The Journal of Supercomputing (2023) 79:20076–20115
https://doi.org/10.1007/s11227-023-05387-w

Energy-efficient computation offloading using hybrid GA with PSO in internet of robotic things environment

Noha El Menbawy[1] · Hesham A. Ali[1,2] · Mohamed S. Saraya[1] · Amr M. T. Ali-Eldin[1] · Mohamed M. Abdelsalam[1]

Accepted: 5 May 2023 / Published online: 8 June 2023

Abstract

The Internet of Robotic Things (IoRT) is an integration between autonomous robots and the Internet of Things (IoT) based on smart connectivity. It's critical to have intelligent connectivity and excellent communication for IoRT integration with digital platforms in order to maintain real-time engagement based on efficient consumer power in new-generation IoRT apps. The proposed model will be utilized to determine the optimal way of task offloading for IoRT devices for reducing the amount of energy consumed in IoRT environment and achieving the task deadline constraints. The approach is implemented based on fog computing to reduce the communication overhead between edge devices and the cloud. To validate the efficacy of the proposed schema, an extensive statistical simulation was conducted and compared to other related works. The proposed schema is evaluated against the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Whale Optimization Algorithm (WOA), Artificial Bee Colony (ABC), Ant Lion Optimizer (ALO), Grey Wolf Optimizer (GWO), and Salp Swarm Algorithm to confirm its effectiveness. After 200 iterations, our proposed schema was found to be the most effective in reducing energy, achieving a reduction of 22.85%. This was followed closely by GA and ABC, which achieved reductions of 21.5%. ALO, WOA, PSO, and GWO were found to be less effective, achieving energy reductions of 19.94%, 17.21%, 16.35%, and 11.71%, respectively. The current analytical results prove the effectiveness of the suggested energy consumption optimization strategy. The experimental findings demonstrate that the suggested schema reduces the energy consumption of task requests more effectively than the current technological advances.

Keywords IoRT (Internet of Robotic Things) · Cloud robotics · Fog robotics · Energy optimization in IoRT · Task offloading

✉ Noha El Menbawy
nohahossam@mans.edu.eg

[1] Computer Engineering and Control Systems Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt

[2] Faculty of AI, Delta University for Science and Technology, Mansoura, Egypt

1 Introduction

IoRT is a modern scientific paradigm incorporating various concepts, such as Cloud Robotics (CR), Artificial Intelligence (AI), Machine Learning (ML), and The Internet of Things (IoT) [1]. IoT integration with robotics is mainly aimed at enhancing movements, sensing, monitoring, and autonomous behavior in robotic systems. The IoRT market rise is driven by industrial, e-commerce applications, healthcare, household appliances (personal robots), and vehicle [2]. Many IoRT applications have real-time service needs that will have devastating consequences if the delay in response surpasses the tolerated latency. Moreover, due to limited resources, such as low battery capacity and limited computation, these devices cannot operate for a long period [3]. Therefore, energy consumption is also an essential measure that must be taken into account. Accordingly, how to enhance processing efficiency in real-time for enormous quantities of data with minimizing energy consumption is a major challenge.

Recent studies have shown that CR stands for a merger of robots with cloud technologies in order to extend the capabilities of robots using offloading techniques [4]. The idea of computation offloading was explored in order to decrease energy usage and improve the performance of task processing. However, it is important to note that conventional central clouds are typically located in remote locations, far from their consumers [5]. In addition, cloud robots are frequently unable to operate numerous robots because of their latency, bandwidth, and network congestion restrictions. Hence, long transmissions from IoRT devices to cloud servers might create delays and add to the cost of transmission energy [6]. Therefore, fog computing was invented and constructed to overcome the constraints of cloud computing. Fog computing is a concept that stretches cloud and networking services to the network edge, nearer to terminal devices with minimal data processing latency and high mobility, which may decrease the load on the main network and maximize energy savings. Fog computing puts both computation and radio resources closer to IoRT devices, boosting scalability in both areas [6].

Energy efficiency is extremely essential from the future standpoint of information communication technology (ICT) [7]. There is therefore much effort to minimize energy consumption nowadays. Computation task offloading is believed to be the way to reduce energy consumption [8]. There are two types of computation offloading: Firstly, the binary processing offloading approach, which means that the processing of tasks may be performed either locally or at the edge; and secondly, the partial computation offloading strategy, which considers that users can divide the tasks such that some are performed locally, and others are offloaded to the edge [9]. However, partial offloading is preferred because it achieves higher energy savings and lower computation latency than binary offloading. Computation offloading consists of three main steps: (i) data transmission step, (ii) remote processing step, and (iii) response step [10]. Ignoring the third procedure because the amount of data needed to represent the computation results and send them back to the end user is generally significantly lower than the amount of data necessary to offload the task [11].

This research aims to devise a new technique to offload tasks to meet the needs of IoRT latency-sensitive systems with the optimal use of the resources available on the fog-cloud platform. The objective is to reduce the overall service time and decrease the consumed energy of the IoRT systems. The major contributions of this paper can be summarized as follows:

- Proposing a novel paradigm for the Internet of Robotic Things (IoRT) that is based on fog robotics, as opposed to the traditional approach of using cloud robotics. By leveraging the power of fog computing, we aim to enable robots to complete tasks within strict deadlines, while also ensuring energy efficiency and reducing the burden on the communication network.
- Proposing a partial computation offloading scheme that uses a mathematical model to find the optimal solution, taking into account features of the application such as CPU requirements, network requirements, latency sensitivity, and resource usage to reduce the energy consumed by latency-sensitive IoRT apps that are based on the fog-cloud framework. The scheme optimizes the offloading ratio and transmission rate to minimize energy consumption while ensuring that task deadlines are met.
- Proposing an Improved Hybrid Optimization technique based on GA Combined with PSO called (IHOGCP) to achieve efficient computation offloading. The proposed algorithm provides a trade-off between exploration and exploitation capabilities, which leads to better convergence and higher-quality solutions.
- Conducting simulations to evaluate the performance of the proposed algorithm, and comparing it with full offloading and other state-of-the-art techniques. The simulation results show a 22.85% improvement in overall energy consumption, indicating the superiority of our algorithm over other existing techniques.

This research is outlined as follows: Sect. 2 provides background, including information about IoT and IoRT technologies. Section 3 shows relevant literature pertaining to the study subject. Section 4 discusses the proposed approach for IoRT based on fog computing. Section 5 provides the system model and the problem formulation. The methodology is presented in Sect. 6. Section 7 presents the proposed algorithm and describes GA and PSO separately. Section 8 presents the experimental analysis and results. InSect. 9, the conclusion and future work are presented.

2 Preliminaries and background information

2.1 Overview of the internet of things

The IoT principle allows a vast number of "things" that can be uniquely addressed to communicate and transmit data via existing internet or network protocols like TCP/IP. It produces massive amounts of data and processes vastly different amounts of data, which have yet to be seen [12]. It seeks to make human society intelligent, convenient, and efficient with huge environmental and economic benefits. Processing, energy, bandwidth, and storage restrictions are common in IoT devices. The Internet

of Things might make human life easier, safer, and more intelligent. Many fields, including military, agriculture, manufacturing, healthcare, robotics, and nanotechnology, are benefiting from IoT's steady advancement [13]. Smart homes, smart cities, smart transportation and mobility, smart healthcare, smart factories, and smart manufacturing are just a few of the numerous applications available to users [14]. Table 1 shows the Internet of Things application domains and their properties [15, 16].

IoT technology provides a solid foundation for consumers to add intelligence to their current devices and connect them to the Internet to share information between devices. As theory evolves and contributes to the major technological developments in different fields of usage, new terminology has emerged, such as the Internet of Medical Things (IoMT), Internet of Nano Things (IoNT), Internet of Mobile Things (IoMBT), Internet of Cloud Things (IoCT), Internet of Autonomous Things (IoAT), Internet of Drone Things (IoDT), Industrial Internet of Things (IIoT), and Internet of Robotic Things (IoRT) and many more [17]. In recent years, the Internet of Robotic Things, which combines sensing devices and robotic systems, has gained prominence as a result of their widespread availability.

2.2 Overview of internet of robotic things

This section offers an introduction to the Internet of Robotic Things. It discusses the IoRT and its architecture.

2.2.1 Aims and motivation

Through IoRT technology, one can reduce the need for human intervention in several activities such as health care delivery, transportation, and marketing. This opens up a new world of opportunities for everything from military operations to economic applications [18]. The IoRT is classified according to the application categories in Fig. 1. It can be used in different zones as factory, social and outdoor applications. When it comes to situational and location awareness, the IoRT system has shown its value in a variety of different contexts, including decreasing power consumption while producing high-quality images, increasing system security, and even in clinical services and emergencies for providing medical supplies ahead of rescue teams. However, obstacles such as old network topologies, restricted bandwidth in Wireless Sensor Networks (WSNs), outdated system integrations, and battery life constraints prevent IoRT applications from realizing their full potential [19].

Several improvements have been brought about by robotic systems in different areas of human life. Many manufacturing industries are embracing robotics to conduct all kinds of complex, sensitive, and difficult activities, such as soldering, assembling items, inspecting products, wrapping, and many more. The integration of the smart world with independent agents (robots) has given rise to the Internet of Robotic Things. [20]. Smart world apps include smart homes, manufacturing, agriculture, buildings, and factories. The key aspect of these apps is the monitoring of conditions and operations within a specific control region. Agents can only

Table 1 Internet of things application domains and their properties

Properties	Smart home	Smart city	Smart transportation	Smart healthcare	Smart factory
Utilizers	Extremely few, home members	Large, policy makers, citizens	Large, general public	Very few	Few, policy makers
Network scale	Small	Medium	Large	Small	Medium/large
Energy	Chargeable batteries	Chargeable batteries, Energy harvesting	Chargeable batteries, Energy harvesting	Long term chargeable battery	Rechargeable battery, Energy harvesting
Internet accessibility	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, satellite communication	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, 3G, 4G LTE backbone
Required bandwidth	Little	Large	Moderate/large	Large	Large
Response time	Low	Medium/low	Low	Low	Medium/low

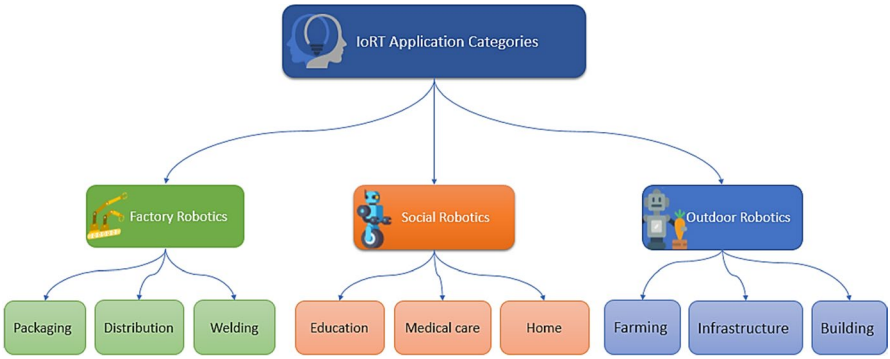


Fig. 1 Classification of IoRT according to application categories

perform well-defined tasks. These intelligent agents are known as robots and come in a variety of forms including service, mobile, and assistance robots [2]. The advent of intelligent agencies in the smart world completes the IoRT concept by merging the Smart world's functions and capabilities with those of robots. Robots are smart machines with sensors that can interpret real-world behavior by acquiring environmental data from sensors and taking appropriate action to resolve issues. The IoRT is a multi-robotic network that is smart, extremely effective, secure, and inexpensive. The IoRT refers to the collecting, analyzing, and utilization of sensor data from a number of sources using local and distributed data to monitor and control physical items. The integration of IoT and robotics in IoRT is mainly aimed to enhance movements, sensing, monitoring, and autonomous behavior in robotic systems.

CR is a term that refers to the use of cloud services and IoT technologies in the area of robotics [20]. This model is becoming more popular in scientific studies throughout the world because it makes it possible to use the robotic infrastructure in conjunction with IoT and the cloud. Therefore, Robotic capabilities can be enhanced by the use of high-performance cloud services at the backend thanks to CR, an interdisciplinary field combining distributed computing and robots. When it comes to cloud computing, robotics is a growing discipline that relies on cloud services and other technologies that allow robots to make use of current data centers' strong computational, storage, and communications capabilities connected via cloud infrastructures [17, 21]. IoRT shares significant characteristics with the IoT and CR, but it also has distinct differences. As a result, it has its own set of benefits and problems that must be met.

