizes multiple factors concurrently	May lack focus on specific factors	Could enhance multi-factor optimization in ODTRA

constrained problems, Equation (7) is explicitly employed for streams that flow directly into the sea. Using Equation (8), the streams' new positions are determined:

$$X_{\text{Stream}}^{\text{new}} = X_{\text{sea}} + \sqrt{\mu} \times \text{randn}\ (1, N_{\text{var}}) \tag{8}$$

The coefficient 'μ' plays a significant role in defining the search range near the sea. By adjusting the value of μ, the algorithm's behaviour is controlled. Specifically, '$\gamma\mu$' is the standard deviation in generating normally distributed random number randn. It is essential to highlight that a higher value of 'μ' expands the feasible search area, allowing for exploring a broader region. In contrast, a reduced weight of 'μ' constrains the search to a narrower space near the sea. In essence, 'μ' is referred to as a measure of variance; the distribution of individuals, generated with a variance of 'μ', around the optimal point is determined, represented by the sea.

B. PROBABILISTIC RECURSIVE LOCAL SEARCH (PRLS)

PRLS, an expansion of traditional Local Search (LS) methods, shares similarities with multi-start LS by conducting multiple LSs. However, it distinguishes itself by adopting

TABLE 4. Literature summary of advancements in MOA.

Category	Study Reference	Main Contribution	Strengths	Weaknesses	Relation to Proposed Work
Advancements in MOA	[42], [43]	Improvement of the WCA	Effective in solving complex scheduling problems	It may require specific problem formulation	Could inform the optimization approach in ODTRA
	[44]	WCA for optimizing Reliability-Redundancy Allocation Problem (RRAP)	Efficient in specific allocation problems	Limited to RRAP scenarios	Offers a model for reliability optimization in ODTRA
	[45]	Integration of LS operator in Discrete Water Cycle Algorithm (DWCA) for remanufacturing rescheduling	Improves efficiency in real-world cases	Specific to remanufacturing scheduling	Can guide scheduling optimization in ODTRA

Algorithm 1 Water Cycle Algorithm

Input:

- N_{var}: Variable count.
- N_{pop} : Raindrops count.
- N_{sr}: the sum of rivers and the sea.
- LB: Lower bound of the search parameter.
- UB: Upper bound of a search parameter.
- Iter Max: maximum iterations.

Output:

- The algorithm found the best solution.

Step 1: Initialization

- Generate N_{pop} raindrops as the initial population.
- Evaluate the cost function for each raindrop.
- Select the best raindrop as the sea.

Step 2: Water Flow

- For each raindrop 'i', calculate the normalized strength of the flow using Equation (3)
- Assign each raindrop to a river or the sea based on the strength of its flow.
- Calculate the new positions for each stream and river using Equations (4) and (5)
- If a stream has a better solution than the river it flows into, its roles are exchanged.
- When rivers offer a better solution, seas take over.

Step 3: Evaporation and Rainfall

- Update the value of $d_{\max}$ using the Equation (6)
- For each stream that flows into the sea, generate a new raindrop with the Equation (8).
- For each stream that flows into a river, update its position with a new random value between LB and UB.

Step 4: Termination

- Return the best solution after the maximum iterations.
- Otherwise, go back to Step 2 and repeat the water flow process.

a probabilistic approach when selecting the starting point. Instead of randomly selecting a location in the state space, the PRLS method determines a point from the recently explored LS trajectory. It is guaranteed that the set point is at least improved over the initial starting point and holds the potential for achieving a superior outcome. The process continues with a new LS branch, identifying the next optimal transition operation until no further improvements are possible, following the termination condition of traditional LS. Moreover, as part of the recursive process, this approach involves revisiting the branch from which the current unit originated, aligning with the behaviour exhibited in depth-first search algorithms.

The algorithm concludes when one of the termination conditions is met: either the specified number of individual PRLS has been performed or the desired solution cost is achieved. In the case of the former scenario, an individual PRLS concludes when the root, which represents the initial LS route, has thoroughly travelled all of its branch points, commonly known as search trees. The primary parameter for PRLS is the branching probability, which determines the likelihood of branching at a specific end along the trajectory

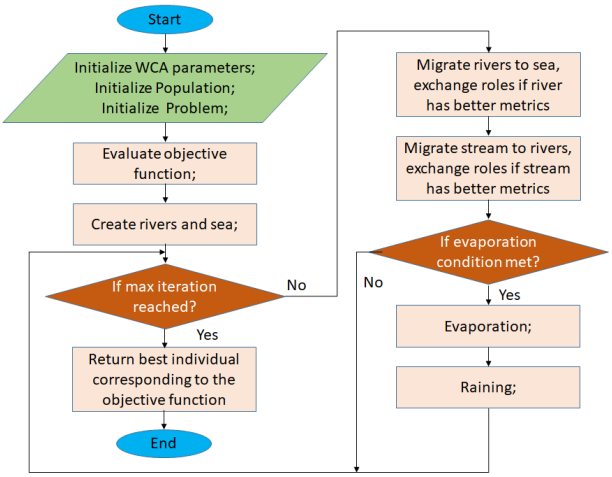


FIGURE 1. WCA block chart.

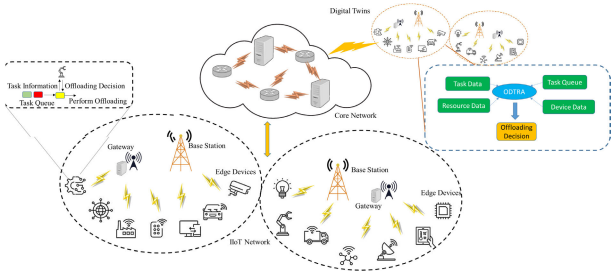


FIGURE 2. System architecture.

of the LS. Increasing the branching probability increases the possibility of generating densely populated search trees. Conversely, reducing the branching chance leads to forming search trees with a sparser distribution. A comprehensive algorithm for PRLS is outlined in detail as follows:

IV. PROPOSED MODEL

A. SYSTEM DESCRIPTION

The proposed edge-based DT framework is depicted in Figure 2. Industrial devices and machines are equipped with IoT sensors connected to EDs and gateways within an industrial setting. Each Edge Node is provided with an Edge Computing server that manages processing tasks obtained from industrial devices. It is assumed that the devices can start communication with Edge Nodes via wireless or wired connections, and there is no overlap in the coverage areas of the individual Edge Nodes.

Furthermore, the devices are set up to communicate with cloud servers and a central controller through Base Stations (BS) or gateways. The DT idea is applied to each industrial device, generating a virtual representation with information such as device status, sensor data, and operational parameters–the DT exchanges data, enabling them to collect comprehensive global information about the industrial system. The DT has offloaded DMS to optimize task execution and resource utilization.

In order to facilitate efficient offloading decision-making, the Offloading with Digital Twins and Raindrop Algorithm (ODTRA) has been implemented within the framework. The capabilities of DT and the Raindrop Algorithm are combined in ODTRA in order to make intelligent decisions regarding the offloading of work that is performed by industrial devices. IoT sensors provide device status, sensor readings, and other operational data to industrial machines' DT. Cloud-based DT offload based on system state, network connectivity, task requirements, and resource availability using the ODTRA algorithm. Signal strength and predicted data transfer rates are network connectivity, while task requirements are data size and computing resources.

The architecture incorporates a DMS for TO that considers both bandwidth and EC. DT within the system utilizes the ODTRA algorithm to decide on TO, factoring in the available bandwidth to prevent network congestion and the EC of devices to ensure efficient use of resources. The algorithm prioritizes TO to edge servers with sufficient bandwidth and results in lower EC than local processing, thus optimizing the operation based on these two critical parameters. The decision finds minimizing EED, maximizing resource utilization, and optimizing EC. After selecting the offloading decision, the DT relays the decision to either the local device or the Edge Node for execution. After task execution, the DT updates its parameters based on the device's status and other relevant information. It then proceeds to make offloading decisions for subsequent tasks, considering the updated system state and applying the ODTRA algorithm iteratively.

In IIoT, the motivation is frequently to confirm reliable and efficient communication within an industrial atmosphere. Hence, it is vital to consider network handovers when a device changes outside the coverage range of an Edge Node connectivity issue. To optimize network resource utilization, task data (without execution output) is typically not communicated between Edge Nodes. As a result, when a device transitions outside the network coverage region of the present Edge Node, any awaiting tasks that have not been transmitted are rejected. By applying ODTRA, which participates in the abilities of DT and the Raindrop Algorithm, this improved model for the IIoT allows it to be the real-time application for industrial devices to make smart offloading decisions, efficient task execution, and effective resource management.

B. THE COMPUTATION TOM

TOM functioned with DT for the IIoT investigation. The design that is proposed for edge computing involves both a server and a device at the edge of the network, and each task is encircled by a specific set of $N \equiv \{1, 2 \ldots n\}$. A wireless base station, which is marked by the symbol 's', is available in order to improve communication among the users of the edge computing servers and the edge computing users. DT, which describes digital representations of real objects or systems,

Algorithm 2 Probabilistic Recursive Local Search

Input:

- α: the probability of branching at a point on the LS trajectory
- *Max_Iterations*: the maximum number of individual PRLS to perform
- *Target_Cost*: the solution cost at which to terminate the overall algorithm

Output:

- *Best_Solution*
- *Best_Cost*

Step 1. Set *Best_Solution*=null and *Best_Cost*=infinity.
Step 2. Initialize iteration = 1
Step 3. While iteration<=*Max_Iterations* and *Best_Cost* > *Target_Cost*:
Step 4. Select an initial point in the state space.
Step 5. Set *Current_Solution*=initial point and *Current_Cost* = cost (*Current_Solution*).
Step 6. While true:
Step 7. Perform a LS from *Current_Solution* to find the best neighbouring solution.
Step 8. The Current Cost is higher than the best neighbouring solution.
Step 9. Set *Current_Solution*=best neighboring solution and *Current_Cost*=cost (*Current_Solution*).
Step 10. If *Current_Cost*<*Best_Cost*:
Step 11. Set *Best_Solution*=*Current_Solution* and *Best_Cost*=*Current_Cost*.
Step 12. With probability 'α', select a point within the recently completed LS trajectory.
Step 13. If no such point is selected, terminate the current PRLS and return to step 3.
Step 14. A branch from the selected point and perform a new LS from the branched solution.
Step 15. Increment iteration.
Step 16. Return *Best_Solution* and *Best_Cost*.

has the ability to improve the performance of the computation offloading process by providing a virtual model of the actual system that is both more precise and contains all of its components – (TO and RA). The DT hypothesis, combined with the communication and computation models, provides the potential to offload computations in edge computing successfully.

1) COMMUNICATION MODEL

In the framework that has been suggested, every single ED user has access to a related digital twin, which is also known as a 'DT_i'. This 'DT_i' acts as a digital copy of the physical device and can provide help in making decisions regarding computation offloading. A channel coordinator for the User Nodes (UNs) and the edge computing server is the wireless base station, which is denoted by the symbol 's'. The group $M = \{1, 2, 3 \ldots m\}$ is applied to signify the total number of wireless channels that the Base Station (BS) permits activation. The offloading decisions made by each ED are represented by the decision vector $X = (x_1, x_2, \ldots, x_n)$, where $x_i (i = 1, 2 \ldots \mathrm{n})$ indicates the fraction of the task ED 'i' chooses to offload for execution on edge servers. The remaining fraction of the task, $(1 - x_i) \times 100\%$ is executed locally on the ED. These functions offloading decisions originate with the support of the DT linked with each ED, $\mathrm{DT_i}$, which makes use of its intimate understanding of the computing power and tasks of the device.