2.2.2 Internet of robotic things architecture

IoRT is an emerging discipline, and there have been many efforts to provide an IoRT architecture. The IoRT general architecture can be divided into five layers, as shown in Fig. 2 [17, 22, 23]. They can be described as Physical, Network,

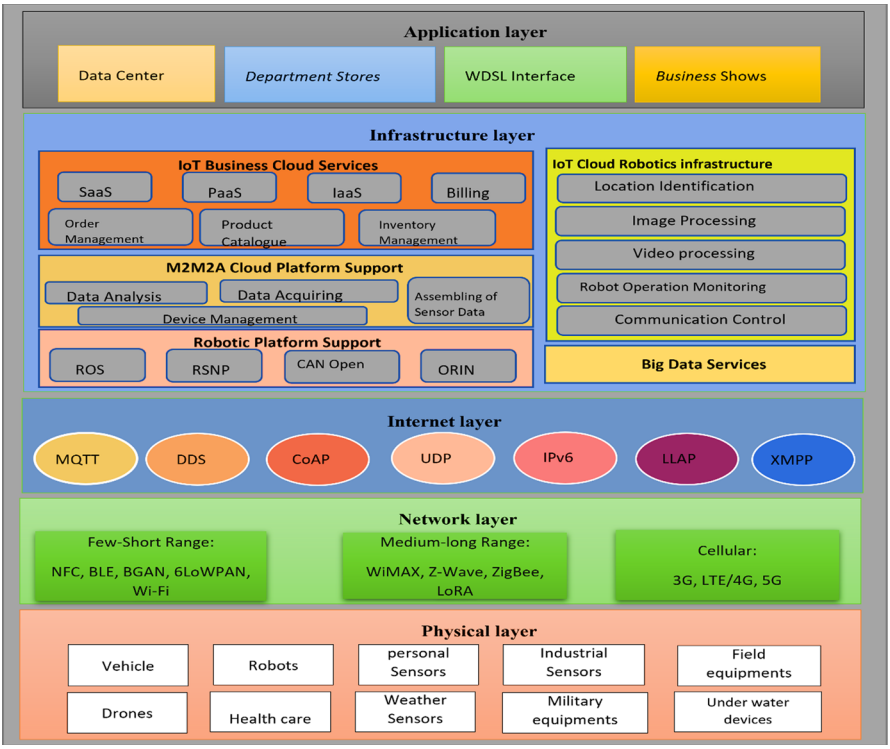


Fig. 2 Internet of robotic things general architecture

Internet, Robotic Infrastructure, and Application layers, respectively. The following section describes each of these layers.

1. Physical layer: The bottom tier in IoRT architecture is the physical layer. This layer is made up of various types of items such as vehicles, robots, home appliances, healthcare equipment, cell phones, sensors, planetary equipment, weather sensors, field equipment, underwater devices, and drones. A physical layer is mainly responsible for working in the environment with signal sensing, data gaining, and data transmission to the upper layer i.e., the Network tier [23].
2. Network Layer: Various network connectivity options are provided on the network tier to allow connectivity between sensors and robots, even for machine-to-machine interaction. In terms of cellular, Few-Short, and Moderate-Long ranges. Cellular connectivity includes 3G [24], LTE/4G, and 5G [25] which is allowed by this option. A few short-range communication technologies are used for easily communicating with robots close to each other, such as Wi-Fi, Bluetooth Low Energy (BLE) [26], 6LoWPAN [27], Broadband Global Area Network (BGAN) [28] and Near Field Communication (NFC) [29]. Moderate-Long range communication technologies such as the Worldwide Interoperability for Microwave Access

(WiMAX) [30], Z-Wave, ZigBee [31], and LoRA [32] have all been integrated for the simple operation of information transmission.

3. Internet Layer: This single option allows for device connectivity and access to information everywhere. This layer contains protocols that perform a variety of network functions. Taking into account IoRT's architecture, the Internet layer is perceived as the core of any communication. Since IoRT uses IoT, different IoT structured protocols like UDP, IPv6, DDS, CoAP, LLAP, MQTT, and XMPP are used [33].
4. Robotic Infrastructure Layer: This architectural portion of the IoT-based CR stack is renovated to become the most precious layer of all. This layer is composed of 5 components, including robotic platform support, Machine-to-Machine-to-Actuator (M2M2A) cloud platform support, Internet of Things business cloud services, Big Data services, and Internet of Things cloud robotics infrastructure. Each of them should be discussed as follows:

(a) Robotic Platform Support: This platform is used by scientists to create robots with rudimentary software abilities for doing experiments. It provides basic robotic service technologies like Robot Operating System (ROS), Robot Service Network Protocol (RSNP), Controller Area Network (CANopen), and Open Resource Interface for the Network (ORIN), etc.

(b) Cloud Platform Support for M2M2A: This component accomplishes many tasks such as data gathering, data analysis, device monitoring, and control. The Machine-to-Machine-to-Actuator model, which is suitable for sophisticated robotic devices, i.e., IoRT, is envisioned.

(c) Internet of Things business cloud services: By supporting enterprises and producers in lowering their total burden, IoRT will provide a platform upon which to build a variety of complicated business activities. All transactions under IoRT can be extended to various cloud computing models like SaaS, PaaS, and IaaS [34]. Both modular and packaged business-oriented APIs make e-commerce performance simpler.

(d) Big Data services: It provides incredibly valuable operations in areas of optimization, forecasting, marketing, and statistical analytics, Big Data offers extremely value-added services.

(e) IoT cloud robotics infrastructure: This part is designed to encompass services only. It allows robots to be equipped with a wide range of services, such as location identification, image and video processing, and communication management [20, 33].

5. Application Layer: This is the highest layer of IoRT architecture, it is designed to spread user experience by investigating a variety of robotics-related applications. When robotics is integrated with the Internet of Things, the cloud can be used in a range of applications, such as healthcare, infrastructure servicing, electronic commerce sites, retail stores, life-critical circumstances, data centers, Web Services Description Language interfaces, and others [23].

2.2.3 Internet of robotic things difficulties and problems

There are a number of unresolved difficulties related to IoRT as well as suggestions for future research guidelines and directions on various IoRT technology and application subjects. The computational problem, scalability, and handling of data generated by a billion devices are some of the quality of service (QoS) metrics that are mentioned in this section. In order to summarize the most significant issues related to the IoRT, Table 2 is provided [35, 36].

This research focuses on the energy efficiency of the computation offloading network, with the goal of reducing system energy consumption while meeting the latency requirements of the computing workloads. Both computational and communication energy are consumed in total energy consumption. Since IoRT is based on CR so, it suffers from a high latency response [4]. Real-time tasks are processed on the IoRT devices rather than offloaded to the cloud due to the high latency response of the cloud. In order to decrease the amount of energy consumed during processing real-time tasks on the mobile device, it is needed a new approach to perform these tasks with regard to their sensitivity [37]. Also, it is a very energy-consuming process to move massive data sets to a central cloud. Therefore, this study proposes using fog robotics instead of CR for achieving efficient-energy consumption.

Fog robotics is an expansion of "Cloud Robotics" that shares storage, computation, and resources between the cloud and the edge. The name "Fog Robotics" was coined by Gudi et al. (like Fog Computing) [38]. The Edge of the network is characterized by a limited number of heterogeneous resources, while the cloud may be considered an infinite pool of identical resources in remote centralized servers. The edge's resources include lightweight microservices, and networking devices such as gateways, routers, switches, and access points. The purpose is to leverage the available resources in both the Edge and the Cloud to improve latency, bandwidth, reliability, and energy efficiency [39]. Therefore, the study will move from centralization to decentralization using fog computing.

Cisco has introduced fog computing [40]. It is an expansion that is based on the cloud, and it is located much closer to the Internet of Robotic Things systems. It gives the edge of the network the ability to compute, communicate, store data, and provide services. Fog's distinguishing features include its closeness to end devices, its widespread geographic dispersion, and its locomotion-supporting. Fog computing has merits such as: reducing the amount of data that is sent to the centralized servers; delay reduction; reducing the bandwidth cost; increasing security; and enhancing services for remote locations [41]. Now, one can introduce a new definition of IoRT as it is a modern scientific paradigm incorporating various concepts including Fog Robotics (FR), Artificial Intelligence (AI), Machine learning (ML), and The Internet of Things (IoT).

3 Related work

Inconsistency of sensing equipment's limited computing resources with demanding real-time processing tasks results in huge amounts of energy use and inadequate performance. In recent years, computation offloading has been an important hotspot

Table 2 The key difficulties relating to the internet of robotic things

Problem	Definition	Solution
Latency issue	Real-time data analysis is critical for cloud robotic systems. Latency is an important consideration in such systems. It is the time required to obtain the relevant information or send the tasks to be processed remotely and receive the outcomes from a cloud-based robotics platform. Several elements are involved, including the local computing duration on the robot, the remote cloud computing duration, the size of the data, the duration it takes to transfer data, and the amount of time it takes for the network to respond	In contrast to a cloud situation, the fog tier in a conventional fog-cloud architecture decreases the system's total latency. The cloud and devices are separated by a layer of fog. Consequently, to reduce total network latency, the fog layer performs minor and urgent computations instead of the cloud layer
Heterogeneity issue	IoRT systems are heterogeneous in three ways. To begin, differences in operating systems, software, hardware architecture, features, platforms, and communication mediums cause robot heterogeneity. Secondly, heterogeneity is also caused by a range of services being offered by different cloud service providers, each with its own typical policies. Third, there is a wide range of wireless networks, including cell phones, WiMAX, wireless local area networks, and satellite communication	These issues necessitate the development of middleware such as Open RTM and ROS while developing IoRT systems
Energy efficiency	In order to run the IoRT system and relay the massive volumes of data created by the IoRT devices for computation, analysis, or storage requires a great amount of energy. As a result, the energy consumption of IoRT systems continues to be a significant issue. Assuring network QoS while consuming the least amount of energy possible for IoRT devices is an open problem for any future IoRT application	Fog computing provides yet another benefit to IoRT technologies in terms of energy efficiency. Building a system that is energy-efficient may be accomplished in several methods. The spread of data and the execution of tasks at fog nodes allow for a significant reduction in the amount of traffic that is generated on the servers in compared to a traditional cloud environment. Fog nodes may also save updates while the sensor is sleeping and then send them to the device when it wakes up
Security issue	In IoRT applications, reliability and security are crucial. There are two primary causes of this. As a first step in protecting sensitive information that travels over the Internet, as well as a second step in protecting the devices and systems involved. Additionally, since data moves across the web, the prospect of numerous security threats like denial of service (DoS), account hijacking, and data loss arises	Edge computing has been offered as a possible method to solve this issue. Because of this, the majority of data analysis occurs at the edge. As a result, the transmission of sensitive data over the Internet has been restricted to a limited extent

Table 2 (continued)

Problem	Definition	Solution
Safety issue	Due to the mission-critical nature of several CR systems, including air and ground mobility, catastrophe tracking and alerting systems, and healthcare systems, robot safety is crucial. Radiation leaks, processing unit malfunctions, thermal impacts, technical failures, electrical dangers, and physical dangers are all examples of safety failures. Anomalies can occur due to different fault causes in the physical, connectivity, perception, processing, and actuating fields	First and foremost, high-performance emulators are required, as are robotic testing areas. As a second step, robots should be tested in real-world circumstances (such as disaster locations)
Data storage location	Cloud data centers are located a long distance away from the end devices. However, the location of the data center should be as close as possible to the position of the user in order to minimize latency, as well as reduce the amount of energy needed	When fog computing is employed instead of cloud computing, users may access data quickly and easily from a nearby area rather than a vast distance away

for these difficult challenges because of its benefits of timeliness and energy efficiency [42]. Hence, in this section, some related works will be reviewed to address the energy consumption by the computation offloading technique. Offloading strategies have been suggested in the literature in a variety of forms. There are two types of offloading strategies: (a) performance-based offloading strategies [43–49] and (b) energy-performance-based offloading strategies [42, 50–59].

3.1 Performance-based offloading

The performance-based offloading technique aims to improve end-user processing time. Hence, in the following section, studies that enhance the computation offloading execution time will be discussed. Sheng et al. [43] investigated the relationship between differential uploading delay and co-channel interference. In addition, they suggested a computational offloading method that aimed to lower the customers' average delay by concurrently optimizing the decision on the offloading and allocating resources. To address the challenge of reducing the total completion period, Li et al. [44] suggested a combined optimization technique established on the Genetic Algorithm (GA) for offloading a proportion of the task, link bandwidth, and mobile edge server (MES) computational elements. This study is simple because it only looked at the time it takes to complete the task and ignored energy consumption. Joint Partial Offloading and Resource Allocation (JPORA) is a significant approach presented by Saleem et al. [45]. JPORA iteratively changes data segmentation to reduce task execution delay by allocating QoS-aware communication resources to cellular lines and interference-aware resources to D2D links. A target expression with a constraint item was obtained by Li et al. [46] by transforming the task offloading problem of fog computation into the matching problem between task and node. However, the improved differential evolution (IDE) technique was presented to address the task offloading issue.

Miao et al. [47] suggest a method for decreasing the processing duration of applications through an algorithm that involves the migration of tasks and the offloading of computation. The algorithm involves several components, including a task prediction mechanism based on the LSTM algorithm, a computation offloading strategy for mobile devices that utilizes task prediction, and a task migration scheme for edge cloud scheduling. These components work together to improve the efficiency of the edge computing offloading model. For mobile device users who have enough energy and solely care about the computation time overhead, Zhou and Jadoon [48] suggested a partial computation offloading technique based on game theory for multi-user edge computing. A digital twin edge network (DITEN) was suggested by Sun et al. [49]. Through deep reinforcement learning, an approach is presented to reduce the offloading latency.

3.2 Energy-performance-based offloading

The objective of an energy-performance-based offloading technique is to decrease the energy consumption and task processing duration of end-user devices. As a

result, computation-intensive tasks need to be carried out on remote servers. Cheng et al. [50] presented a task offloading decision technique for Internet of Vehicle-based edge computing with Particle Swarm Optimization (PSO). The PSO was performed in order to transform task offloading into the process and achieve the best offloading strategy. The suggested approach for offloading can efficiently decrease the terminal devices' energy consumption while ensuring user service quality. Han et al. [51] utilized nonorthogonal multiple access (NOMA) so that a user is able to delegate tasks to numerous adjacent devices with idle computer resources. Through the use of a decomposition strategy, they proposed a two-tier optimization scheme called Multitask Joint Computation Offloading and Resource Allocation (MT-JCORA). Transmission Scheduling and Computation Offloading (TSCO) was developed by Hazra et al. [52] to meet energy and delay requirements in a fog environment. In addition to that, they presented an approach for task offloading that is based on graphs and utilizes constrained-restricted mixed linear programming. To address the issue of energy-efficient and delay-aware task allocation problems, Singh et al. [53] introduced the Energy-efficient task offloading strategy (EETOS) based on the Levy-flight moth flame optimization (LMFO) method. Huang et al. [54] suggest a method called Multi-Objective Whale Optimization Algorithm (MOWOA) that takes into account time and energy consumption to determine the best computation offloading mechanism. Yan et al. [55] have addressed the challenge of jointly optimizing task offloading and resource allocation, taking into account both energy consumption and execution time. They have also demonstrated that optimal offloading decisions follow a "one climb" policy, which enabled the development of a Gibbs sampling algorithm with reduced complexity to determine the optimal offloading decisions. Gu et al. [56] proposed a method for offloading computations and allocating transmit power that uses less energy and takes less time to complete. By using the offloading of computation dynamically and transmission power allocation for MEC, they created an energy efficiency cost reduction issue that fulfills Mobile devices' completion time deadline restriction.

With a supervised deep learning approach, Abbas et al. [57] examined the partial offloading strategy in mobile edge computing (MEC). They effectively chose the partial offloading strategy and the amount of each unit of the task to decrease service latency and energy usage of the user equipment using the suggested approach, the complete and energy-efficient deep learning-based offloading method (CEDOT). By using a collaborative computing paradigm that addressed the vertical and horizontal cooperation across fog and cloud nodes while reducing total end device energy according to service latency limitations. In hierarchical fog-cloud systems, Nguyen et al. [58] developed innovative and effective methods for combining the compression of data and computation discharging to reduce energy and delay costs. In particular, they looked at scenarios in which data compression only was utilized by end devices and both end devices and the fog node. In Multiple user fog computing platform, Chang et al. [59] developed an energy-efficient approach for computation offloading. They believed that users must decide whether or not to discharge processes to the nearest fog node, depending on energy usage and latency constraints. They devised an energy-efficient improvement solution to minimize energy consumption, subject to delay limitations. Chen et al. [42] devised a fog-assisted IoT network

data aggregation computation offloading strategy that was both privacy and energy conscious. The aim of the suggested strategy is to reduce the overall energy usage of computing tasks that provide a secure three-layer computing architecture and an energy-efficient offloading decision technique based on the momentum gradient descent. Table 3 summarizes several techniques used in related work.

Furthermore, many of the previous related works in the literature review focus solely on either performance and ignore energy consumption or energy performance but suffer from some limitations such as scalability, heterogeneity, and complexity. In contrast, our proposed approach, which combines the genetic algorithm with Particle Swarm Optimization, achieves a balance between energy efficiency and performance. By applying partial offloading of the computational tasks between IoRT devices and fog nodes, our approach reduces energy consumption while maintaining a high level of performance. Moreover, our approach is designed to address the limitations of previous studies, such as homogeneity, complexity, and scalability. Therefore, our approach contributes to the development of more practical and effective solutions for energy-efficient and high-performance computation offloading in fog-based IoRT networks.

4 The proposed approach

This section focuses on explaining the proposed approach to reduce the energy consumption of IoRT in more detail. The approach is based on developing an IoRT hierarchical architecture for deploying IoRT applications into the three-layer fog landscape. Then, the proposed framework is introduced to minimize energy consumption using a partial task offloading strategy.

4.1 The proposed IoRT architecture

Figure 3 shows the hierarchical architecture of IoRT based on fog computing technology to help IoRT devices perform better and use less energy. The hierarchical architecture consists of the following three layers:

1. Hardware Layer: It consists of various IoRT devices. Here, we solely employ IoRT devices for intelligent sensing. Typically, these devices are geographically dispersed. It is accountable for detecting and transmitting the perceived data of physical objects to the layer above for analysis and storage. These devices are connected to upper layer nodes by wireless technologies like SigFox, Wi-Fi, ZigBee, Bluetooth Low Energy (BLE), 4G, and 5G [2]. This layer is responsible for taking the offloading decision by selecting the most acceptable task offloading ratio α.
2. Fog Layer: This layer lies close to end devices at the edge of the network. It consists of a variety of nodes that typically involve routers, gateways, switches, access points, base stations, fog servers, etc. These nodes have the power to process, transfer, and store sensed data. In the fog layer, real-time analysis and latency-sensitive applications can be done. In addition, the fog nodes are linked

Table 3 The summarization of the computation offloading researches

Research	Energy-aware	Delay aware	Technique	Drawbacks	Network type	Published year
Sheng et al. [43]	No	Yes	Optimization-Theory Based Computation Offloading and Resource Allocation (OCORA) and Matching-Theory Based Computation Offloading and Resource Allocation (MCORA) schemes	The presumption that user equipment (UE) and computation tasks are identical may not be practical The paper makes no mention of the prospective energy consumption implications of the suggested approach	MEC	2020
Li et al. [44]	No	Yes	Genetic Algorithm (GA)	The limited assessment of the proposed method Small-scale simulation environment	MEC	2020
Saleem et al. [45]	No	Yes	Joint Partial Offloading and Resource Allocation (JPORA)	The research doesn't address how the proposed technique would work with heterogeneous devices	MEC	2020
Li et al. [46]	No	Yes	Improved Differential Evolution (IDE) technique	The paper does not address the security and privacy implications of the proposed methodology	Fog computing	2020
Miao et al. [47]	No	Yes	A deep learning algorithm based on the Long Short-Term Memory (LSTM) architecture	It may be necessary to conduct additional research and evaluation to determine the approach's practicality and scalability	MEC	2020
Zhou and Jadoon [48]	No	Yes	Game theory-based strategy	The technique may be influenced by various factors, including the number of users, task complexity, and resource availability	MEC	2020
sun et al. [49]	No	Yes	Digital twin edge networks (DITEN) using deep reinforcement learning	The suggested method has only been tested on digital twin edge networks and may not be relevant to other edge computing environments	MEC	2020
Cheng et al. [50]	Yes	Yes	Particle Swarm Optimization (PSO)	The effect of shifting network conditions or the number of connected devices on the mechanism's performance may not be taken into account	MEC	2021
Han et al. [51]	Yes	Yes	Multitask Joint Computation Offloading and Resource Allocation (MT-JCORA)	The proposed method's applicability to other contexts may be ambiguous	MEC	2022

Table 3 (continued)

Research	Energy-aware	Delay aware	Technique	Drawbacks	Network type	Published year
Hazra et al. [52]	Yes	Yes	Transmission Scheduling and Computation Offloading (TSCO)	Implementation costs were not discussed in the article	Fog computing	2022
Singh et al. [53]	Yes	Yes	Energy-efficient task offloading strategy (EETOS) based on Levy-flight moth flame optimization (LMFO)	For practical deployment, the authors have not discussed the algorithm's hardware and software requirements	Fog computing	2022
Huang et al. [54]	Yes	Yes	Multi-Objective Whale Optimization Algorithm (MOWOA)	The complexity of the proposed algorithm may make it challenging to apply to large-scale problems	MEC	2021
Yan et al. [55]	Yes	Yes	Gibbs sampling algorithm	The suggested approach has a high computational cost, which might make it impractical to use on devices with limited resources	MEC	2020
Gu et al. [56]	Yes	Yes	Lagrange multiplier method and the Karush–Kuhn–Tucker (KKT) conditions	The algorithm may not scale well to larger networks due to its computational complexity	MEC	2019
Abbas et al. [57]	Yes	Yes	The comprehensive and energy-efficient deep learning-based offloading technique (CEDOT)	The lack of information regarding the dataset employed	MEC	2021
Nguyen et al. [58]	Yes	Yes	Joint Data compression (DC), Computation offloading, and Resource Allocation (JCORA) algorithm	Crucial elements like system throughput, latency, and reliability were not considered	MEC	2020
Chang et al. [59]	Yes	Yes	Alternating direction method of multipliers (ADMM)-based distributed algorithm	The suggested optimization approach presumes that the fog node has complete information of the network circumstances, which may not be realistic	Fog computing	2017
Chen et al. [42]	Yes	Yes	Momentum Gradient Descent Based Energy Efficient Offloading Decision Algorithm	The suggested approach may not adequately address privacy issues since data may be disclosed to unauthorized persons during the offloading of computations	Fog computing	2020

to the centralized cloud servers to achieve more efficient computing and storage capabilities [42].

3. Cloud Layer: It is composed of many highly efficient servers and storage units and offers diverse applications. It has strong computational and storage resources for the comprehensive processing and storing of a large volume of data [60].

4.2 The proposed IoRT framework

To reduce the energy consumption of IoRT devices, the energy consumed in the computation process will decrease by offloading a ratio of a task to fog nodes and the rest of the task will be processed locally on the IoRT devices. Also, IoRT devices generate data that differ in size, e.g. data transmitted from sensors (e.g. Ultrasonic sensors) usually are less than images transmitted from cameras. The variety of the data packages impacts the behavior of the fog node during the processing event. Heavy data packages thus take more time than light data packets to process. On the other hand, the computation capacity of fog nodes is weaker than that of cloud servers [41]. Therefore, to achieve a low latency response and reduce the power consumed in the idle state sector at the same time, we should apply the collaboration concept in the fog layer to process heavy tasks faster. Therefore, if the fog node accepts a request based on of its current load, it either performs the whole request or begins handling the application portion and transfers the remainder of the request to a separate node (in case of heavy package size). Figure 4 demonstrates the proposed IoRT based on the fog framework that will be suitable for time-sensitive and efficient-energy consumption applications [61].

5 System model and problem formulation

A new energy modeling method for IoRT devices is suggested in this section to increase energy efficiency. The robot will quantify and forecast energy consumption by using the energy model as a reference to energy-saving strategies. The IoRT device's power usage takes into account six key factors: the transmission cost, the computation system cost, the idle state cost, the sensor system cost, the control system cost, and the movement system cost, as shown in Fig. 5. Figure 5 represents the top six research challenges related to the design of energy-efficient IoRT systems [62, 63].