To determine the transmission rate r_i of device '^{i}', the communication model accounts for various factors, such as the channel gain among BS 's' and ED 'i' (h_i), the transmission power of ED 'i' (p_i), the background noise power $\left(\sigma^2\right)$, and the effects from other nodes and their DT $\left(-p_i h_i + \sum_{j=1}^{n} p_j h_j + \tau_{DT_j}^2 + \lambda_{DT_j}^2\right.$, where τ_j^2 and λ_j^2 are the power levels of the interference caused by the DT of ED 'j' and the cross-talk from neighbouring channels. The transmission rate r_i is computed using the following Equation (9):

$$r_i = \omega \log_2 \left(1 + \frac{p_i h_i}{\sigma^2 + \sum_{j=1}^{n} p_j h_j + \tau_{DT_j}^2 + \lambda_{DT_j}^2 - p_i h_i}\right) r_i \tag{9}$$

The communication model considers the bandwidth 'ω' of the channel. It acknowledges that simultaneous offloading of computation by multiple EDs and their DT through the same channel can result in significant intrusion and reduced data rates, ultimately impacting the performance of edge computing. DT optimizes ED offloading decisions to improve system performance and minimize interference.

2) COMPUTATION MODEL

For the computation model, $C \equiv \{c_1, c_2 \ldots c_B\}$ refers to the tasks, with c_i representing the computation task for user 'i', measured in CPU cycles. Also, $B = \{\mathrm{b_1}, \mathrm{b_2} \ldots \mathrm{b}_n\}$ represents the data size, with each 'b_i' meaning the task data size for user

'i'. The set 'C' represents computation tasks, where each task 'i' has a total CPU cycle count 'c_i'. Similarly, the set **B** is used to represent the data sizes, where each task 'i' is associated with a data size denoted as b_i.

Furthermore, the set **U** represents the available EDs, denoted as $U = \{u_1, u_2 \ldots u_n\}$, while the set **E** represents the available edge servers for computation of TO, denoted as $E = \{e_1, e_2 \ldots e_n\}$. The set **S** represents the resources managed by each edge server, denoted as $S = \{s_1, s_2, \ldots, s_n\}$. The DT associated with each resource device in the edge server provides real-time information on resource usage and is represented as $DT_r = \{r_1, r_2 \ldots r_n\}$, where $r_i = \{q_1, q_2 \ldots q_n\}$, and q_i represents a specific resource of r_i. Then the set as $J = \{j_1, j_2 \ldots j_l\}$ is used to denote the edge server's task model, where the i^{th} task is $j_i = \{y_1, y_2 \ldots y_m\}$, and y_i represents the demand for a resource for a particular task.

RA, usage schedules obtained from the DT and the specific computation and data required for the task are all considered for view by the RA system.

a: LOCAL COMPUTING

In local computing, the task is executed in the local device 'n', leveraging the support and assistance the deployed DT provides. The DT stores real-time data on resource usage, enabling the ED to optimize its computation capability. The TET as T_i^L of user task 'i', when executed locally, is expressed as the following Equation (10)

$$T_i^L = \frac{u}{Z_i^L + D_i} \tag{10}$$

Here, Z_i^L represents the initial computation capability of ED user 'i', D_i is the delay caused by DT processing, and $u = (1 - x_i)\, c_i.x_i$. The decision vector 'x', as mentioned earlier, is used as a balance factor between local computing and offloading. Regarding the EC of the computation, it is described by the Equation (11):

$$E_i^L = \eta \left(\left(Z_i^L + D_i \right)^2 + B_i^2 \right) u \tag{11}$$

In the equation, 'η' represents a parameter related to the EC per CPU cycle, $Z_i^L + D_i$ is the total computation time for task 'i' (including the DT processing delay), B_i represents the data size associated with task 'i', and 'u' is the computation capability adjusted by the decision vector x_i. The term B_i^2 represents the EC of data transmission between the ED and the DT. It is proportional to the square of the data size B_i. It indicates that EC increases exponentially as the data size increases, emphasizing the importance of efficient data transmission devices in minimizing EC.

b: OFFLOAD COMPUTING

In the remote computing approach using the DT, the task is offloaded to the edge servers from EDs 'i' through BS (s). Nevertheless, the process of computation offloading introduces additional overhead in terms of time and EC, as it requires the transmission of computation data. To calculate the time overhead incurred by ED 'i' when the task is offloaded to the edge servers, the following Equation (12) is used:

$$T_{\text{i}}^M = v + \frac{k}{z_i^M + DT_i^M} \tag{12}$$

where 'v' represents the time required for data transmission and 'k' is a constant factor. The term Z_i^M denotes the user i 's computation capability while DT_i^M represents the information provided by the DT specific to user 'i'. Assuming that each user has a similar computation capability, it is as $Z_1^M = Z_2^M = \cdots = Z_n^M \cdot v = \frac{x_i b_i}{r_i}, k = x_i c_i * b_i$. The time required for data transmission, 'v', is calculated as $\frac{x_i b_i}{r_i}$, where x_i denotes the offloading decision of user 'i', b_i is the data size associated with the user i 's task, and r_i is a fixed factor.

Additionally, 'k' is defined as $x_i c_i * b_i$, which denotes the data transmission time to the server by each user 'i'. The term $\frac{k}{Z_i^M + DT_i^M}$ represents the server's processing time for each received task from user ' i ', considering the computation capability Z_i^M and the information provided by the DT layer, DT_i^M. In this context, the time overhead associated with sending back the computation results is considered negligible because the result data size is smaller than the input data size. The following Equation (13) displays the EC for ED 'i' to conduct task offload to the edge server:

$$E_i^M = p_i v + \eta \left(\left(Z_i^M + DT_i^M \right)^2 + B_i^2 \right) k \tag{13}$$

where, $p_i v$ represents the EC generated by user 'i' during the data transmission to the edge servers. The term $\eta \left(\left(Z_i^M + DT_i^M \right)^2 + B_i^2 \right) k$ represents the EC required by the servers to execute the user i' s task, considering the computation capability $\left(Z_i^M \right)$ and the information provided by the digital twin layer $\left(DT_i^M \right)$. The data size of the task is represented by B_i.

The optimization model for the problem is defined as following Equations (14) to (16)

$$\min F(X) = (f_1(X), f_2(X)) \tag{14}$$

$$f_1(X) = \sum_{i=1}^{n} \left(T_i^L + T_i^M \right) \tag{15}$$

$$\begin{aligned} f_2(X) &= \sum_{i=1}^{n} \left(E_i^L + E_i^M \right) \\ \text{s.t } X &= (x_1, x_2, \ldots, x_n) \\ x_i &\in [0, 1] \\ i &= 1, 2, \ldots\ldots, n \end{aligned} \tag{16}$$

where $F(X)$ represents the objective function to be minimized, consisting of two components: $f_1(X)$ and $f_2(X)$.The objective is to find the optimal values of **X** that minimize the total computation time ($f_1(X)$) and the total EC ($f_2(X)$) for all the tasks. $f_1(X)$ represents the sum of execution times for all tasks, including the local execution time $\left(T_i^L \right)$ and the

TABLE 5. Notation and description.

Notation	Description
$N = 1, 2, \ldots, n$	Set of tasks denoted by numbers from 1 to n
s	Wireless BS that facilitates communication between ED users and edge computing servers
$M = \{1, 2, 3, \ldots, m\}$	Set of wireless channels available for the BS to use
$X = (x_1, x_2, \ldots, x_n)$	Decision vector representing offloading decisions made by ED users
$[\mathrm{DT}]_i$	DT associated with ED user 'i'
h_i	BS-user 'i' channel gain
p_i	User 'i's transmitting power
σ^2	Background noise power
τ_j^2	The power level of interference caused by the DT of user 'j'
λ_j^2	The power level of cross-talk from neighboring channels
ω	Channel bandwidth
r_i	A transmission rate of 'i'
$C = \{c_1, c_2, \ldots, c_n\}$	Set of computation tasks where c_i represents the sum of CPU cycles required to complete the task 'i'
$B = \{b_1, b_2, \ldots, b_n\}$	Set of data sizes, where b_i is the volume of data associated with task 'i'
$U = \{u_1, u_2, \ldots, u_n\}$	Set of user equipment
$E = \{e_1, e_2, \ldots, e_n\}$	Set of edge servers available for offloading computation tasks
$S = \{s_1, s_2, \ldots, s_n\}$	Each edge server manages resources
$[\mathrm{DT}]_r = \{r_1, r_2, \ldots, r_n\}$	The DT of resource devices provides real-time information on resource usage
$J = j_1, j_2, \ldots, j_l$	Task model in an edge server
T_i^L	Local computation execution time of task by user 'i'
Z_i^L	The initial computation capability of ED user 'i' (measured in CPU cycles)
D_i	The delay caused by DT processing
x_i	A decision vector balances the trade-off between local computing and offloading
$u = (1 - x_i)c_i x_i$	The sum of CPU cycles necessary to finish task 'i' on ED user 'i'
E_i^L	EC of the computation for task user 'i' executed locally on their ED
η	Coefficient of CPU-EC
B_i^L	ED-DT data transmission EC
T_i^M	The offloading computation time of task by user 'i' to the edge server
Z_i^M	The computation capacity is accessible to ED user 'i', valued by the DT, and subsequently determined by the edge servers
$v = (x_i b_i)/r_i$	User 'i's server transmission time
$k = x_i c_i b_i$	The servers take the time to execute the task assigned by user 'i'
$[\mathrm{DT}]_i^M$	Time required to access the necessary resources from the DT layer

offloading TET $\left(T_i^M\right)$, as discussed in the previous section. It captures the overall time required to compute all the tasks. $f_2(X)$ represents the total EC for all tasks, including the local EC $\left(E_i^L\right)$ and the offloading EC $\left(E_i^M\right)$. It captures the overall EC in performing the computation tasks. The decision variables $X = (x_1, x_2, \ldots, x_n)$ Does each ED user make the offloading decisions x_i is the fraction of the TO (between 0 and 1). The constraints ensure that the values of x_i are within the valid range for each device user. The notations used in this work are summarized in Table 5.

C. PROBABILISTIC RECURSIVE LOCAL SEARCH (PRL) OPTIMIZED WATER CYCLE ALGORITHM (WCA) (PRL-WCA)

The WCA simulates nature's water cycle with a population-based MO. The algorithm generates a population of raindrops representing potential solutions to the optimization problem. Raindrops flow from streams to rivers and the sea, and the process is repeated until an acceptable solution is found. While the WCA is a robust optimization algorithm, it still has advantages from additional optimization techniques. The PRLS is a Probabilistic Local Search (PLS) technique that enhances the exploration and exploitation capabilities of the optimization algorithm–recursively searching for the best solution and accepting a new key if it improves the objective function value. The PRLS could effectively enhance the performance of many optimization algorithms, including GA, simulated annealing, and PSO.

To use the PRLS to optimize the WCA, PRLS is incorporated as a LS operator within the WCA framework. After the WCA generates an initial population of raindrops, PRLS is applied to each raindrop to refine its position within the solution space. To be more apparent, the PRLS is performed on the raindrop that was determined to be the best in the initial set of samples, and the solution that results is used as the baseline for the WCA cycles. The PRLS is used to apply to the finest raindrops in the population after the water flow step in each iteration of the WCA. This is done in order to refine the raindrop's position within the resulting domain while the WCA is being performed. This can be achieved through utilizing the PRLS to execute a recursive search in a nearby area of the best solution that is currently available and then accepting novel ideas that improve the value of the objective function in a method that is probabilistic. When the PRLS is employed as an LS operator in a WCA, its levels of exploration and exploitation are enhanced, which in turn increases the possibility of the WCA finding a high-quality result. The collective use of the PRLS and the WCA can, in more familiar terms, contribute to improved optimization performance. The algorithm's LS capabilities are enhanced by PRLS and WCA. This methodology could prove helpful in complex and robust solution environments where the WCA may have problems finding a solution that is of high quality.

D. OFFLOADING WITH DIGITAL TWINS AND RAINDROP ALGORITHM (ODTRA)

With DT, the offloading problem in edge cloud computing settings can be addressed with the help of ODTRA, which is a MO method. By examining parameters such as server resources, ED-server bandwidth, and energy consumption rate, the ODTRA technique is able to decrease the overall time required to finish tasks significantly. The metaphor used is based on the water cycle, where the water cycle symbolizes the best solution, and each raindrop shows a possible solution. There are five stages to the algorithm: starting, moving water, using probabilistic iterative LS, addressing rainfall and evaporated water, and finally employing DT to offload. A cost function that includes TET, server computation resources, the amount of bandwidth, and device EC generates the cost of each raindrop. The raindrop collection is bombarded with the water flow process, which represents the flow of water from optimal to defiled solutions. Raindrops are divided into

rivers or the sea according to their capacity, which is directly correlated to the cost of each one, by continuously changing their functions in accordance with the quality of each stream's and river's solution. This is because the system is able to investigate better solutions by reversing its functions when a stream's solution is better than the river it integrates into or when a river's solution is better than the sea's.

Assuming a random set of solutions and a set of candidate solutions generated from causing variables, the PRLS method is then applied to each raindrop. For each solution, the cost function is evaluated, and the one with the most significant result is selected as the preferred solution. Their comparable quality is used to determine and continuously improve the PRLS methodology. If the technique is better, it substitutes the raindrop and refreshes flow strength. The evaporation and rainfall process is then applied, which involves reducing the maximum flow strength (d_{max}) and generating new raindrops (or) updating the positions of existing ones based on their flow. DT offloads each task and device to a cloud-based server with minimal execution time. DT changes state with task execution. Steps 2-5 are repeated until the threshold for stopping is reached, generating the best solution. Edge cloud computing offloading optimization method ODTRA is good.

V. SIMULATION AND RESULTS

Simulation experiments were conducted to evaluate the proposed model and solution method. Simulations were run on a PC with a 3.7 GHz Intel Core i9-10900K processor with 64 GB RAM. In order to prove that the assessments were conducted effectively and that accurate results could be attained, this set-up was the best choice. By applying this setup, the performance and effectiveness of the model are evaluated, and the result is advanced under practical scenarios.

A. SIMULATION SETUP

To assess the performance measure of the ODTRA system in IIoT environments, this work considered a simulation setup encompassing a range of node configurations and functional variables. This setup mimics diverse functional scenarios, ranging from small-scale to large-scale IIoT environments, providing a complete knowledge of ODTRA's capabilities.

These research simulations included a difference in node counts to signify dissimilar scales of IIoT environments. Select a series from 20 to 100 nodes. This range was selected to simulate settings from localized, minimum networks (20 nodes) to maximum networks (100 nodes), thus laying a broad spectrum of real-world IIoT environments.

This network simulation combined three dissimilar task sizes: Small (1 KB), Medium (500 KB), and Large (1000 KB). These sizes were ideal to signify an extensive array of data processing and computational requirements distinctive in IIoT environments.

The standard benchmark test performance of the ODTRA system was related to numerous well-developed optimization approaches, namely ACO, PSO, GWO, WOA, and WCA.

Algorithm 3 Offloading With DT and Raindrop Algorithm

Input:

- M: Sum of EDs
- N: Sum of cloud servers
- D: Sum of tasks to be offloaded
- E: Execution time for each task on each device
- C: Computation capacity of each server
- T: Sum of TET on all devices
- N_{pov} : Sum of raindrops in the initial population.
- N_{sr}: Sum of rivers and the sea.
- LB: the lower bound of the search parameters.
- UB: the upper bound of the search parameters.
- Iter r_{Max} : the maximum number of iterations.
- PRL_{iter}: the number of iterations for PRLS.
- PRL_{step} : the step size for PRLS.
- B: the bandwidth between the local device and each ED.
- P: the EC rate of each ED.

Output:

- The algorithm found the best solution.

Step 1: Initialization

(a) Generate N_{pop}: raindrops as the initial population.

(b) Evaluate the cost function for each raindrop using the offloading problem Equation (17):

$$\text{Cost}_{\,i} = \sum_{d=1}^{D} E_d \frac{1 + (B \times D_i)}{P \times F_i} + \sum_{n=1}^{N} C_n \max(T - \sum_{d=1}^{D} E_d \times F_{(n,d)}, o \tag{17}$$

(c) where D_i and F_i are the decision variables for raindrop i and $F_{(n,d)}$ is the computation capacity of server ' n ' for task ' d '.

(d) Select the best raindrop as the sea.

(e) Set the current iteration count to 0.

Step 2: Water Flow

(a) For Each raindrop 'i', calculate the normalized strength of the flow as follows Equation (18)

$$NS_n = \text{round}\left\{\left|\frac{\text{Cost}_n}{\sum_{i=1}^{N_{sr}} \text{Cost}_{\,i}}\right| \times N_{pop}\right\}, n = 1, 2, \ldots, N_{sr} \tag{18}$$

(b) Assign Each raindrop to a river or the sea based on the strength of its flow.

(c) Compute the new positions for each stream and river using the following Equations (19) and (20):

$$X_{\text{Stream}}^{i+} = X_{\text{Stream}}^{i} + \text{rand} \times C \times \left(X_{\text{River}}^{i} - X_{\text{Stream}}^{i}\right) \tag{19}$$

$$X_{\text{River}}^{i+} = X_{\text{River}}^{i} + \text{rand} \times C \times \left(X_{\text{Sea}}^{i} - X_{\text{River}}^{i}\right) \tag{20}$$

Here, C denotes a value ranging from 1 to 2, and rand represents a randomly generated number uniformly distributed between 0 and 1.

(d) If the solution represented by a stream is better than the one described by the river it flows into, exchange their roles.

(e) If the solution represented by a river is superior to the solution represented by the sea, their roles are swapped.

Step 3: Apply PRLS

(a) For Each raindrop 'i', select a random neighbourhood n_i of size N_n such that $N_n \ll N_{\text{pop}}$.

(b) For Each raindrop 'i', generate a set of candidate solutions S_i by perturbing the variables in its neighborhood n_i.

(c) Evaluate the cost function for each candidate solution in S_i.

(d) For Each raindrop 'i', apply the following PRLS procedure:

(i) Select the best candidate solution in S_i as the current solution.

(ii) Setk $= 1$.

(iii) While k $<$ PRL_{iter} r:

* Select a random neighbour of the current solution.
* If the neighbour is better than the current solution, set the neighbour as the current solution and set k $= 1$.
* Otherwise, with the probability PRL_{step} , set the neighbour as the current solution, and increment k by 1.

(iv) If the current solution is better than the original raindrop, replace the original raindrop with the current solution.

(e) Update the strength of the flow for each raindrop.

Step 4: Evaporation and Rainfall

(i) Update the value of d_{max} using the following Equation (21):

$$d_{\max}^{i+1} = d_{\max}^{i} - \frac{d_{\max}^{i}}{\text{Iter}_{\text{Max}}} \tag{21}$$

(ii) For Each stream that flows into the sea, generate a new raindrop with the following Equation (22):

$$X_{\text{Stream}}^{\text{new}} = X_{\text{sea}} + \sqrt{\mu} \times \text{rand}_n\,(1, N_{\text{var}}) \tag{22}$$

where μ is a coefficient that controls the search range and rand $_n$ is a normally distributed random number.

(iii) For each stream that flows into a river, update its position with a new random value between LB and UB.

Step 5: Offloading with DT

(i) For Each task 'D' and each ED 'M', select the server 'n' with the lowest execution time when using the DT for device M.

(ii) Offload task D from device M to server 'n'.

(iii) Update the state of the DT for device M based on the executed task.

(iv) Update the TET 'D' on device M to be the execution time obtained when offloading to server 'n'.

(v) Update the total TET to 'T' tasks and devices.

(vi) Repeat Steps 2-5 until the stopping criterion is met.

Step 6: Termination

(a) If the maximum number of iterations Iter_{Max} is reached, return the best solution found so far.

(b) Otherwise, increment the iteration count and go to Step 2.

This proportional method lets us contextualize ODTRA's performance within the current optimization techniques environment. All the simulation tests were shown using MATLAB Release 2018a (R2018a).

Specific parameters for each algorithm for the analysis are listed below:

(a) Ant Colony Optimization (ACO): Set with a pheromone evaporation rate of 0.5 and 100 ants, tailoring it for network pathfinding tasks in IIoT environments.

(b) Particle Swarm Optimization (PSO): Configured with particle velocities having an inertia weight of 0.5 and social and cognitive parameters set to 2, ensuring effective convergence in network optimization.

(c) Grey Wolf Optimizer (GWO): Adjusted with a pack size of 50 and a maximum iteration count of 100, optimizing its efficiency for network-related tasks.

(d) Whale Optimization Algorithm (WOA): Calibrated with a spiral constant of 1.5 for spiral updating position, and bubble-net behavior engaged every ten iterations, aligning with the dynamic nature of IIoT environments.

(e) Water Cycle Algorithm (WCA): Set with a raindrop evaporation rate of 0.05 and flow rate parameters optimized for searching network solutions.

The simulation parameters are described below:

(a) $\boldsymbol{c_s}$: It is the size of small tasks, which is $1KB$ (kilobyte). Small tasks typically involve processing (or) transmitting a relatively small volume of data.

(b) $\boldsymbol{c_m}$: It is the size of medium tasks, which is 500 KB. Medium tasks are more significant than small tasks and require more computational resources.

(c) c_b: This symbol represents the size of large tasks, 1000 KB. Large tasks are the largest among the three groups and may require considerable computational capabilities.

(d) $\boldsymbol{P}$: It has the maximum transmission power and has a value range from 10 to 100 W. It indicates the total energy that is used for transmitting data between nodes.

(e) $\boldsymbol{f}_l$: It is the computation capacity of local devices with a value range from 0.5 to 1GHz. It indicates the processing energy or computational ability of the individual devices in the network.

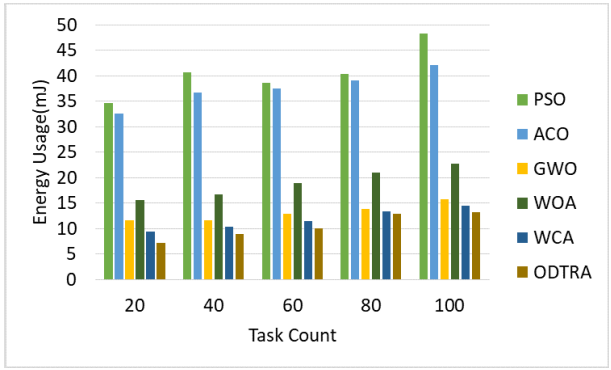
(f) f_c: It is the computation capacity of edge servers and has a value of 10GHz. Edge servers are typically more potent than local devices and manage complex computations.
(g) L: This symbol represents the distance between local devices and edge servers, ranging from 1 to 30 m. The distance between devices affects the quality and efficiency of communication.
(h) ω: It is the channel bandwidth, which is 5.0×10^{-3}GHz. It signifies the maximum data transfer rate or capacity of the communication channel.
(i) σ^2: It is the background noise power, which is 1.0×10^{-13} W. It represents the redundant or random noise present in the communication channel.