Firstly, since IoRT is based on fog computing, tasks may be executed on the IoRT device or moved to the fog nodes and implemented there. Afterward, the results may be sent to the device, allowing further operations to be carried out. Deciding whether the computation process on mobile devices or fog is best applied is a significant matter because it has a huge impact on the use of energy. Also, the gathered data from sensors will be transferred to the fog and this will consume energy in the communication process. The amount of energy used to send data over a network will depend on the amount of data and the data transmission rate. Secondly,

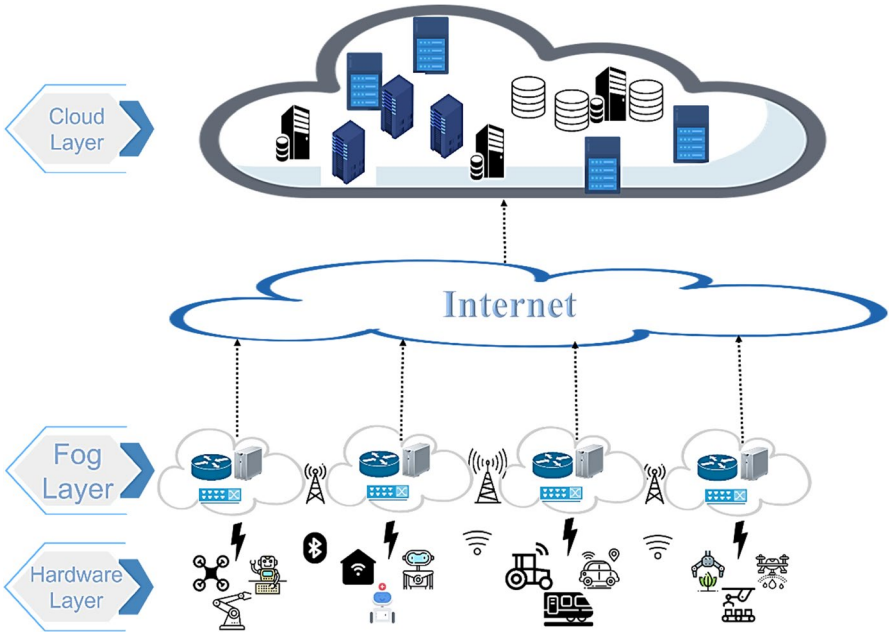


Fig. 3 The hierarchical architecture of IoRT based on fog computing

IoRT devices need to describe and recognize the surrounding environment, so it is connected to many sensors such as GPS, ultrasonic, cameras, encoders, gyroscopes, compasses, etc. Also, these sensors affect the energy consumed, so it must be considered when calculating the energy of the robot. Finally, the control circuit board and motion system will affect the robot's energy. Equation (1) represents the consumed energy based on the previous scheme [63].

$$E_{total} = E_{trans} + E_{comp} + E_{idle} + E_{sensor} + E_{control} + E_{mov} \tag{1}$$

In this article, we apply partial computation offloading, so we mainly take into account computation and transmission energy and the rest of (1) will not be affected and assumed as Δ as shown in (2).

$$E_{total} = E_{trans} + E_{comp} + \Delta \tag{2}$$

In the remainder of this part, the content of the computation and transmission energy sectors will be examined in detail. Let's assume having a system consisting of K end users (*EU*) scattered in the system at random, and each end user is expressed as EU_k, a fog node, and a distant cloud server, as demonstrated in Fig. 6. Each EU has a computation task $T_k = \{M_k, C_{comp}, P_{comp}\}$, M_k explains the size of the task to be done, C_{comp} illustrates the computation capacity of the EU_k which is the number of processed bits in a second, and P_{comp} represents the computation power for implementing one cycle at the EU_k in Watt, where k ∈ (1,2,3,……8).

By applying partial offloading each EU_k must offload α_k the ratio of tasks to the fog node for computation, and this offloading ratio must be greater than 0 and less than or equal to 1, leaving the rest of the fractional processing tasks $(1 - \alpha_k)$ to be processed locally, hence the local computation energy E_{comp} and local computation time $T_{k,local}$ can be calculated as:

$$E_{comp} = P_{comp} * \frac{(1 - \propto_k) * M_k}{C_{comp}} \tag{3}$$

$$T_{k,local} = \frac{(1 - \propto_k) * M_k * L_k}{C_{comp}} \tag{4}$$

where L_k represents the CPU cycles needed to process 1bit of the EU_k.

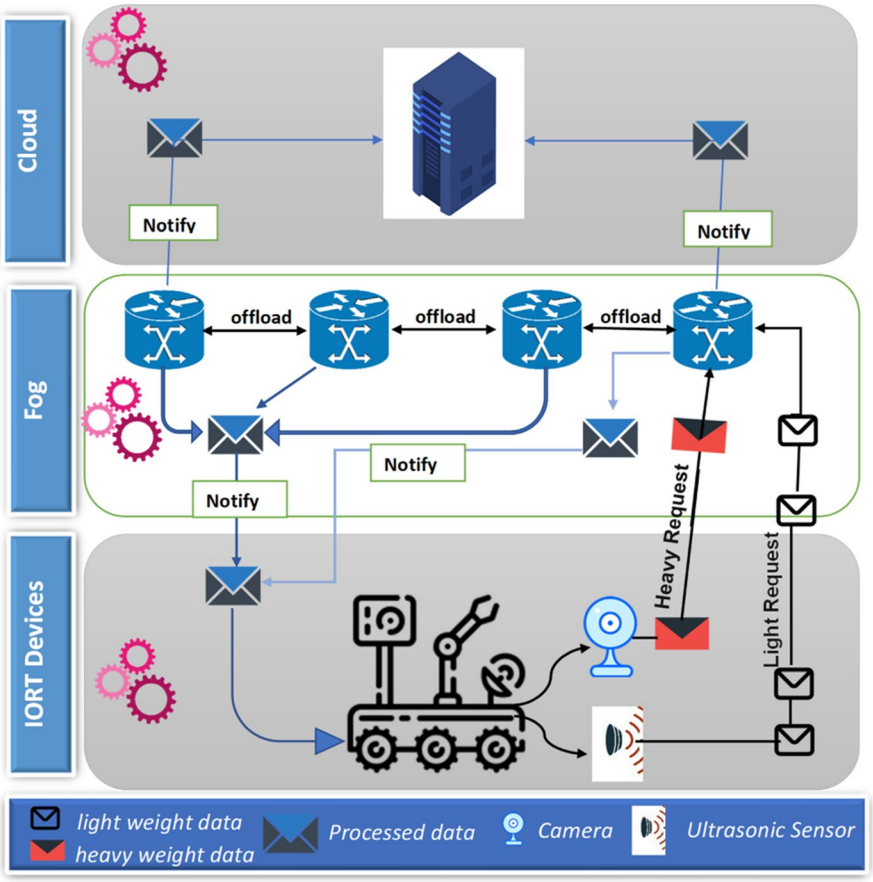


Fig. 4 Internet of robotic things based on fog framework

Frequency division multiple access is used for uploading the offloading data. The system's total bandwidth is B_{total} and the bandwidth assigned to EU_k is b_k. So, in the transmission term, energy usage involves transmitted data size, network transmission rate R_k, and the amount of energy needed to transmit network data e_{send} as shown in (5).

$$E_{trans} = \frac{\propto_k * M_k}{R_k} * e_{send} \tag{5}$$

Energy consumption for transferring data in the network e_{send} is calculated according to the Shannon equation, and the transmission rate of the EU_k may be expressed as:

$$R_k = b_k \log_2 \left(1 + \frac{e_{send} h_k^2}{N_0 b_k}\right) \tag{6}$$

where h_k represents the wireless channel gain and N_0 represents the spectral density of the channel noise power. Then the energy consumption for data transmission in the network can be indicated as:

$$e_{send} = \frac{N_0 b_k}{h_k^2}\left(2^{\frac{R_k}{b_k}} - 1\right) \tag{7}$$

The final form of the transmission energy consumption and the uplink time is expressed below:

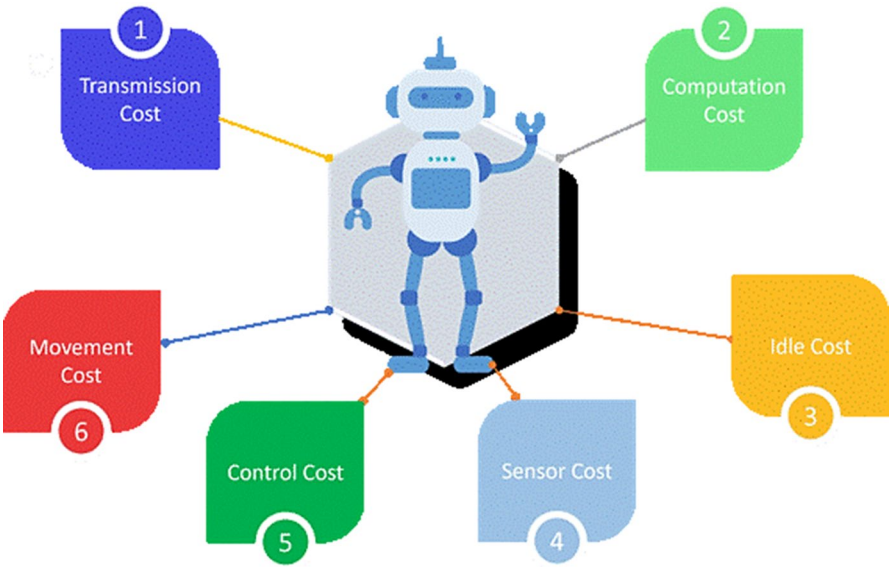


Fig. 5 The top six design considerations for energy-efficient IoRT devices

$$E_{trans} = \frac{N_0 b_k}{h_k^2}\left(2^{\frac{R_k}{b_k}} - 1\right) * \frac{\propto_k * M_k}{R_k} \tag{8}$$

$$T_{trans} = \frac{\propto_k * M_k}{R_k} \tag{9}$$

Finally, the total computation time of the offloaded tasks consists of the offloading transmission time and the fog processing time. Therefore, the final form of the total energy consumed during the offloading process for all user devices in the system and the total computation time of the offloaded task can be expressed, respectively, as:

$$E_{total} = \sum_{k=1}^{j} \frac{N_0 b_k}{h_k^2}\left(2^{\frac{R_k}{b_k}} - 1\right) * \frac{\propto_k * M_k}{R_k} + P_{comp} * \frac{(1 - \propto_k) M_k}{C_{comp}} + \Delta \tag{10}$$

$$T_{k,offload} = \frac{\propto_k * M_k}{R_k} + \frac{\propto_k * M_k * L_k}{F_{comp}} \tag{11}$$

The primary goal of the proposal presented in this article is to reduce the total amount of energy consumed while meeting the delay requirement and improve the performance of all IoRT devices by implementing a flexible network model based on fog computing technology. The IoRT system efficiency optimization problem has several competing objectives that must be optimized against a set of possible and achievable solutions that are not predetermined but proposed by a set of boundaries and limitations. In order to reach the proposed goal, we leverage the QoS concepts of computation time and energy consumption. They are characterized as follows:

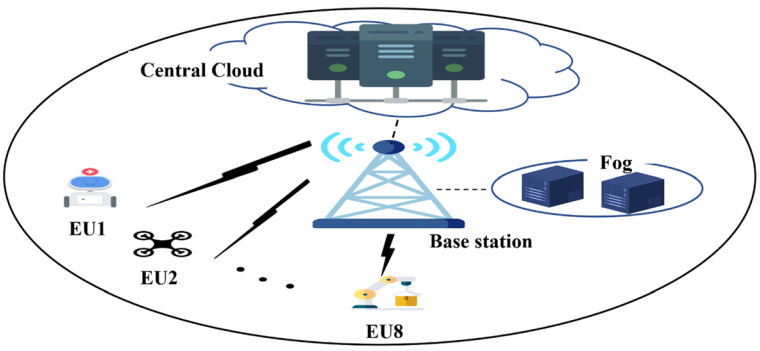


Fig. 6 System model

$$Optimal\ solution = \min_{\{\propto_k and R_k\}} E_{total} = \left\{ \begin{array}{l} \min\limits_{0 \le \alpha_k \le 1 and} \sum\limits_{k=1}^{j} \frac{N_0 b_k}{h_k^2} \left(2^{\frac{R_k}{b_k}} - 1 \right) * \frac{\propto_k * M_k}{R_k} + \\ \sum\limits_{k=1}^{j} R_k \le B_{total} \\ \min\limits_{0 \le \alpha_k \le 1 and} \sum\limits_{k=1}^{j} \left(P_{comp} * \frac{(1-\propto_k) M_k}{C_{comp}} \right) \\ \sum\limits_{k=1}^{j} R_k \le B_{total} \end{array} \right\} \tag{12}$$

where $T_{k,max} = \max\left[T_{k,local}, T_{k,offloaded}\right] \le \mathrm{T}$.

In (12), the ideal scenario for optimal or near-optimal consumed energy in the first half is to reduce the consumed energy in the transmutation in order to get the best possible result (i.e., reducing the amount of energy needed to transport the data packet to its destination and back to its sender). The consumed energy in the first half can be optimized by getting the optimal value of the transmission rate R_k and offloading ratio $\propto_k$ of the data. The second half of (12) represents the optimal scenario in terms of reducing energy consumption during local computing of the portion of the task (1- $\propto_k$) depending on the offloading ratio.

In the cost function (12), the goal is to modify the parameters α_k and R_k to minimize the overall energy consumption E_{total}. The following are a few constraints to keep in mind:

a. The device k offloading ratio should be between 0 and 1.
b. Ensures that the total data transmission rate of each device in the sub-region does not surpass the channel bandwidth value.
c. Guarantees that the processing time after offloading procedure must be less than or equal to the duration when all the tasks are computed locally and meet the task deadline.