By varying these simulation parameters and observing their impact on the performance metrics, such as latency, throughput, or EC, the simulation aims to compare the proposed model with other algorithms. Let's find out how well the proposed approach performs under different scenarios and get a few recommendations for how to improve on it more successfully with this examination. Network EC, cumulative delay, and the total amount of tasks offloaded are the three primary parameters evaluated to assess the results achieved by the models. For the aim of performing accurate testing, the values for T (delay) and E (energy) have been specified as 0.5. This shows that both factors have been assigned similar weights.

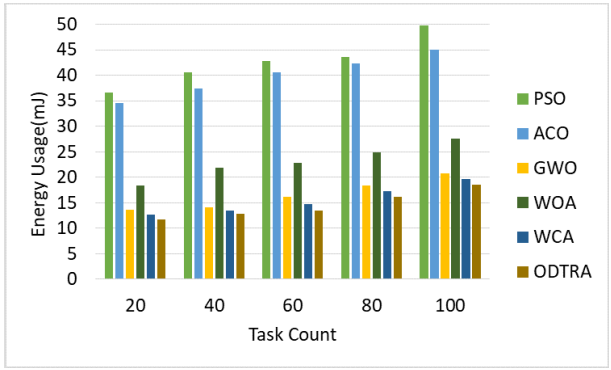
Every node's average time delay and EC are considered when measuring the total performance. The objective is to mitigate EC and EED within the entire network by incorporating network-wide metrics with consideration, compared to emphasizing particular nodes in the network. With the above approach, the entire network as a whole gets advantages from offloading decisions that take neighboring nodes' requirements into account. To better discover the algorithms' success and effectiveness and to make intelligent choices about TOM, these network-level parameters have been investigated.

B. NETWORK ENERGY CONSUMPTION

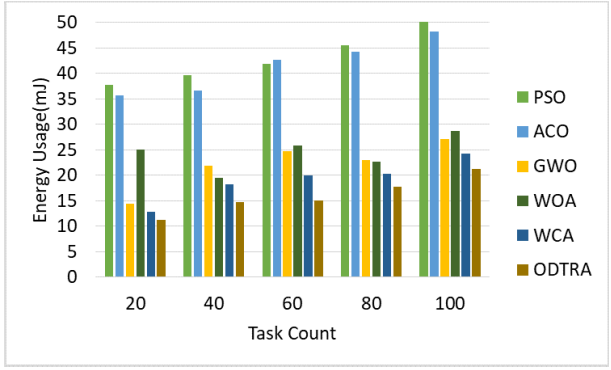
A key variable in the functional batteries and performance of Internet of Things (IoT) devices is energy, an abundant resource on the network that supports them. Research into the EC connected with tasks of different sizes in the field of IoT is presented for the purpose of analysis of resultant graphs. On trivial tasks, as demonstrated in Figure 3a, the ODTRA approach frequently outperforms its peers like ACO, PSO, GWO, and WOA. In ODTRA, the EC data were significantly reduced irrespective of the task count, and the WCA subsequently followed the match. The design highlights how well the technique optimizes EC for trivial tasks, which is essential for devices with minimum power resources. If we look at small to medium-sized tasks in Figure 3b, ODTRA is additionally the most successful algorithm in EC. The ability of this algorithm to attain lower EC metrics for small to medium-sized tasks, despite the task count, constantly sets it ahead of similar algorithms. Reduced energy



(a) For Small Task



(b) For Medium Task



(c) For Large Task

FIGURE 3. Analysis of EC for different-sized tasks.

use during computation implies that ODTRA can optimize EC and extend the functional lifespan of IoT devices. Figure 3c, requiring significant tasks into consideration, additionally demonstrates how ODTRA executes higher in terms of EC. ODTRA achieves decreased levels of EC and is reliably superior to comparable algorithms as the task count increases. The present study deals with power-related issues in enormous-scale IoT deployments by illustrating how the methodology effectively uses energy resources for challenging computational functions.

The ODTRA method has been able to minimize energy on tasks that have various sizes compared to the aggregate analysis. For Internet of Things (IoT) devices, ODTRA increases EC by typically superior to other approaches in

the context of EC value. The consequences for boosting the energy utilization of IoT systems as a whole, reducing the amount of time for charging or replacement of batteries, and prolonging the life span of particular devices are substantial.

In addition, the findings demonstrate which EC correlates positively with task size, subject to whether the task has been assigned or implemented locally. Also, there is a corresponding increase in EC usage when the total number of tasks gets higher. The results of this study have been linked to the higher load of computation on the IoT device or the edge server, according to the specific circumstances.

The utilization of CPU resources is one of the primary sources of EC in local computation. However, EC evolves with tasks that are offloaded whenever data is sent from the IoT device to the edge server and then obtained and returned by the device–the overall EC advantages from these additional communication processes.

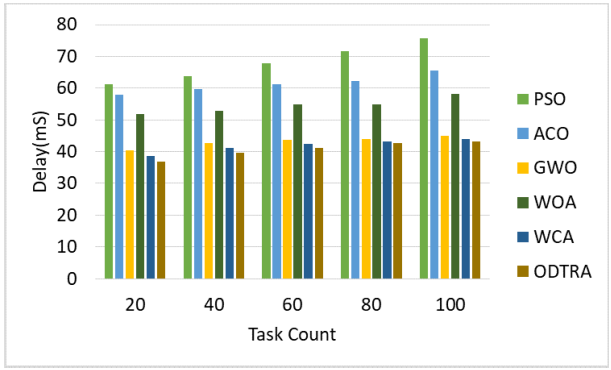
The significance of implementing task size as well as quantity of tasks into consideration when improving EC in IoT devices and edge servers is made clear by this research. In order to minimize EC and boost efficiency, it is vital to properly manage all of these variables and select optimal offloading methods.

C. ANALYSIS OF COMPUTATION LATENCY

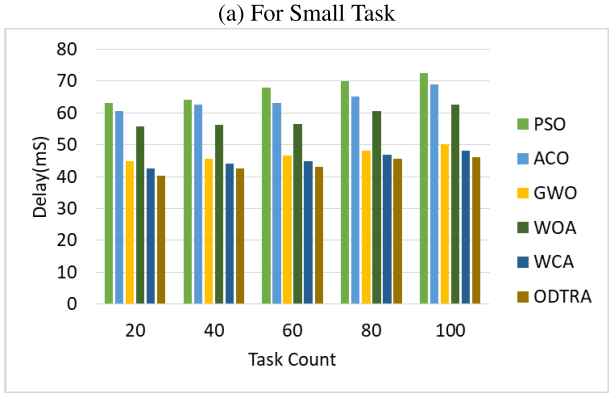
The total amount of time that expires between an IoT device receiving or analyzing a data input and then resulting in output is known as computation latency, which is additionally referred to as delay. With the increasing number of applications that function in real-time in the IoT environment, reducing delay is of greater significance than ever before for ensuring user requirements. The new ODTRA methodology has been investigated empirically by contrasting its findings against the outcomes of different methods previously in use. Finding the best method of action that would have resulted in a minimal impact on the computation of time for a definite task was the objective. The end result from graphical representations, marked ''Small,'' ''Medium,'' and ''Large,'' depending on the task sizes, illustrate the conclusions of this research.

The following graphs explain how methods perform in computation delay for Small, Medium, and Large tasks. For real-time IoT applications to function appropriately, computation latency–the time from input sensing to output–is significant. Reducing latency enhances the user experience. Evaluating the data in the tables shows that the recommended ODTRA method outperformed every other algorithm considered across all task types. In Figure 4a, ODTRA consistently performs better than WCA, along with additional algorithms in tasks with small latency values, irrespective of task count. Research shows that ODTRA proves helpful in reducing the negative impacts of computation latency on tasks of a smaller scale.

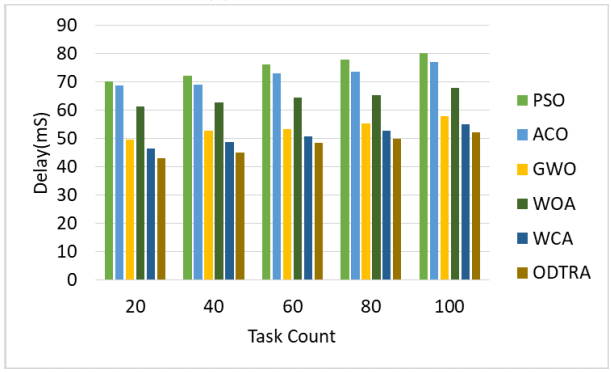
In medium tasks, Figure 4b follows the pattern that has been observed. ODTRA significantly reduces delay across all task counts, which is superior to other algorithms. The



(a) For Small Task



(b) For Medium Task



(c) For Large Task

FIGURE 4. Delay analysis for different task sizes.

algorithm's function to operate effectively and decrease computation latency, even for small to medium-sized tasks, is made clear here. Figure 4c indicates that when large tasks are factored in, ODTRA is once again the most appropriate selection. As the number of tasks gets higher, it continually attains decreased delay values when compared with other similar algorithms. That ODTRA is capable of minimizing computation latency for large tasks can be seen from this problem. In optimizing computation latency over different task sizes and counts, the research emphasizes the successful performance of the ODTRA method. By decreasing delays, ODTRA enhances the user experience and performance in cases where delays are essential to the success of an Internet of Things (IoT) application.

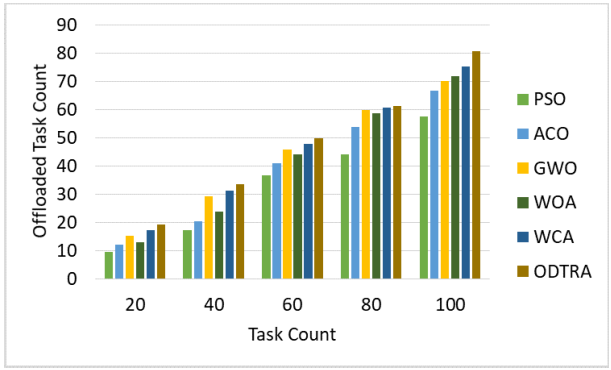
D. NUMBER OF TASK OFFLOADING

The selection of tasks must be optimal in order to succeed in efficient computation offloading. One of the most significant actions to take is to develop a balance between offloading tasks to the edge server and performing them locally on emergency devices. The server may become overloaded, resulting in delays if every task is offloaded. In addition, the device's battery energy may be consumed if all tasks are performed locally. When evaluating whether or not to offload, the computational workload is considered. Tasks that demand a higher amount of CPU cycles are more likely to be offloaded. Smaller tasks are better suited for local execution, EC, reducing bandwidth usage, and optimizing server utilization. The required CPU cycles and input data size influence the decision to offload a task. Local execution is preferred for tasks with larger input data sizes and a lower CPU cycle-to-data size ratio. This helps minimize the energy and time consumed during data transmission. Optimally, TO can improve energy efficiency, bandwidth utilization, and overall system performance by considering CPU cycle requirements, task size, and data transmission overhead.

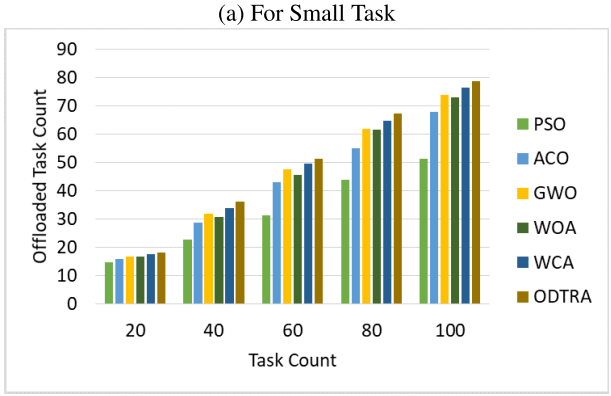
The illustration of the result graphs the distribution of TO across different execution scenarios at a specific point in time. The result provides insights into offloading tasks based on their sizes (Small, medium, and large) using different optimization algorithms. In Figure 5a, which focuses on small tasks, ODTRA consistently outperforms the other algorithms by offloading more tasks at all task counts. It indicates that ODTRA effectively identifies small tasks that require a more considerable number of CPU cycles for execution and selects them for offloading. Among the other algorithms, the WCA presented better results than the others; the GWO and WOA also demonstrate competitive performance, offloading a relatively higher number of tasks than PSO and ACO. Moving to Figure 5b, representing medium tasks, ODTRA excels in TO by offloading more tasks than the other algorithms that WCA follows. It highlights ODTRA's ability to identify medium tasks that meet the offloading criteria.

Similarly, GWO and WOA show promising results by many TOs, surpassing PSO and ACO. In Figure 5c, focusing on significant tasks, ODTRA again stands out by offloading more tasks than the other algorithms. It indicates that ODTRA effectively recognizes large tasks that require substantial computational resources and selects them for offloading. WCA also demonstrates competitive performance in large TO, outperforming other models.

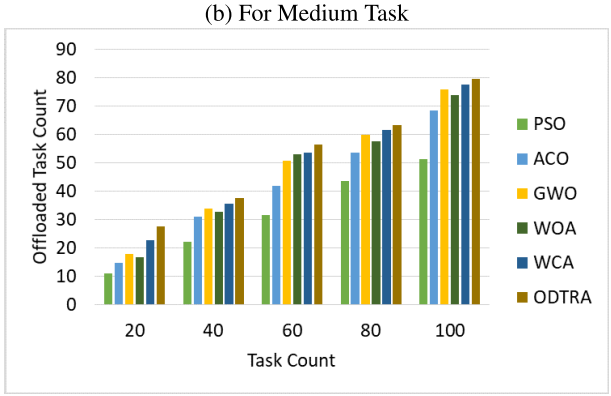
The analysis of the tables highlights the effectiveness of ODTRA in selecting TO, regardless of their sizes. By selecting tasks that require higher CPU cycles to execute, the ODTRA algorithm achieves reduced resource utilization and improved response time in the system. These findings emphasize the importance of selecting an appropriate optimization algorithm for offloading that considers the features and requirements of the tasks, ultimately achieving a balance between EC, response time, and server utilization. ODTRA's ability to select the optimal set of TO contributes to EC and reduced execution time for all available tasks.



(a) For Small Task



(b) For Medium Task



(c) For Large Task

FIGURE 5. Analysis for varied sizes of task.

E. ENERGY AND DELAY TRADEOFF ANALYSIS

Typically, small tasks require higher energy to offload to an edge server than large tasks, as they need additional communication overhead. Concurrent execution of multiple tasks on the Edge server can lead to higher response times and potential overloading, resulting in increased latency. The communication overhead raises EC when small tasks are offloaded to the Edge server. These factors are crucial in low-latency, low-battery applications. To address this, TO selection for offloading is vital, considering factors such

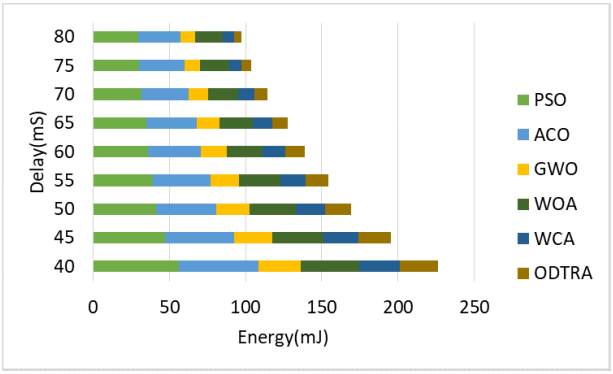


FIGURE 6. Energy-delay tradeoff for small task size.

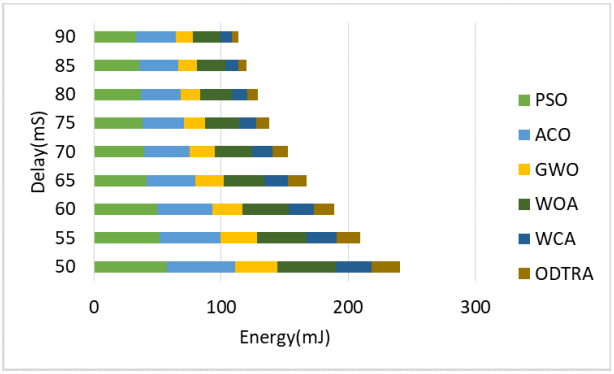


FIGURE 7. Energy-delay tradeoff for large task size.

as task size, communication overhead, and energy limitations of IoT devices. The aim is to minimize EC while meeting the response time requirements of delay-sensitive applications.

Figure 6 presents the EC values for different optimization algorithms when executing a small task size (1 KB) over an Edge server. The delay (milliseconds (ms)) measures response time. ODTRA consistently achieves minimal EC across all delay values compared to the other algorithms. As the delay increases, there is a general trend of decreasing EC for all algorithms. It is expected since a higher delay allows more time for task execution, resulting in minimal EC. Among the other algorithms, WCA exhibits maximum EC compared to other models. ODTRA achieves the most significant reduction in EC, providing a substantial improvement over the different algorithms.

Figure 7 presents the EC values for the same optimization algorithms when executing a larger task size (1000 KB) over an Edge server. Once again, ODTRA consistently achieves the minimum EC values across all delay values, outperforming the other algorithms. As the delay increases, there is a general trend of decreasing EC for all algorithms; WCA exhibits higher EC followed by GWO and WOA; further, ODTRA demonstrates the most significant reduction in EC, highlighting its effectiveness in OTO decisions for large task sizes. The resultant graphs underline the superiority of ODTRA in minimizing EC for small and large task sizes, leading to improved energy efficiency in offloading decisions. ODTRA consistently outperforms the other optimization algorithms, offering a better tradeoff between delay and EC. ODTRA is an excellent choice for delay-sensitive applications in IoT environments where minimizing EC is crucial to preserve limited battery energy and improve the system's overall efficiency.

F. ENERGY AND DELAY TRADEOFF ANALYSIS

The study evaluation was conducted to assess the cost performance of the proposed ODTRA approach and to analyze the effects of the number of UNs in order to minimize the overall system cost. The count of UNs is a crucial factor in general cost considerations, reflecting the TO competition in edge servers and cloud networks. Figure 8 presents the total cost results for different models, including the ODTRA

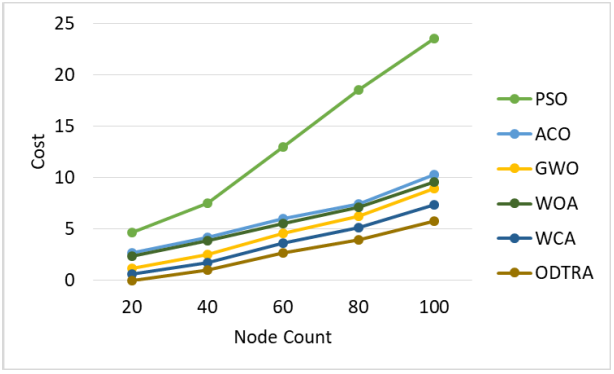


FIGURE 8. Cost analysis based on average computational resource requirements of tasks.

algorithm, compared to other models. The analysis focuses on the number of UNs in the system.

From the results, specific key observations can be made:

1) COMPARISON WITH OTHER MODELS

ODTRA consistently outperforms the other models regarding cost efficiency, demonstrating significantly lower total costs across all UN counts. The percentage differences in total costs between ODTRA and the other models highlight the substantial cost reductions achieved by ODTRA. ODTRA performs cost savings from around 21% to nearly 100% against these algorithms.

The analysis reveals ODTRA's cost efficiency, particularly at lower node counts, where it exhibits near-complete cost reductions compared to most models. ODTRA maintains a considerable cost advantage as the number of nodes increases, with savings from about 21% to over 75% relative to other algorithms, even in more significant network scenarios. This trend reflects ODTRA's scalability and ability to adapt to numerous operational scales.

2) IMPACT OF UN COUNT

To competition increases costs as UNs increase, increasing the total cost. As UNs increase, ODTRA's efficiency in managing TO competition and resource allocation lowers total cost.

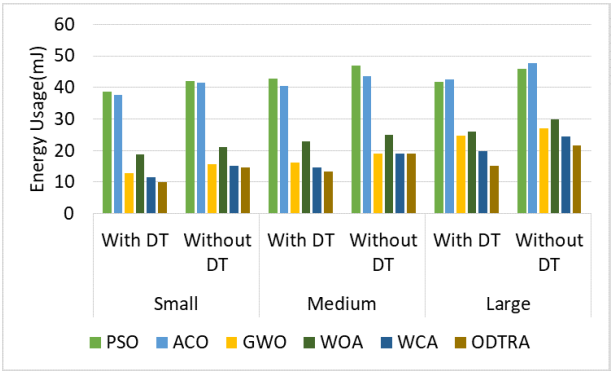


FIGURE 9. DT influence on TO time.

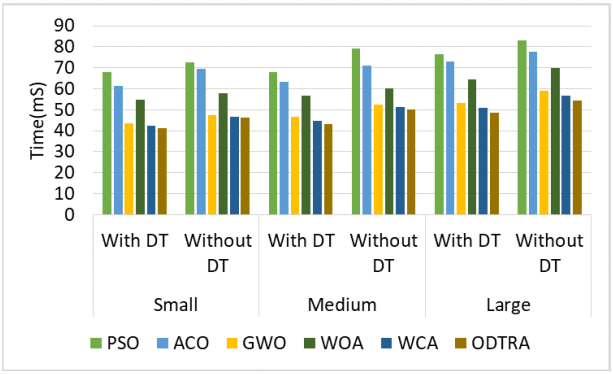


FIGURE 10. DT influence on energy usage.

The ODTRA algorithm reduces costs by 99.73% to 99.99%, making the offloading method more economically efficient. A cost-effective offloading solution for IIoT systems, ODTRA reduces costs and improves economic viability, according to the study.

G. ANALYSIS OF DT'S IMPACT ON OFFLOADING TIME AND ENERGY USAGE

The analysis compares the EC and TO time of numerous algorithms in IIoT systems with and without DT, with results presented in Figures 9 and 10, respectively. Energy efficiency and TO time are better than other algorithms, showing that ODTRA optimizes offloading decisions and reduces IIoT-EC. Without DT proof, it outperforms other models in TO time. These findings indicate that ODTRA can optimize IIoT offloading, promoting sustainable and efficient environments.

H. STATE-OF-THE-ART (SOTA) COMPARISON

In the conducted comparative analysis of various SOTA-TOM in the context of IIoT, ODTRA was evaluated against four other SOTA models across key performance metrics for small tasks, generating perceptive results. The details of the models used for comparison are presented in Table 6.