6 The proposed methodology

Next, the flow of an Improved Hybrid Optimization technique based on GA Combined with PSO named IHOGCP will be discussed. Figure 7 illustrates the main steps of the IHOGCP algorithm. The IHOGCP algorithm begins with the random population generation of individuals, defines a certain number of iterations, and determines the algorithm parameters. The initialized population is fed into the GA algorithm for the specified number of iterations. Each solution is represented by a

chromosome, which is a fixed-length string. A chromosome consists of numerous genes, and the overall chromosomal count indicates the population size. A fitness function is used to evaluate the quality of one solution over another. Several operators, including selection, mutation, and crossover, are employed during the development process to improve solutions passed to the next generation. In consecutive iterations, solutions are developed until a suitable criterion, i.e., a maximum number of generations or a preset fitness value is reached.

Following that, PSO's particles are initialized once the GA-optimized solutions are supplied, in which the position of the particles reflects the GA population's solutions and the velocity of each particle is initialized at the beginning. Position update and velocity update are the main two operators in the PSO scheme. To find the optimum particle position, the two "best" values, pBest and gBest, are modified. Based on these two "best" values, the particle's position may be changed by altering its velocity dynamically toward the optimal value of calculation, known as the global optimum. Each computation updates the velocity and position of each particle. The optimal position of the particle, known as pBest, is assessed for every particle of the swarm. If the present position is better than the prior one, the particle position shall be altered. The old position shall otherwise be maintained. Afterward, determine the gBest, which is the particle's best position in the whole population. These four stages are continued until a stopping condition is satisfied, indicating that the population's particles are in the best position. Finally, solutions are continuously optimized until the maximum number of generations is achieved.

7 Proposed algorithm formulation

The optimization problem (12) cannot be solved by standard optimization techniques such as the Lagrange multiplier method or the KKT condition, since it is a non-convex problem. This problem is well suited to the usage of meta-heuristic algorithms, as meta-heuristic algorithms are a type of computational intelligence paradigm that is used to address sophisticated optimization problems. Due to their robustness and ability to manage non-linearity and discontinuity, meta-heuristic optimization algorithms are extensively used. The meta-heuristic algorithms are including Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and others. In any case, each algorithm has its own advantages and disadvantages. For example, PSO's convergence is rapid, but it can be locked into locally optimum solutions when it is used to solve complicated problems with high-dimensional solution spaces. In addition, genetic algorithms have a great ability for global search and various solutions, but their convergence process is time demanding. This study incorporates the benefits of these two algorithms and proposed an Improved Hybrid Optimization technique based on GA Combined with PSO named IHOGCP, as shown in Algorithm 1. Following that, we'll go through the algorithm's sequence, followed by the GA and PSO used in this study.

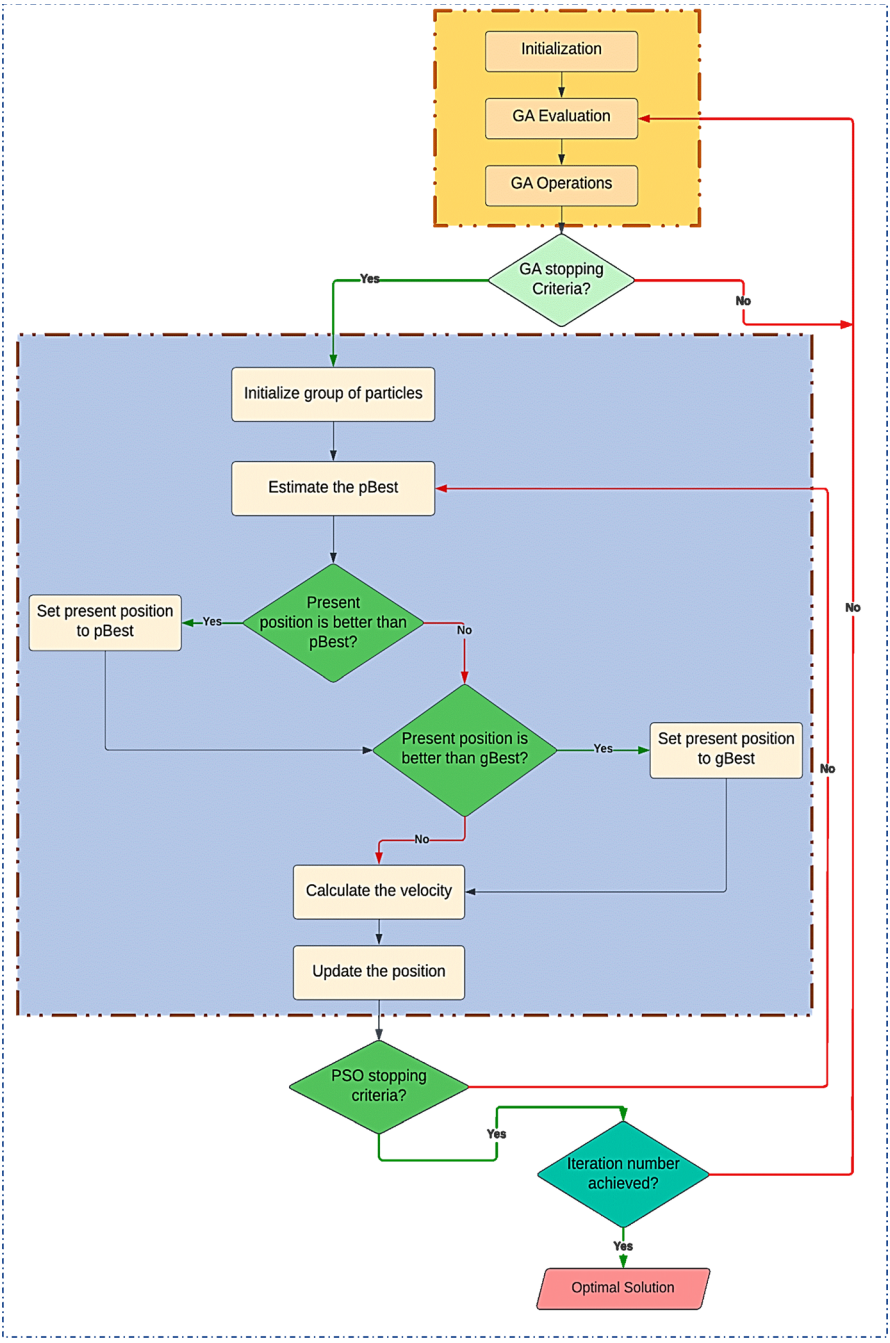


Fig. 7 The flowchart of the IHOGCP algorithm

Algorithm 1: Improved Hybrid Optimization technique based on GA combined with PSO (IHOGCP)

Input: the size of the population S, the number of variables N, the Max iteration of IHOGCP I, the parameters of GA P_m, P_c, I_1, tournament size t_S, the parameters of PSO w, C_1, C_2, I_2.

Output: the optimal values $\propto_K^*, R_K^*$ and E_{total}^*.

begin

Randomly generate individuals in initial population P and assign number of generations to 0 (i = 0).

While termination criteria is not satisfied

do

- Perform GA operations in Algorithm 2
- Perform PSO operations in Algorithms 3
- i = i + 1

End

Obtaining the optimal value of $\propto_K^*, R_K^*$

The optimal energy consumption can be obtained by equation (12) with optimal values in the previous step

End

Algorithm 2: Genetic Algorithm operations

Input: the population P, P_c, P_m, t_S , Max iteration I_1.

Output: The S solutions after I_1 iterations.

Begin

calculate the fitness of individuals and assign number of generations to 0 ($i_1 = 0$).

While termination criteria is not satisfied

do

- Select parents by the tournament selection
- Apply crossover by probability P_c
- Apply mutation by probability P_m
- Evaluate the new candidates
- $i_1 = i_1 + 1$

End

PSO is used to increase the performance of GA. As demonstrated in Algorithm 1, we utilize GA as the algorithm's foundation, while PSO is used to enhance the solution provided by GA. Algorithm 1 starts with a population initialization randomly. Then, the individuals are handled by GA operations in Algorithm 2 for I_1 iterations. Following that, PSO's particles are initialized once the GA-optimized solutions are provided from Algorithm 2 and perform the PSO operations in Algorithm3 for I_2 iterations. The individuals are repeatedly optimized using GA and PSO until convergence or the maximum number of generations is reached.

Algorithm 3: PSO Algorithm operations

Input: the best solutions (particles) produced by algorithm 2, inertia wight (w), C_1, C_2, Max iteration I_2.

Output: The position of S particles after I_2 iterations.

Begin

Initialize the velocity of S particles and assigning number of generations to 0 ($i_2 = 0$).

While termination criteria are not satisfied

do

  calculate the fitness function of particles

  Update the p_{best} and g_{best} for each particle

  Update the velocity of the particles

  Update the position of the particles

  $i_2 = i_2 + 1$

End

7.1 GA schema

In order to tackle complicated problems, GA uses a random search strategy that mimics biological evolution, which uses the notion of survival of the fittest as its evolution concept. Selection, crossover, and mutation are the three fundamental operators in a typical GA. Each individual is represented by a chromosome, which is a string (typically binary or decimal) that encodes the solution. The selection operator replicates chromosomes of higher quality to pass on to the next generation in order to improve fitness values in the population as a whole. Evolutionary populations benefit from mutations and crossovers, which give them different, but presumably higher-quality genetic resources.

7.1.1 Chromosome structure and selection operation

The initial stage in solving the problem is chromosomal encoding, which is highly dependent on the problem. Figure 8 shows the procedure of chromosome encoding. The population matrix is made up of a number of individuals, each individual has a set of genes. {∝, R} becomes a gene for the individual in the population. In each iteration, the individual with the lowest energy consumption is selected as the optimum individual. There are two widely used ways for the selection operation: roulette wheel and tournament. This study adopts the tournament since the selection technique for the roulette wheel is more suited for maximization problems. In tournament selection, choosing K individuals from the population randomly and selecting the fittest to become a parent. For picking the new parent, an identical process is

performed. The crossover and mutation procedures are used by parents to generate offspring to enhance variety and provide better solutions to this problem.

7.1.2 Crossover and mutation operation

Like reproduction and biological crossover, the crossover operator works in the same way. This includes the selection of more than one parent and the production of one or more offspring utilizing the parents' genetic material. The goal is to produce a bigger offspring pool with higher fitness levels. The concept underlying reproduction is that the offspring is even more fitting by combining some features of two already suitable individuals, as it may be best inherited from each of their parents. Therefore, crossover aids in optimization and enhancing convergence. Crossover is carried out when the parents are interchanged by picking a random point on the chromosome. After then, the crossover creates a new offspring depending on the exchange point selected with certain portions of the parents. Figure 9 shows an example of a crossover operation. The GA has a probability of crossover, which decides whether crossover will occur or not. From practical and theoretical research, results point to a substantially greater likelihood probability P_c of crossover in the 0.6–0.95 range.

In short, mutation may be described as a little random chromosomal modification in order to find a new solution. It is utilized in order to preserve and add variety to the population and is generally employed with low probabilities P_m around 0.0001–0.05. This operator generates new adaptive solutions by applying modifications randomly to one or more "genes" in order to produce new offspring, thereby creating new adaptive individuals that prevent local optimum. For the integer representation, a random value is assigned to a randomly selected gene from the list of possible values. Figure 10 shows the exact procedure of the operation. A new random value is assigned to the mutation if the modified gene is above the limit of the constraint.

7.2 PSO schema

One of the approaches for evolutionary computation is Particle Swarm Optimization (PSO). The PSO is a population-based optimization approach inspired by bird flock and school fish movements. This approach searches for the best solution using agents, called particles, which have a stochastic and a determinist component to adapt their paths. The population of moving particles is called a swarm. Particles

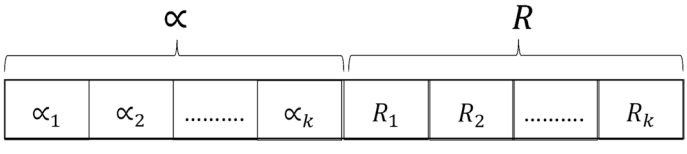


Fig. 8 Chromosome encoding method

only have two properties: position and velocity. Each particle adjusts its position in the search space and enhances its speed in response to its own and other particles' movement experiences in order to achieve a better suitable position. The particle's position in PSO represents the solution in the GA population, and the velocity shows the rate of change in the position of the particle. Each particle has a record that recalls its prior best position and is termed the particle's best position as (pBest). Each particle has a pBest, and the particle with the highest fitness value is referred to as the group's best position (gBest). Hence, the "best" position of each particle, as well as the "best" position of the group, have an influence on the particle. Suppose that the search space is S-dimensional, and the ith individual in the swarm may be expressed by an S-dimensional vector $X_i = \left(X_{i,1}, X_{i,2}, \ldots, X_{i,S}\right)$. Another S-dimensional vector can be used to describe the particle's velocity $V_i = \left(V_{i,1}, V_{i,2}, \ldots, V_{i,S}\right)$. Each particle adjusts its position according to the new velocity with each iteration. During the iteration procedure, the velocity and position of a particle are updated as (13) and (14).

$$V_{i,d}^{t+1} = \omega V_{(i,d)}^{t} + r_1.c_1(pBest_{(i,d)}^{t} - X_{(i,d)}^{t}) + r_2.c_2(gBest_{(i,d)}^{t} - X_{(i,d)}^{t}) \tag{13}$$

$$X_{i,d}^{t+1} = X_{i,d}^{t} + V_{i,d}^{t+1} \tag{14}$$

Here, t = iteration number, $d = 1,2,3, \ldots, S$; $i = 1, 2, \ldots, M$, where M is a swarm size. ω, c_1, c_2, r_1, and r_2 are, respectively, inertia weight which means the particle's weight at its prior speed, accelerator constants, and two random parameters within [0, 1] to increase the randomness in the search. A particle is guided to its optimal position by a set of acceleration parameters, which are represented by c_1 and c_2, which are two constants. It's conceivable that the optimal answer will be neglected if the values of c_1 and c_2 are large, resulting in a fast search. It is possible that the search time will be slow if the values of c_1 and c_2 are low, and a local optimum may be found. Thus, we assume that $c_1 = 2$ and $c_2 = 2$.

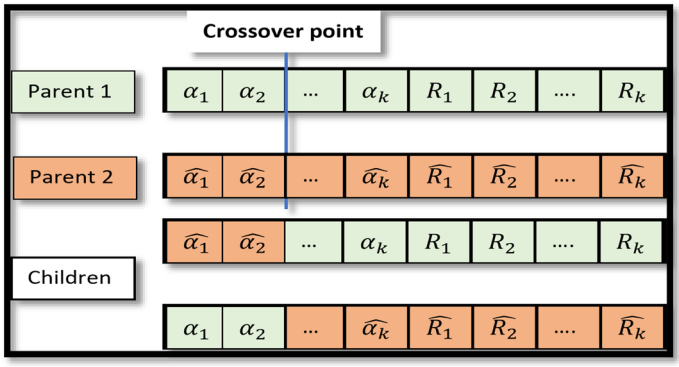


Fig. 9 Crossover operation

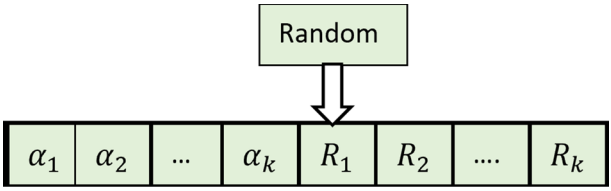


Fig. 10 Mutation operation

8 Performance evaluation

The numerical experiments in this section will be based on the system model presented in Fig. 6 and the algorithm provided above. All experiments are executed on a PC with a Windows 10 operating system and 12 GB of RAM, and the algorithms are programmed in MATLAB 2021a.

We will suppose that the sub-area is equipped with a fog node and eight IoRT devices. Specifically, each fog node has a computation capacity of 15 GHz, whereas the computation capacity of each IoRT device is designed to be in the range of 0.1–1GHZ. Furthermore, the amount of CPU cycles required to process one bit of data is assumed to equal 1000 cycles/bit. Assuming that the spectrum is partitioned into k channels, with the bandwidth of each channel being identical, we set the channel bandwidth $\boldsymbol{b_k} = 5$ MHZ and the white Gaussian noise to be $\boldsymbol{N_0} = 10^{-7}$ W in the communication environment. Each end user has a unique task that has to be calculated, and the data amount of the computing task is chosen at random from 1 to 20 Mb for each end user. In Table 4, we summarize the simulation parameters.

Figure 11 shows the simulation results for the total energy consumption with the number of iterations for eight end devices. As a first step, we ran our system through a variety of iterations to evaluate how the number of iterations influences the energy consumption of IoRT devices. In the beginning, the cost of the end devices decreased rapidly. In the following rounds, as seen in Fig. 11, the rate of decline was reduced as the number of repetitions increased. From Fig. 11, it can be observed that the suggested algorithm's overall energy consumption tends to converge after 90 iterations. For instance, increasing the number of iterations from one to ninety resulted in a drop in energy consumption by 18 percent from, 21,684 to 17,810 (mJ). Following that, the rate of decline reduced when the number of generations was above ninety, as seen in Fig. 11.

The energy consumption of the IoRT devices was estimated and compared for three different situations in order to evaluate the proposed approach performance versus different task sizes ranging from 1 to 15 MB. Figure 12 depicts the effect of increasing the task size on the overall amount of energy consumed throughout the task computation in the three different scenarios. The first situation was one in which all the tasks are offloaded remotely. Tasks in the second situation were all performed locally. The IHOGCP was implemented in the third situation. Figure 12 shows that the energy consumption rose as the size of the task increased. This indicates that the proposed algorithm has the lowest energy consumption of the

three situations that were tested. Local computing, on the other hand, has the largest energy consumption. In contrast to local computing, the full-task offloading technique uses more energy in task transmission, although it costs less energy than local computing. For instance, the suggested approach used 22,035 (mJ) of energy when the task was 12 MB in size. When comparing full-task offloading and local computing, the amount of energy consumed was reduced by 28.3 percent and 52.2 percent, respectively.

Then, as seen in Fig. 13, the performance of IHOGCP has been analyzed and compared to the Genetic Algorithm (GA) [64], Particle Swarm Optimization (PSO) [65], Whale Optimization Algorithm (WOA) [66], Artificial Bee Colony (ABC) [67], Ant Lion Optimizer (ALO) [68], Grey Wolf Optimizer (GWO) [69], and Salp Swarm Algorithm (SSA) [70]. According to the perspective of convergence, the PSO is the best option, followed by IHOGCP, then GA, and finally WOA. However, the other four algorithms take longer to converge, requiring more than 500 iterations to reach their convergence points. However, in terms of energy consumption, IHOGCP is the most efficient, whereas PSO is the most inefficient. GA's performance falls somewhere in the middle of that of IHOGCP and PSO. This is due to the fact that IHOGCP combines the benefits of GA and PSO, with GA being better at searching the global domain and PSO being faster at convergence. This means that compared to the single GA or PSO, IHOGCP performs better. For example, in the proposed approach when the number of iterations increased from 1 to 90 resulted in a drop in energy consumption by 18% in contrast to GA makes convergence after 322 iterations to reach the same result approximately. However, the PSO makes convergence after 20 generations but has the highest energy consumption. When comparing the proposed approach to the PSO algorithm, the consumed energy by the proposed approach has been reduced by 8 percent compared with PSO. Finally, due to the whale optimization algorithm suffering from slow convergence speed, low precision, and a propensity to slip into local optimums, it achieves convergence after 386 iterations and consumes 6 percent more energy than IHOGCP.

The best results obtained by various algorithms on eight IoRT devices in terms of task offloading ratio and transmission rate are shown in Table 5. One can see that each device has its own offloading ratio and transmission rate, which differs from

Table 4 Simulation parameters

Parameter	Value
Task size	[1, 20] Mb
Gaussian noise N_0	10^{-7} w
CPU cycles needed to process 1bit	1000 cycles/bit
Local computation capacity C_{comp}	[0.1, 1] GHz
Fog node computation capacity	15 GHz
User channel bandwidth b_k	5 MHz
A certain deadline constraint	4 s
Crossover probability	0.85
Mutation probability	0.0625

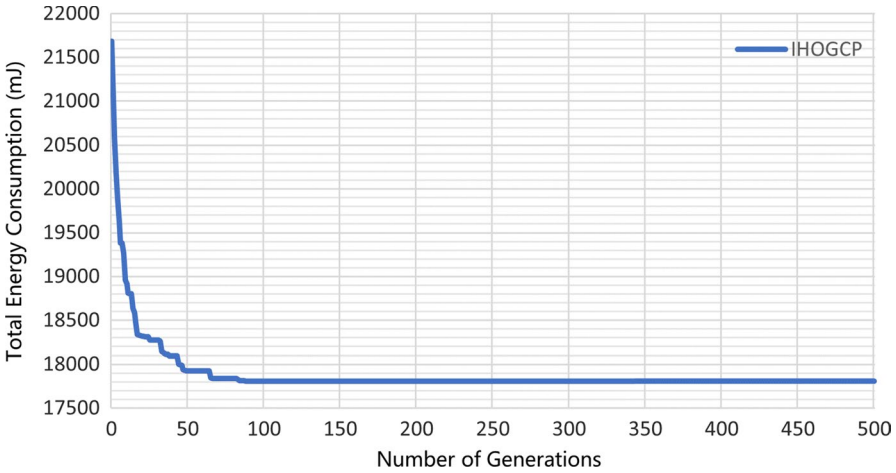


Fig. 11 Total energy consumption with iteration times

the other devices depending on the amount of the task being performed. It can be easily observed that the IHOGCP approach has the best results compared with other approaches under different task sizes in MATLAB experimental environment.

The influence of IoRT devices' local computation capacities on energy consumption and task offloading ratios in different schemas is illustrated in Fig. 14. According to the task sizes in Fig. 14a, we set them identical across all IoRT devices and equals 13 MB. It is obvious that as local computation capacity increased, total energy consumption decreased, and the offloading ratio decreased as well because

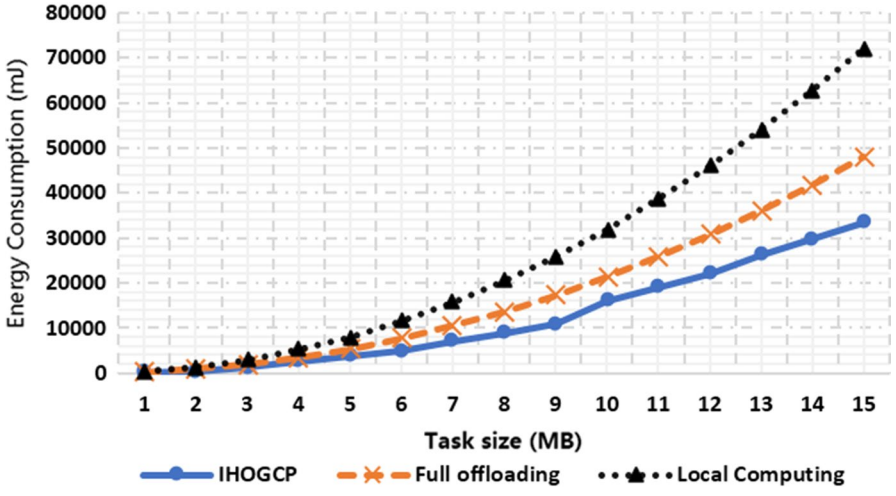


Fig. 12 The impact of task size on energy consumption

of that IoRT devices can process a large ratio of tasks locally to achieve the deadline constraints.

Clearly, the energy consumption in all schemas was near and substantial when the local computation capacity was as low as possible and equivalent to 0.1 GHz. As a result, the task-offloading ratio was as high as possible in order to complete the majority of tasks at the fog node as quickly as possible. However, when local computation capacity increased, the proposed approach was more efficient than alternative approaches and achieved a lower offloading ratio, owing to the fact that the IoRT's devices, which have high computation capacity, can complete tasks in a short period of time.