Regarding network EC (Figure 11), ODTRA consistently exhibited superior energy efficiency, maintaining the lowest EC across all task counts. This starkly contrasted to Model 1, which generally recorded the highest EC, while Models 2, 3, and 4 demonstrated intermediate energy efficiencies.

TABLE 6. Analysis of the differences between the various SOTA-TOS.

Model	Key Features	Strengths	Limitations
Model 1 [46]	Intelligent TO with AI	High adaptability, security, low processing delay	It may not address challenges in heterogeneous edge environments
Model 2 [47]	AI-enhanced framework focusing on service accuracy	Balances delay and service accuracy	Might not cater to diverse computational capabilities
Model 3 [48]	TOM using Q-learning in MEC and D2D	Energy efficiency, reduced execution delays	Limited general applicability, focused on MEC and D2D
Model 4 [49]	Joint offloading and RA for ML tasks	Minimizes long-term system cost, considers inference accuracy	Complex implementation, high computational resource needs

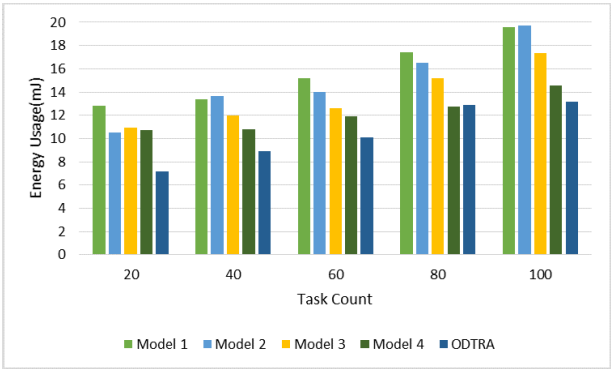


FIGURE 11. Task count vs. energy.

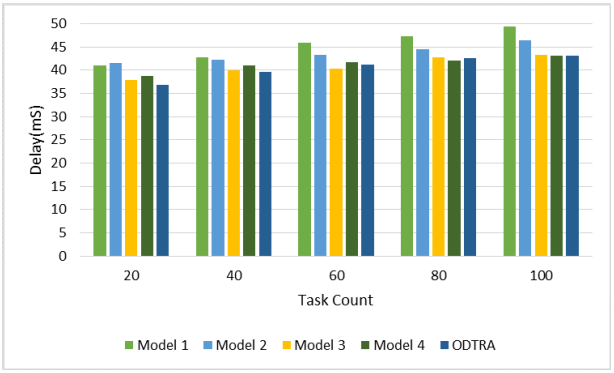


FIGURE 12. Task count vs delay.

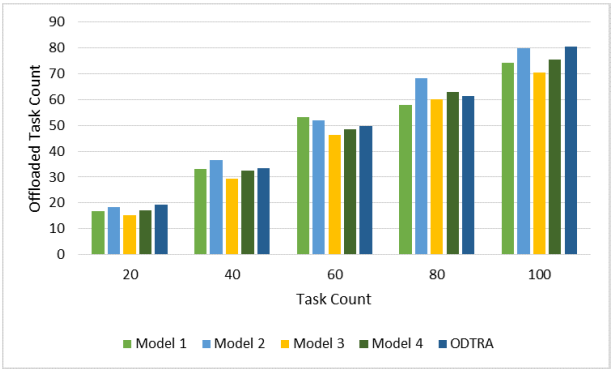


FIGURE 13. Number of TO.

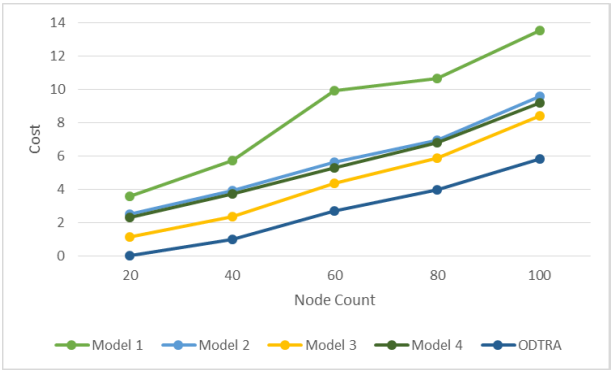


FIGURE 14. Influence of node count on cost.

When analyzing computation latency, ODTRA excelled in reducing processing delays (Figure 12), especially at lower task counts, and maintained competitive performance even as the task count increased. This efficiency in latency reduction was critical, particularly when compared to Models 1 and 2, which exhibited increasing delays with higher task counts, hinting at potential scalability challenges.

Regarding the number of TO (Figure 13), a key metric in assessing offloading efficiency, ODTRA proved robust capabilities, outperforming the other models, especially at higher task counts. While Models 2 and 4 showed competitive offloading performance, they did not match the efficiency of ODTRA. Models 1 and 3 lagged in this regard, indicating lesser offloading efficiency.

Perhaps most notably, ODTRA demonstrated significant cost advantages regarding the effect of User Nodes (UNs) on system costs (Figure 14). It consistently incurred the lowest system costs across various node counts, starkly contrasting with Model 1, which tended to be the costliest. Models 2 and 4 offered moderate cost performances, while Model 3, despite showing some cost efficiency, did not approach the extent of ODTRA's cost-effectiveness.

Overall, the comparative analysis highlighted ODTRA's comprehensive superiority across all evaluated metrics, including energy efficiency, latency reduction, TO efficiency, and cost-effectiveness. This positions ODTRA as a potent model in IIoT environments, adept at addressing the multi-faceted demands of high energy efficiency, rapid processing, effective TO, and optimized system costs.

VI. CONCLUSION AND FUTURE WORK

Task Offloading (TO) is crucial in managing the high volume of data generated in IIoT environments. Transferring resource-intensive tasks to remote cloud servers reduces the burden on Edge Devices (ED), mitigates resource constraints, and minimizes latency. The proposed model leverages Digital Twins (DT) as the virtual counterparts of IIoT systems, enabling real-time monitoring, predictive analysis, proactive decision-making, and efficient resource management. The ODTRA algorithm, designed for TO in IIoT with DT, optimizes performance by considering server capacity, bandwidth constraints, and device Energy Consumption (EC). It explores the solution space iteratively using the WCM and PRLS algorithms, generating high-quality solutions to minimize Task Execution Time (TET) and improve system efficiency. The model offers a viable solution for OTO performance with the potential for practical implementation in real-world IIoT scenarios. The integration of DT enhanced its applicability by providing real-time monitoring and informed decision-making. This innovative approach addresses the challenges of TO in IIoT, presenting opportunities to enhance system performance, reduce latency, improve energy efficiency, and meet quality service requirements. As the IIoT evolves, innovative solutions will be crucial to realizing its full potential and driving advancements in industrial sectors.

DECLARATION

Funding–Not Applicable
Conflicts of Interest/Competing Interests–Not applicable
Availability of Data and Material–Not applicable
Code Availability–Not Applicable

REFERENCES

[1] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, "Edge computing in industrial Internet of Things: Architecture, advances and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2462–2488, 4th Quart., 2020.

[2] H. Khayyam, B. Javadi, M. Jalili, and R. N. Jazar, "Artificial intelligence and Internet of Things for autonomous vehicles," in *Nonlinear Approaches in Engineering Applications: Automotive Applications of Engineering Problems*, 2020, pp. 39–68.

[3] A. S. Kumar and E. Iyer, "An industrial IoT in engineering and manufacturing industries—Benefits and challenges," *Int. J. Mech. Prod. Eng. Res. Develop.*, vol. 9, no. 2, pp. 151–160, 2019.

[4] S. Latif, Z. Idrees, Z. E. Huma, and J. Ahmad, "Blockchain technology for the Industrial Internet of Things: A comprehensive survey on security challenges, architectures, applications, and future research directions," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 11, Nov. 2021, Art. no. e4337.

[5] E. O'Connell, D. Moore, and T. Newe, "Challenges associated with implementing 5G in manufacturing," *Telecom*, vol. 1, no. 1, pp. 48–67, Jun. 2020.

[6] K. Wójcicki, M. Biegańska, B. Paliwoda, and J. Górna, "Internet of Things in industry: Research profiling, application, challenges and opportunities—A review," *Energies*, vol. 15, no. 5, p. 1806, Feb. 2022.

[7] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, A. I. A. Ahmed, M. Imran, and A. V. Vasilakos, "The role of big data analytics in Internet of Things," *Comput. Netw.*, vol. 129, pp. 459–471, Dec. 2017.

[8] C. Jiang, X. Cheng, H. Gao, X. Zhou, and J. Wan, "Toward computation offloading in edge computing: A survey," *IEEE Access*, vol. 7, pp. 131543–131558, 2019.

[9] A. Hazra, M. Adhikari, T. Amgoth, and S. N. Srirama, "A comprehensive survey on interoperability for IIoT: Taxonomy, standards, and future directions," *ACM Comput. Surv.*, vol. 55, no. 1, pp. 1–35, Jan. 2023.

[10] A. Rahman, J. Jin, A. Rahman, A. Cricenti, M. Afrin, and Y.-N. Dong, "Energy-efficient optimal task offloading in cloud networked multi-robot systems," *Comput. Netw.*, vol. 160, pp. 11–32, Sep. 2019.

[11] N. A. Abu-Taleb, F. Hasan Abdulrazzak, A. T. Zahary, and A. M. Al-Mqdashi, "Offloading decision making in mobile edge computing: A survey," in *Proc. 2nd Int. Conf. Emerg. Smart Technol. Appl. (eSmarTA)*, Oct. 2022, pp. 1–8.

[12] S. Mihai, M. Yaqoob, D. V. Hung, W. Davis, P. Towakel, M. Raza, M. Karamanoglu, B. Barn, D. Shetve, R. V. Prasad, H. Venkataraman, R. Trestian, and H. X. Nguyen, "Digital twins: A survey on enabling technologies, challenges, trends and future prospects," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2255–2291, 4th Quart., 2022.

[13] Z. Zhou, Z. Jia, H. Liao, W. Lu, S. Mumtaz, M. Guizani, and M. Tariq, "Secure and latency-aware digital twin assisted resource scheduling for 5G edge computing-empowered distribution grids," *IEEE Trans. Ind. Informat.*, vol. 18, no. 7, pp. 4933–4943, Jul. 2022.

[14] V. Sharma and A. K. Tripathi, "A systematic review of meta-heuristic algorithms in IoT based application," *Array*, vol. 14, Jul. 2022, Art. no. 100164.

[15] O. Zedadra, A. Guerrieri, N. Jouandeau, G. Spezzano, H. Seridi, and G. Fortino, "Swarm intelligence-based algorithms within IoT-based systems: A review," *J. Parallel Distrib. Comput.*, vol. 122, pp. 173–187, Dec. 2018.

[16] E. VanDerHorn and S. Mahadevan, "Digital twin: Generalization, characterization and implementation," *Decis. Support Syst.*, vol. 145, Jun. 2021, Art. no. 113524.

[17] C. Zhang, G. Zhou, H. Li, and Y. Cao, "Manufacturing blockchain of things for the configuration of a data- and knowledge-driven digital twin manufacturing cell," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11884–11894, Dec. 2020.

[18] G. Mylonas, A. Kalogeras, G. Kalogeras, C. Anagnostopoulos, C. Alexakos, and L. Muñoz, "Digital twins from smart manufacturing to smart cities: A survey," *IEEE Access*, vol. 9, pp. 143222–143249, 2021.