On the other hand, on each IoRT device, task sizes were randomly selected according to Fig. 14b. It's obvious that our proposed approach consumes less energy than others at different local computation capacities. However, we show that when a small amount of local computing capacity is used, the difference in energy consumption between algorithms is minor, and the difference arises when the amount of local computing capacity is increased. For instance, at 1 GHz computation capacity, the PSO algorithm consumes the most energy (19,255 mJ), followed by the WOA algorithm (18,916 mJ). In contrast, our suggested approach, GA algorithm, and ABC algorithm were found to use the least energy (17,804 mJ). Additionally, three other algorithms were tested, and they consumed energy as follows: Algorithm GWO used 18,454 mJ, Algorithm SSA used 18,410 mJ, and Algorithm ALO used 18,591 mJ. Overall, the additional algorithms performed better than PSO and WOA but worse than our proposed algorithm. Hence, it is noticed that the suggested

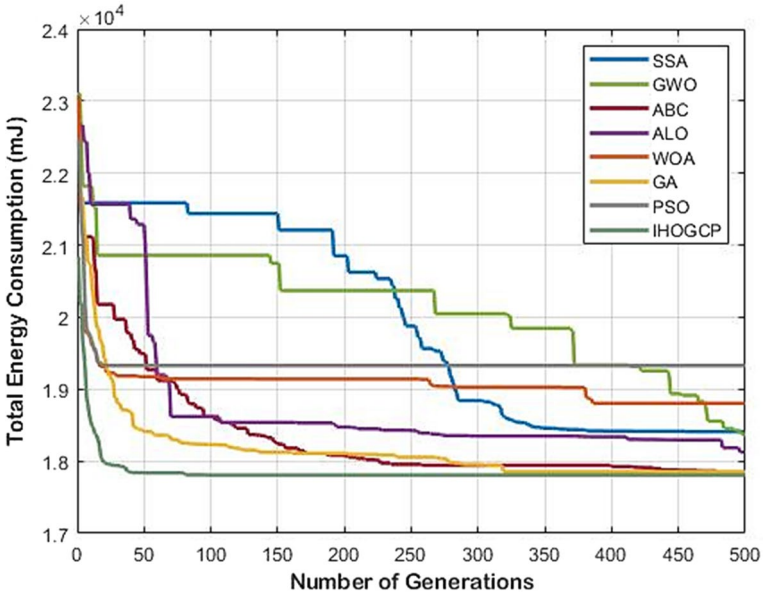
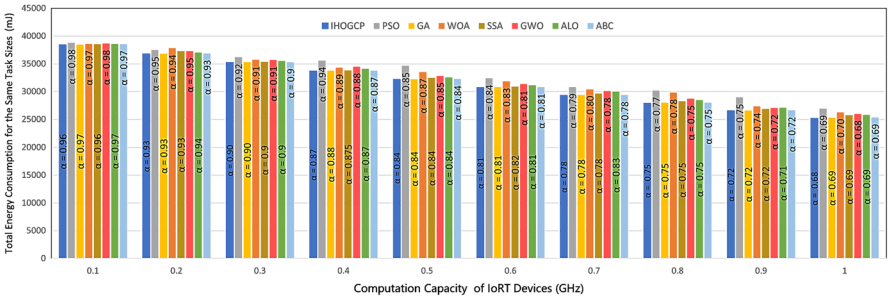


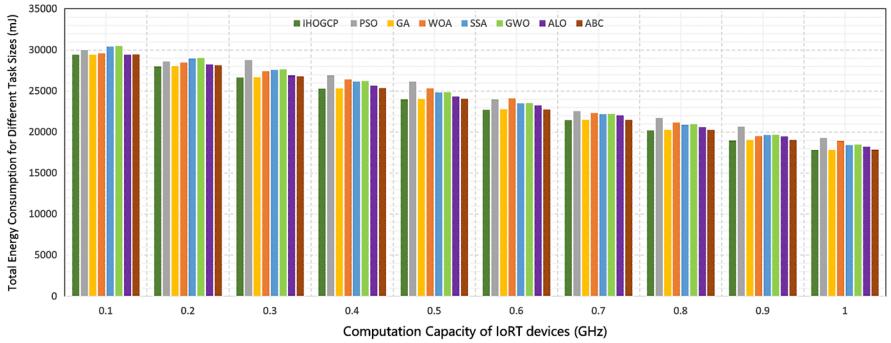
Fig. 13 Convergence comparison between IHOGCP, GA, PSO and WOA

Table 5 Offloading ratio and transmission rate comparison for eight IoRT devices using four different approaches

		EU_1	EU_2	EU_3	EU_4	EU_5	EU_6	EU_7	EU_8
SSA	Offloading ratio	0.50	0.49	0.69	0.66	0.59	0.64	0.68	0.63
	Transmission rate	2.50E+06	1.08E+06	2.65E+06	2.39E+06	1.88E+06	1.65E+06	2.60E+06	2.13E+06
	Total consumed energy	1.845E+04 mJ							
GWO	Offloading ratio	0.49	0.49	0.69	0.66	0.59	0.55	0.69	0.63
	Transmission rate	1.34E+06	2.56E+06	2.74E+06	2.43E+06	2.17E+06	1.63E+06	3.00E+06	2.29E+06
	Total consumed energy	1.838E+04 mJ							
ABC	Offloading ratio	0.49	0.49	0.68	0.66	0.59	0.55	0.69	0.64
	Transmission rate	1.09E+06	1.06E+06	2.60E+06	2.30E+06	1.66E+06	1.35E+06	2.61E+06	1.99E+06
	Total consumed energy	1.785E+04 mJ							
ALO	Offloading ratio	0.50	0.48	0.68	0.66	0.60	0.64	0.69	0.63
	Transmission rate	1.50E+06	1.08E+06	2.65E+06	2.39E+06	1.88E+06	1.65E+06	2.60E+06	2.13E+06
	Total consumed energy	1.819E+04 mJ							
WOA	Offloading ratio	0.51	0.52	0.69	0.67	0.60	0.55	0.69	0.66
	Transmission rate	1.80E+06	1.12E+06	2.59E+06	3.85E+06	1.98E+06	2.64E+06	2.85E+06	2.63E+06
	Total consumed energy	1.880E+04 mJ							
GA	Offloading ratio	0.50	0.50	0.69	0.66	0.60	0.55	0.69	0.63
	Transmission rate	1.07E+06	1.09E+06	2.60E+06	2.26E+06	1.63E+06	1.35E+06	2.62E+06	1.94E+06
	Total consumed energy	1.780E+04 mJ							
PSO	Offloading ratio	0.49	0.50	0.78	0.66	0.60	0.55	0.70	0.71
	Transmission rate	1.07E+06	1.13E+06	3.96E+06	2.26E+06	1.63E+06	2.83E+06	3.02E+06	3.09E+06
	Total consumed energy	1.930E+04 mJ							
IHOGCP	Offloading ratio	0.49	0.49	0.69	0.66	0.60	0.55	0.69	0.63
	Transmission rate	1.04E+06	1.04E+06	2.59E+06	2.27E+06	1.63E+06	1.34E+06	2.60E+06	1.94E+06
	Total consumed energy	1.780E+04 mJ							



(a) Impact of Computation Capacity on Offloading Ratio and Energy Consumption with Task Size 13 MB



(b) Impact of Computation Capacity on Energy Consumption in Different Schemas with Different Task Sizes

Fig. 14 Energy consumption versus local computation capacity of IoRT devices

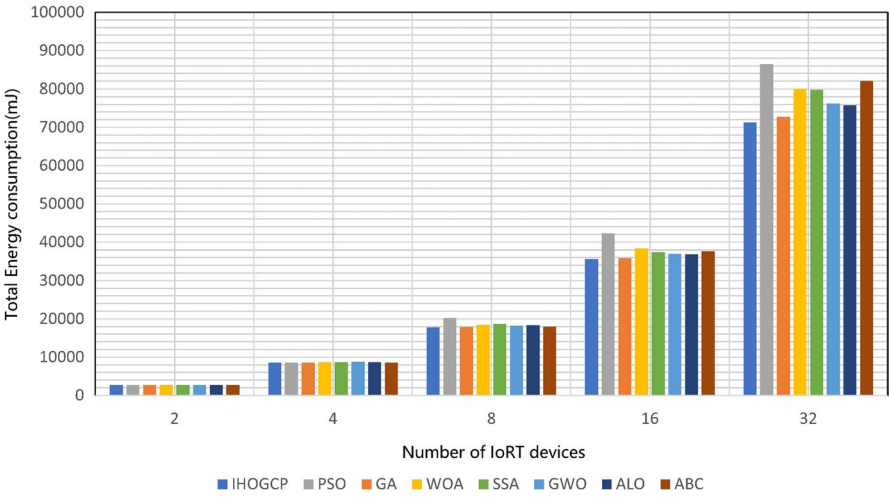


Fig. 15 Comparison of the energy consumption in different schemas under different numbers of IoRT devices

algorithm performs well under high computation capacity in terms of its speed of convergence and energy consumption.

Figure 15 shows the energy consumption of various schemas as a function of the number of IoRT devices, ranging from two to thirty-two. As illustrated in Fig. 15, IHOGCP has a low computational cost in comparison to other algorithms. It's easy to notice that as the number of IoRT devices decreases, the difference in energy consumption between the algorithms in comparison becomes nearly non-existent. Nevertheless, when the number of IoRT devices increases, the difference in energy consumption between the comparison algorithms becomes obvious and the proposed approach becomes more efficient than the others. This is because, according to PSO and WOA, the more devices present, the greater the likelihood of becoming trapped in the local optimum.

The elapsed time in the run for our proposed algorithm and the other three algorithms in different three scenarios is illustrated in Fig. 16. The three scenarios tested were: the first scenario, where the sub-areas were equipped with four IoRT devices, the second scenario where the sub-areas were equipped with eight IoRT devices, and the third scenario where the sub-areas were equipped with sixteen IoRT devices. Our proposed algorithm was evaluated against these three algorithms in terms of elapsed time during execution. The results showed that our algorithm consumed less elapsed time compared to the other three algorithms in all three scenarios tested. This indicates that our proposed algorithm is more efficient in terms of time consumption during execution than the other three algorithms.

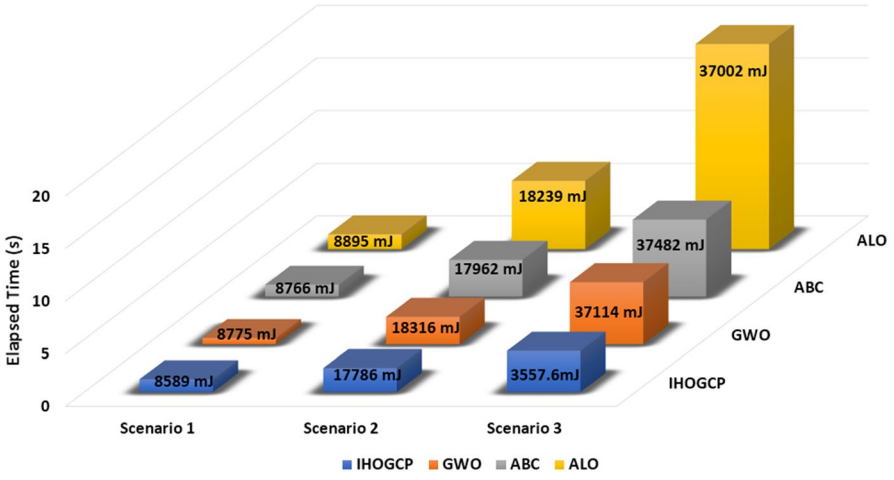


Fig. 16 Elapsed time in the run for four algorithms at different scenarios

9 Conclusion

In this research, a computation offloading strategy was presented to address the issue of IoRT devices consuming excessive amounts of energy for task computation while still fulfilling deadlines. Improving the task offloading ratio and transmission rate using an Improved Hybrid Optimization technique based on GA Combined with PSO named IHOGCP to reduce the total energy consumption. Based on the results of the simulations, it appears that the proposed technique can reduce IoRT devices' energy consumption under time limitations. Furthermore, simulation is used to evaluate the convergence of the proposed approach. The simulation results show that the proposed algorithm is able to outperform GA and PSO methods separately.

In future works, keep working to improve the computation offloading model by applying it to more realistic settings. Additional approaches for offloading optimization will be conducted as well. Furthermore, examining the difficulty of offloading tasks in a dynamic moving environment for user devices with inter-task dependencies.

Authors' contributions Conceptualization by HAA and NHE-M. Methodology and Analysis by AA-E, MMA, and NHE-M. Writing, review and editing by MSS and NHE-M. Analysis, Validation, Writing and editing by MMA and NHE-M. All the authors reviewed and approved this manuscript.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). The authors declare that they did not receive any funding for the current study.

Data availability We confirm that no datasets were users, generated, or analyzed during the current study.

Declarations

Conflict of interest The authors declare that they have no competing interests nor conflict of interests for the current study.

Ethics approval Not applicable.

Consent for publication Consent has been granted by all authors, and there is no conflict.