[19] F. K. Moghadam, G. F. D. S. Rebouças, and A. R. Nejad, "Digital twin modeling for predictive maintenance of gearboxes in floating offshore wind turbine drivetrains," *Forschung Ingenieurwesen*, vol. 85, no. 2, pp. 273–286, Jun. 2021.

[20] R. Chen, H. Shen, and Y. Lai, "A metaheuristic optimization algorithm for energy efficiency in digital twins," *Internet Things Cyber-Phys. Syst.*, vol. 2, pp. 159–169, Aug. 2022.

[21] B. He and K.-J. Bai, "Digital twin-based sustainable intelligent manufacturing: A review," *Adv. Manuf.*, vol. 9, no. 1, pp. 1–21, Mar. 2021.

[22] R. He, G. Chen, C. Dong, S. Sun, and X. Shen, "Data-driven digital twin technology for optimized control in process systems," *ISA Trans.*, vol. 95, pp. 221–234, Dec. 2019.

[23] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Y. C. Nee, "Enabling technologies and tools for digital twin," *J. Manuf. Syst.*, vol. 58, pp. 3–21, Jan. 2021.

[24] A. Madni, C. Madni, and S. Lucero, "Leveraging digital twin technology in model-based systems engineering," *Systems*, vol. 7, no. 1, p. 7, Jan. 2019.

[25] C.-H. Hsiao and W.-P. Lee, "OPIIoT: Design and implementation of an open communication protocol platform for industrial Internet of Things," *Internet Things*, vol. 16, Dec. 2021, Art. no. 100441.

[26] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multi-agent deep reinforcement learning for vehicular edge computing and networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1405–1413, Feb. 2022.

[27] D. Van Huynh, V.-D. Nguyen, S. R. Khosravirad, V. Sharma, O. A. Dobre, H. Shin, and T. Q. Duong, "URLLC edge networks with joint optimal user association, task offloading and resource allocation: A digital twin approach," *IEEE Trans. Commun.*, vol. 70, no. 11, pp. 7669–7682, Nov. 2022.

[28] X. Xu, B. Shen, S. Ding, G. Srivastava, M. Bilal, M. R. Khosravi, V. G. Menon, M. A. Jan, and M. Wang, "Service offloading with deep Q-network for digital twinning-empowered Internet of Vehicles in edge computing," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1414–1423, Feb. 2022.

[29] B. Fan, Y. Wu, Z. He, Y. Chen, T. Q. S. Quek, and C.-Z. Xu, "Digital twin empowered mobile edge computing for intelligent vehicular lane-changing," *IEEE Netw.*, vol. 35, no. 6, pp. 194–201, Nov. 2021.

[30] T. Do-Duy, D. van Huynh, O. A. Dobre, B. Canberk, and T. Q. Duong, "Digital twin-aided intelligent offloading with edge selection in mobile edge computing," *IEEE Wireless Commun. Lett.*, vol. 11, no. 4, pp. 806–810, Apr. 2022.

[31] T. Liu, L. Tang, W. Wang, Q. Chen, and X. Zeng, "Digital-twin-assisted task offloading based on edge collaboration in the digital twin edge network," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1427–1444, Jan. 2022.

[32] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning for digital twin edge networks in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5709–5718, Aug. 2021.

[33] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12240–12251, Oct. 2020.

[34] W. Sun, P. Wang, N. Xu, G. Wang, and Y. Zhang, "Dynamic digital twin and distributed incentives for resource allocation in aerial-assisted Internet of Vehicles," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5839–5852, Apr. 2022.

[35] B. Li, Y. Liu, L. Tan, H. Pan, and Y. Zhang, "Digital twin assisted task offloading for aerial edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 10863–10877, Oct. 2022.

[36] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2276–2288, Feb. 2021.

[37] T. Alfakih, M. M. Hassan, and M. Al-Razgan, "Multi-objective accelerated particle swarm optimization with dynamic programing technique for resource allocation in mobile edge computing," *IEEE Access*, vol. 9, pp. 167503–167520, 2021.

[38] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Minimizing the delay and cost of computation offloading for vehicular edge computing," *IEEE Trans. Services Comput.*, vol. 15, no. 5, pp. 2897–2909, Sep. 2022.

[39] Y. Dai, D. Xu, S. Maharjan, G. Qiao, and Y. Zhang, "Artificial intelligence empowered edge computing and caching for Internet of Vehicles," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 12–18, Jun. 2019.

[40] S. Subbaraj, R. Thiyagarajan, and M. Rengaraj, "A smart fog computing based real-time secure resource allocation and scheduling strategy using multi-objective crow search algorithm," *J. Ambient Intell. Humanized Comput.*, vol. 14, no. 2, pp. 1003–1015, Feb. 2023.

[41] J. Liu, C. Liu, B. Wang, G. Gao, and S. Wang, "Optimized task allocation for IoT application in mobile-edge computing," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 10370–10381, Jul. 2022.

[42] M. V. Jahan, M. Dashtaki, and M. Dashtaki, "Water cycle algorithm improvement for solving job shop scheduling problem," in *Proc. Int. Congr. Technol., Commun. Knowl. (ICTCK)*, Nov. 2015, pp. 576–581.

[43] S. K. Nayak, C. S. Panda, and S. K. Padhy, "Efficient multiprocessor scheduling using water cycle algorithm," *Soft Comput. Appl.*, pp. 131–147, Jan. 2018.

[44] N. Mahdavi-Nasab, M. A. Ardakan, and M. Mohammadi, "Water cycle algorithm for solving the reliability-redundancy allocation problem with a choice of redundancy strategies," *Commun. Statist. Theory Methods*, vol. 49, no. 11, pp. 2728–2748, Jun. 2020.

[45] K. Gao, P. Duan, R. Su, and J. Li, "Bi-objective water cycle algorithm for solving remanufacturing rescheduling problem," in *Proc. Asia–Pacific Conf. Simulated Evol. Learn.*, Springer, 2017, pp. 671–683.

[46] K. Ma, Z. Li, P. Liu, J. Yang, Y. Geng, B. Yang, and X. Guan, "Reliability-constrained throughput optimization of industrial wireless sensor networks with energy harvesting relay," *IEEE Internet Things J.*, vol. 8, no. 17, pp. 13343–13354, Sep. 2021.

[47] G. Liu, "Data collection in MI-assisted wireless powered underground sensor networks: Directions, recent advances, and challenges," *IEEE Commun. Mag.*, vol. 59, no. 4, pp. 132–138, Apr. 2021.

[48] Y. Jiang and X. Li, "Broadband cancellation method in an adaptive co-site interference cancellation system," *Int. J. Electron.*, vol. 109, no. 5, pp. 854–874, May 2022.

[49] C. Zheng, Y. An, Z. Wang, X. Qin, B. Eynard, M. Bricogne, J. Le Duigou, and Y. Zhang, "Knowledge-based engineering approach for defining robotic manufacturing system architectures," *Int. J. Prod. Res.*, vol. 61, no. 5, pp. 1436–1454, Mar. 2023.

[50] G. Sun, D. Liao, D. Zhao, Z. Xu, and H. Yu, "Live migration for multiple correlated virtual machines in cloud-based data centers," *IEEE Trans. Services Comput.*, vol. 11, no. 2, pp. 279–291, Mar. 2018.

[51] G. Sun, Z. Xu, H. Yu, and V. Chang, "Dynamic network function provisioning to enable network in box for industrial applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 7155–7164, Oct. 2021.

[52] Y. Xu, E. Wang, Y. Yang, and Y. Chang, "A unified collaborative representation learning for neural-network based recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5126–5139, Nov. 2022.

[53] W. Zheng, S. Lu, Z. Cai, R. Wang, L. Wang, and L. Yin, "PAL-BERT: An improved question answering model," *Comput. Model. Eng. Sci.*, vol. 139, no. 3, pp. 2729–2745, 2024.

[54] M. Yang, Y. Wang, C. Wang, Y. Liang, S. Yang, L. Wang, and S. Wang, "Digital twin-driven industrialization development of underwater gliders," *IEEE Trans. Ind. Informat.*, 2023.

[55] B. Cheng, M. Wang, S. Zhao, Z. Zhai, D. Zhu, and J. Chen, "Situation-aware dynamic service coordination in an IoT environment," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2082–2095, Aug. 2017.

[56] X. Yang, X. Wang, S. Wang, K. Wang, and M. B. Sial, "Finite-time adaptive dynamic surface synchronization control for dual-motor servo systems with backlash and time-varying uncertainties," *ISA Trans.*, vol. 137, pp. 248–262, Jun. 2023.

[57] X. Yang, X. Wang, S. Wang, and V. Puig, "Switching-based adaptive fault-tolerant control for uncertain nonlinear systems against actuator and sensor faults," *J. Franklin Inst.*, vol. 360, no. 16, pp. 11462–11488, Nov. 2023.

[58] W. Dai, X. Zhou, D. Li, S. Zhu, and X. Wang, "Hybrid parallel stochastic configuration networks for industrial data analytics," *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2331–2341, Apr. 2022.

[59] X. Liu, S. Lou, and W. Dai, "Further results on 'system identification of nonlinear state-space models,'" *Automatica*, vol. 148, Feb. 2023, Art. no. 110760.

[60] L. Li and L. Yao, "Fault tolerant control of fuzzy stochastic distribution systems with packet dropout and time delay," *IEEE Trans. Autom. Sci. Eng.*, 2004.

[61] Y. Xiao and A. Konak, "The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion," *Transp. Res. E, Logistics Transp. Rev.*, vol. 88, pp. 146–166, Apr. 2016.

[62] W. Zheng, G. Gong, J. Tian, S. Lu, R. Wang, Z. Yin, X. Li, and L. Yin, "Design of a modified transformer architecture based on relative position coding," *Int. J. Comput. Intell. Syst.*, vol. 16, no. 1, p. 168, Oct. 2023.

[63] M. K. A. Jannat, M. S. Islam, S.-H. Yang, and H. Liu, "Efficient Wi-Fi-based human activity recognition using adaptive antenna elimination," *IEEE Access*, vol. 11, pp. 105440–105454, 2023.

[64] Z. Jiang and C. Xu, "Disrupting the technology innovation efficiency of manufacturing enterprises through digital technology promotion: An evidence of 5G technology construction in China," *IEEE Trans. Eng. Manag.*, 2023.

[65] R. Guo, H. Liu, and D. Liu, "When deep learning-based soft sensors encounter reliability challenges: A practical knowledge-guided adversarial attack and its defense," *IEEE Trans. Ind. Informat.*, vol. 20, no. 2, pp. 2702–2714, Feb. 2024.

[66] S. Li, J. Chen, W. Peng, X. Shi, and W. Bu, "A vehicle detection method based on disparity segmentation," *Multimedia Tools Appl.*, vol. 82, no. 13, pp. 19643–19655, May 2023.

[67] B. Chen, J. Hu, Y. Zhao, and B. K. Ghosh, "Finite-time observer based tracking control of uncertain heterogeneous underwater vehicles using adaptive sliding mode approach," *Neurocomputing*, vol. 481, pp. 322–332, Apr. 2022.

[68] J. Shen, H. Sheng, S. Wang, R. Cong, D. Yang, and Y. Zhang, "Blockchain-based distributed multiagent reinforcement learning for collaborative multiobject tracking framework," *IEEE Trans. Comput.*, vol. 73, no. 3, pp. 778–788, Mar. 2024.

[69] Z. Fang, J. Wang, J. Liang, Y. Yan, D. Pi, H. Zhang, and G. Yin, "Authority allocation strategy for shared steering control considering human-machine mutual trust level," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 1, pp. 2002–2015, Jan. 2024.

[70] Z. Xiao, J. Shu, H. Jiang, J. C. S. Lui, G. Min, J. Liu, and S. Dustdar, "Multi-objective parallel task offloading and content caching in D2D-aided MEC networks," *IEEE Trans. Mobile Comput.*, 2022.

[71] X. Dai, Z. Xiao, H. Jiang, M. Alazab, J. C. S. Lui, S. Dustdar, and J. Liu, "Task co-offloading for D2D-assisted mobile edge computing in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 480–490, Jan. 2023.

[72] Z. Xiao, J. Shu, H. Jiang, G. Min, H. Chen, and Z. Han, "Perception task offloading with collaborative computation for autonomous driving," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 457–473, Feb. 2023.

[73] Y. Chen, L. Zhu, Z. Hu, S. Chen, and X. Zheng, "Risk propagation in multilayer heterogeneous network of coupled system of large engineering project," *J. Manage. Eng.*, vol. 38, no. 3, May 2022, Art. no. 04022003.

[74] Q. Wang, J. Hu, Y. Wu, and Y. Zhao, "Output synchronization of wide-area heterogeneous multi-agent systems over intermittent clustered networks," *Inf. Sci.*, vol. 619, pp. 263–275, Jan. 2023.

[75] H. Shi, Z. Song, X. Bai, Y. Hu, T. Li, and K. Zhang, "A novel digital twin model for dynamical updating and real-time mapping of local defect extension in rolling bearings," *Mech. Syst. Signal Process.*, vol. 193, Jun. 2023, Art. no. 110255.

[76] Y. Ding, W. Zhang, X. Zhou, Q. Liao, Q. Luo, and L. M. Ni, "FraudTrip: Taxi fraudulent trip detection from corresponding trajectories," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12505–12517, Aug. 2021.

[77] J. Qu, B. Mao, Z. Li, Y. Xu, K. Zhou, X. Cao, Q. Fan, M. Xu, B. Liang, H. Liu, X. Wang, and X. Wang, "Recent progress in advanced tactile sensing technologies for soft grippers," *Adv. Funct. Mater.*, vol. 33, no. 41, Oct. 2023, Art. no. 2306249.

[78] J. Qu, Q. Yuan, Z. Li, Z. Wang, F. Xu, Q. Fan, M. Zhang, X. Qian, X. Wang, X. Wang, and M. Xu, "All-in-one strain-triboelectric sensors based on environment-friendly ionic hydrogel for wearable sensing and underwater soft robotic grasping," *Nano Energy*, vol. 111, Jun. 2023, Art. no. 108387.

[79] T. Li, H. Shi, X. Bai, K. Zhang, and G. Bin, "Early performance degradation of ceramic bearings by a twin-driven model," *Mech. Syst. Signal Process.*, vol. 204, Dec. 2023, Art. no. 110826.

[80] J. Mou, K. Gao, P. Duan, J. Li, A. Garg, and R. Sharma, "A machine learning approach for energy-efficient intelligent transportation scheduling problem in a real-world dynamic circumstances," *IEEE Trans. Intell. Transp. Syst.*, 2022.

[81] J. Wu, J. Zhu, J. Zhang, P. Dang, W. Li, Y. Guo, L. Fu, J. Lai, J. You, Y. Xie, and C. Liang, "A dynamic holographic modelling method of digital twin scenes for bridge construction," *Int. J. Digit. Earth*, vol. 16, no. 1, pp. 2404–2425, Oct. 2023.

[82] H. Sheng, S. Wang, H. Chen, D. Yang, Y. Huang, J. Shen, and W. Ke, "Discriminative feature learning with co-occurrence attention network for vehicle ReID," *IEEE Trans. Circuits Syst. Video Technol.*, 2024.

[83] B. Cao, Z. Li, X. Liu, Z. Lv, and H. He, "Mobility-aware multiobjective task offloading for vehicular edge computing in digital twin environment," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3046–3055, Oct. 2023.

[84] J. Luo, C. Zhao, Q. Chen, and G. Li, "Using deep belief network to construct the agricultural information system based on Internet of Things," *J. Supercomput.*, vol. 78, no. 1, pp. 379–405, Jan. 2022.

[85] Y. Xie, X.-Y. Wang, Z.-J. Shen, Y.-H. Sheng, and G.-X. Wu, "A two-stage estimation of distribution algorithm with heuristics for energy-aware cloud workflow scheduling," *IEEE Trans. Services Comput.*, vol. 16, no. 6, pp. 4183–4197, Nov. 2023.

[86] C. Liu, T. Wu, Z. Li, T. Ma, and J. Huang, "Robust online tensor completion for IoT streaming data recovery," *IEEE Trans. Neural Netw. Learn. Syst.*, 2022.

[87] B. Cao, X. Wang, W. Zhang, H. Song, and Z. Lv, "A many-objective optimization model of industrial Internet of Things based on private blockchain," *IEEE Netw.*, vol. 34, no. 5, pp. 78–83, Sep. 2020.

[88] B. Cao, J. Zhao, Y. Gu, S. Fan, and P. Yang, "Security-aware industrial wireless sensor network deployment optimization," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5309–5316, Aug. 2020.

[89] B. Cao, J. Zhao, P. Yang, Y. Gu, K. Muhammad, J. J. P. C. Rodrigues, and V. H. C. de Albuquerque, "Multiobjective 3-D topology optimization of next-generation wireless data center network," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3597–3605, May 2020.

[90] Z. Chen and L. Gao, "CURSOR: Configuration update synthesis using order rules," Tech. Rep., 2023.

[91] J. Lu and C. Osorio, "A probabilistic traffic-theoretic network loading model suitable for large-scale network analysis," *Transp. Sci.*, vol. 52, no. 6, pp. 1509–1530, Dec. 2018.

[92] T. Lyu, H. Xu, L. Zhang, and Z. Han, "Source selection and resource allocation in wireless-powered relay networks: An adaptive dynamic programming-based approach," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 8973–8988, Mar. 2024.
[93] J. Chen, Q. Wang, W. Peng, H. Xu, X. Li, and W. Xu, "Disparity-based multiscale fusion network for transportation detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18855–18863, Oct. 2022.
[94] J. Chen, Q. Wang, H. H. Cheng, W. Peng, and W. Xu, "A review of vision-based traffic semantic understanding in ITSs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 19954–19979, Nov. 2022.
[95] K. Li, L. Ji, S. Yang, H. Li, and X. Liao, "Couple-group consensus of cooperative-competitive heterogeneous multiagent systems: A fully distributed event-triggered and pinning control method," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 4907–4915, Jun. 2022.
[96] H. Guo, J. Liu, and J. Lv, "Toward intelligent task offloading at the edge," *IEEE Netw.*, vol. 34, no. 2, pp. 128–134, Mar. 2020.
[97] W. Sun, J. Liu, and Y. Yue, "AI-enhanced offloading in edge computing: When machine learning meets industrial IoT," *IEEE Netw.*, vol. 33, no. 5, pp. 68–74, Sep. 2019.
[98] S. Koo and Y. Lim, "Optimal task offloading decision in IIoT enviornments using reinforcement learning," in *Proc. IEEE 3rd Eurasia Conf. IoT, Commun. Eng. (ECICE)*, Oct. 2021, pp. 86–89.
[99] W. Fan, S. Li, J. Liu, Y. Su, F. Wu, and Y. Liu, "Joint task offloading and resource allocation for accuracy-aware machine-learning-based IIoT applications," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3305–3321, Feb. 2023.
[100] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.*, vols. 110–111, pp. 151–166, Nov. 2012.

DHIVYA SWAMINATHAN received the B.E. and M.E. degrees from Annamalai University, Tamil Nadu, India, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree with Vellore Institute of Technology, Chennai, Tamil Nadu. She has published seven international journals and seven international conference papers. Her research interests include power system optimization, power distribution system analysis, and soft computing techniques. She received the Gold Medal in the UG Program and the Best Outgoing Student Award.

ARUL RAJAGOPALAN received the B.E., M.E., and Ph.D. degrees in electrical and electronics engineering from Annamalai University, in 1995, 1996, and 2016, respectively. He is a Faculty Member with the School of Electrical Engineering, Vellore Institute of Technology, Chennai. He has 24 years of teaching and research experience. He has published more than 60 international journals indexed in Scopus and Web of Science database. His research interests include power systems optimization, soft computing techniques, the IoT, and e-vehicles.

VENKATRAM NIDUMOLU (Senior Member, IEEE) has been a Pro-Vice Chancellor and a Professor with the ECE Department, K. L. University, for 25 years. He has published more than 100 research papers in international and national journals and conferences. He has also published Scopus and SCI publications of 37 with an H-index of eight. He is a reviewer of several international journals, including IEEE, Elsevier, Springer, Taylor & Wiley, and Francis. He served as a reviewer, a keynote speaker, and a TPC member for several international conferences. His research interests include deep learning, data science, and the IoT.

ROOBAEA ALROOBAEA received the bachelor's degree (Hons.) in computer science from King Abdulaziz University (KAU), Saudi Arabia, in 2008, and the master's degree in information systems and the Ph.D. degree in computer science from the University of East Anglia, U.K., in 2012 and 2016, respectively. He is an Associate Professor with the College of Computers and Information Technology, Taif University, Saudi Arabia. His research interests include human–computer interaction, software engineering, cloud computing, the Internet of Things, artificial intelligence, and machine learning.

HOSSAM KOTB received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the Faculty of Engineering, Alexandria University, Alexandria, Egypt, in 2009, 2013, and 2020, respectively. His Ph.D. research work was focused on the performance enhancement of renewable energy conversion systems. He is an Assistant Professor with the Electrical Power and Machines Department, Faculty of Engineering, Alexandria University. His research interests include power system analysis, electrical drives, modern control techniques, smart grids, optimization, electric vehicles, and renewable energy systems. He is an Associate Editor of *Alexandria Engineering Journal* (AEJ).

KAREEM M. ABORAS received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the Faculty of Engineering, Alexandria University, Alexandria, Egypt, in 2010, 2015, and 2019, respectively. His Ph.D. research work was focused on the performance enhancement of renewable energy conversion systems. He is an Assistant Professor with the Electrical Power and Machines Department, Faculty of Engineering, Alexandria University. His research interests include power electronics, control, drives, power systems, smart grids, microgrids, power quality, optimizations, electric vehicles, machine learning, modeling, fuel cells, HVDC, and renewable energy systems. He is a Reviewer of IEEE TRANSACTIONS ON ENERGY CONVERSION, *Electric Power Systems Research*, *Smart Science*, *Alexandria Engineering Journal*, IET, *Energy Reports*, IEEE ACCESS, *Cybernetics and Systems*, *Protection and Control of Modern Power Systems*, MDPI, *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *Cogent Engineering*, and Hindawi journals.

ALI ELRASHIDI is a Professor with the Electrical Engineering Department, University of Business and Technology (UBT). He was with academia for the past 23 years in various administrative and teaching roles, including the Associate Dean for Engineering, the College of Engineering Academic Consultant, and a Secretary of UBT Scientific Council. He has published over 65 research papers in national/international journals and conferences. He was a PI or a Co-PI of many funded research projects in the past ten years. Sponsors include ARDEC (United States), Kuwait College of Science and Technology, and King Abdul Aziz University. He received the Distinguished Assistant Professor of the Year Award from the University of Business and Technology, in 2015.