References

1. Khalid S (2021) Internet of Robotic Things: a review. J Appl Sci Technol Trends 2(03):78–90. https://doi.org/10.38094/jastt203104
2. Vojić S (2020) Internet of Robotic Things (IoRT) applications. STUME J 5(4):156–159
3. Heidari A, Jabraeil Jamali MA, Jafari Navimipour N, Akbarpour S (2020) Internet of Things offloading: ongoing issues, opportunities, and future challenges. Int J Commun Syst. https://doi.org/10.1002/dac.4474
4. Chen W, Yaguchi Y, Naruse K, Watanobe Y, Nakamura K, Ogawa J (2018) A study of robotic cooperation in cloud robotics: architecture and challenges. IEEE Access 6:36662–36682. https://doi.org/10.1109/ACCESS.2018.2852295
5. Yuan H, Zhou MC (2021) Profit-maximized collaborative computation offloading and resource allocation in distributed cloud and edge computing systems. IEEE Trans Autom Sci Eng 18(3):1277–1287. https://doi.org/10.1109/TASE.2020.3000946
6. Aazam M, Zeadally S, Harras KA (2018) Offloading in fog computing for IoT: review, enabling technologies, and research opportunities. Futur Gener Comput Syst 87:278–289. https://doi.org/10.1016/j.future.2018.04.057
7. Dehghan Shabani Z, Shahnazi R (2019) Energy consumption, carbon dioxide emissions, information and communications technology, and gross domestic product in Iranian economic sectors: a panel causality analysis. Energy 169:1064–1078. https://doi.org/10.1016/j.energy.2018.11.062
8. Zhang K et al (2016) Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks. IEEE Access 4:5896–5907. https://doi.org/10.1109/ACCESS.2016.2597169
9. Meng Y, Dai J (2020) Energy-efficient joint computation offloading and resource allocation in multi-user MEC systems. J Phys: Conf Ser. https://doi.org/10.1088/1742-6596/1693/1/012042
10. Guo F, Zhang H, Ji H, Li X, Leung VCM (2018) An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing. IEEE/ACM Trans Networking 26(6):2651–2664. https://doi.org/10.1109/TNET.2018.2873002
11. Peng K et al (2021) Joint optimization of service chain caching and task offloading in mobile edge computing. Appl Soft Comput. https://doi.org/10.1016/j.asoc.2021.107142
12. Gupta BB, Quamara M (2020) An overview of Internet of Things (IoT): architectural aspects, challenges, and protocols. Concurr Comput. https://doi.org/10.1002/CPE.4946
13. Thilakarathne NN, Yassin H, Bakar MSA, Abas PE (2022) Internet of Things in smart agriculture: challenges, opportunities and future directions. Institute of Electrical and Electronics Engineers (IEEE), pp 1–9. https://doi.org/10.1109/csde53843.2021.9718402
14. Kant K, Anand S, Sudeep Tanwar N, Abouhawwash M (2021) Application of Internet of Thing and cyber physical system in Industry 4.0 smart manufacturing. In Emergence of Cyber Physical System and IoT in Smart Automation and Robotics: Computer Engineering in Automation. Springer, pp 203–217. https://doi.org/10.1007/978-3-030-66222-6_14
15. Asghari P, Rahmani AM, Javadi HHS (2019) Internet of Things applications: a systematic review. Comput Netw 148:241–261. https://doi.org/10.1016/J.COMNET.2018.12.008
16. Alavi AH, Jiao P, Buttlar WG, Lajnef N (2018) Internet of Things-enabled smart cities: state-of-the-art and future trends. Measurement 129:589–606. https://doi.org/10.1016/J.MEASUREMENT.2018.07.067
17. Batth RS, Nayyar A, Nagpal A (2019) Internet of robotic things: driving intelligent robotics of future-concept, architecture, applications and technologies. In: Proceedings - 4th International Conference on Computing Sciences, ICCS 2018, Institute of Electrical and Electronics Engineers Inc., pp 151–160. https://doi.org/10.1109/ICCS.2018.00033
18. Vermesan O et al (2020) Internet of robotic things intelligent connectivity and platforms. Front Robot AI. https://doi.org/10.3389/frobt.2020.00104
19. Simoens P, Dragone M, Saffiotti A (2018) The internet of robotic things: a review of concept, added value and applications. Int J Adv Robot Syst. https://doi.org/10.1177/1729881418759424
20. Afanasyev I et al (2019) Towards the internet of robotic things: analysis, architecture, components and challenges. In: Proceedings - International Conference on Developments in eSystems Engineering, DeSE, pp 3–8. https://doi.org/10.1109/DESE.2019.00011
21. Romeo L, Petitti A, Marani R, Milella A (2020) Internet of robotic things in smart domains: applications and challenges. Sensors 20(12):3355. https://doi.org/10.3390/S20123355

22. Masuda Y, Shepard DS, Nakamura O, Toma T (2020) Vision paper for enabling internet of medical robotics things in open healthcare platform 2030. Smart Innov Syst Technol 192:3–14. https://doi.org/10.1007/978-981-15-5852-8_1
23. Masuda Y, Zimmermann A, Shirasaka S, Nakamura O (2021) Internet of robotic things with digital platforms: digitization of robotics enterprise. Smart Innov Syst Technol 189:381–391. https://doi.org/10.1007/978-981-15-5784-2_31
24. Ilori AO, Idowu OA, Mufutau WO (2018) Performance comparison of long term evolution (LTE) and third generation (3G) telecommunication systems. Adeleke University J Eng Technol 1(1):51–58
25. Yadav P et al (2021) Evolution of wireless communications with 3G, 4G, 5G, and next generation technologies in India. Springer 709:355–359. https://doi.org/10.1007/978-981-15-8752-8_35
26. Zhang Y, Weng J, Dey R, Fu X (2019) Bluetooth low energy (BLE) security and privacy. Encyclopedia of wireless networks. Springer, Switzerland. https://doi.org/10.1007/978-3-319-32903-1_298-1
27. Kalyani G, Chaudhari S (2020) Survey on 6LoWPAN security protocols in IoT communication. Lect Notes Electr Eng 601:696–702. https://doi.org/10.1007/978-981-15-1420-3_74
28. Ghanem K, Coffele F, Irvine J (2018) The reliability and optimal data usage of BGAN satellite communications for remote outstations. In: 2018 International Conference on Smart Communications and Networking (SmartNets), 2018 International Conference on Smart Communications and Networking (SmartNets), pp 1–5. https://doi.org/10.1109/SMARTNETS.2018.8707403
29. Lazaro A, Villarino R, Girbau D (2018) A survey of NFC sensors based on energy harvesting for IoT applications. Sensors (Switzerland). https://doi.org/10.3390/s18113746
30. Yadav S, Kumar K, Rashmi NA, Dixit A (2018) A review on overview of worldwide interoperability for microwave access (WIMAX) innovation and its future utilizations. ICTACT J Commun Technol 9(2):1749–1756. https://doi.org/10.21917/ijct.2018.0256
31. Naidu GA, Kumar J (2019) Wireless protocols: Wi-Fi SON, Bluetooth, ZigBee, Z-Wave, and Wi-Fi. Lect Notes Netw Syst 65:229–239. https://doi.org/10.1007/978-981-13-3765-9_24
32. Mekki K, Bajic E, Chaxel F, Meyer F (2018) Overview of cellular LPWAN technologies for IoT deployment: Sigfox, LoRaWAN, and NB-IoT. In 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). pp 197–202. https://doi.org/10.1109/PERCOMW.2018.8480255
33. Liu Y, Zhang W, Pan S, Li Y, Chen Y (2020) Analyzing the robotic behavior in a smart city with deep enforcement and imitation learning using IoRT. Comput Commun 150:346–356. https://doi.org/10.1016/j.comcom.2019.11.031
34. Mustafa Mohammed C, Zeebaree SRM (2021) Sufficient comparison among cloud computing services: IaaS, PaaS, and SaaS: a review. Int J Sci Bus. https://doi.org/10.5281/zenodo.4450129
35. Srinidhi NN, Dilip Kumar SM, Venugopal KR (2019) Network optimizations in the Internet of Things: a review. Eng Sci Technol Int J 22(1):1–21. https://doi.org/10.1016/j.jestch.2018.09.003
36. Kumar M, Dubey K, Pandey R (2021) Evolution of emerging computing paradigm cloud to fog: applications, limitations and research challenges. In: Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering, Institute of Electrical and Electronics Engineers Inc., pp 257–261. https://doi.org/10.1109/Confluence51648.2021.9377050
37. Elgendy IA, Zhang WZ, Zeng Y, He H, Tian YC, Yang Y (2020) Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks. IEEE Trans Netw Serv Manage 17(4):2410–2422. https://doi.org/10.1109/TNSM.2020.3020249
38. Gudi SLKC, Johnston B, Williams MA (2019) Fog robotics: a summary, challenges and future scope. ArXiv
39. Surati S, Patel S, Surati K (2021) Background and research challenges for FC for healthcare 4.0. Fog computing for healthcare 4.0 environments. Springer, Cham, pp 37–54
40. Kashani MH, Ahmadzadeh A, Mahdipour E (2020) Load balancing mechanisms in fog computing: a systematic review. ArXiv. https://doi.org/10.48550/ARXIV.2011.14706
41. Idrissi ME, Elbeqqali O, RIFfi J (2019) A review on relationship between Iot– cloud computing – fog computing (Applications And Challenges). In: 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS). pp 1–7. https://doi.org/10.1109/ICDS47004.2019.8942304

42. Chen S, You Z, Ruan X (2020) Privacy and energy co-aware data aggregation computation offloading for fog-assisted IoT networks. IEEE Access 8:72424–72434. https://doi.org/10.1109/ACCESS.2020.2987749
43. Sheng M, Dai Y, Liu J, Cheng N, Shen X, Yang Q (2020) Delay-aware computation offloading in NOMA MEC under differentiated uploading delay. IEEE Trans Wirel Commun 19(4):2813–2826. https://doi.org/10.1109/TWC.2020.2968426
44. Li Z, Zhu Q (2020) Genetic algorithm-based optimization of offloading and resource allocation in mobile-edge computing. Information (Switzerland). https://doi.org/10.3390/info11020083
45. Saleem U, Liu Y, Jangsher S, Tao X, Li Y (2020) Latency minimization for D2D-enabled partial computation offloading in mobile edge computing. IEEE Trans Veh Technol 69(4):4472–4486. https://doi.org/10.1109/TVT.2020.2978027
46. Li X, Zhang G, Zheng X, Hua S(2020) Delay optimization based on improved differential evolutionary algorithm for task offloading in fog computing networks. In: 12th International Conference on Wireless Communications and Signal Processing, WCSP 2020, Institute of Electrical and Electronics Engineers Inc., pp 109–114. https://doi.org/10.1109/WCSP49889.2020.9299850
47. Miao Y, Wu G, Li M, Ghoneim A, Al-Rakhami M, Hossain MS (2020) Intelligent task prediction and computation offloading based on mobile-edge cloud computing. Futur Gener Comput Syst 102:925–931. https://doi.org/10.1016/j.future.2019.09.035
48. Zhou S, Jadoon W (2020) The partial computation offloading strategy based on game theory for multi-user in mobile edge computing environment. Comput Netw. https://doi.org/10.1016/j.comnet.2020.107334
49. Sun W, Zhang H, Wang R, Zhang Y (2020) Reducing offloading latency for digital twin edge networks in 6G. IEEE Trans Veh Technol 69(10):12240–12251. https://doi.org/10.1109/TVT.2020.3018817
50. Cheng J, Guan D (2021) Research on task-offloading decision mechanism in mobile edge computing-based internet of vehicle. EURASIP J Wirel Commun Netw. https://doi.org/10.1186/s13638-021-01984-6
51. Han Y, Zhu Q (2022) Joint computation offloading and resource allocation for NOMA-enabled multitask D2D system. Wirel Commun Mob Comput 2022:1–14. https://doi.org/10.1155/2022/5349571
52. Hazra A, Donta PK, Amgoth T, Dustdar S (2022) Cooperative transmission scheduling and computation offloading with collaboration of fog and cloud for industrial IoT applications. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2022.3150070
53. Singh P, Singh R (2022) Energy-efficient delay-aware task offloading in fog-cloud computing system for IoT sensor applications. J Network Syst Manag. https://doi.org/10.1007/s10922-021-09622-8
54. Huang M, Zhai Q, Chen Y, Feng S, Shu F (2021) Multi-objective whale optimization algorithm for computation offloading optimization in mobile edge computing. Sensors. https://doi.org/10.3390/s21082628
55. Yan J, Bi S, Zhang YJ, Tao M (2020) Optimal task offloading and resource allocation in mobile-edge computing with inter-user task dependency. IEEE Trans Wirel Commun 19(1):235–250. https://doi.org/10.1109/TWC.2019.2943563
56. Gu X, Jin L, Zhao N, Zhang G (2019) Energy-efficient computation offloading and transmit power allocation scheme for mobile edge computing. Mob Inf Syst. https://doi.org/10.1155/2019/3613250
57. Abbas ZH et al (2021) Computational offloading in mobile edge with comprehensive and energy efficient cost function: a deep learning approach. Sensors. https://doi.org/10.3390/s21103523
58. Nguyen TT, Ha VN, Le LB, Schober R (2020) Joint data compression and computation offloading in hierarchical fog-cloud systems. IEEE Trans Wirel Commun 19(1):293–309. https://doi.org/10.1109/TWC.2019.2944165
59. Chang Z, Zhou Z, Ristaniemi T, Niu Z (2017) Energy efficient optimization for computation offloading in fog computing system. In: GLOBECOM 2017 - 2017 IEEE Global Communications Conference. pp 1–6. https://doi.org/10.1109/GLOCOM.2017.8254207
60. Hussein MK, Mousa MH (2020) Efficient task offloading for IoT-Based applications in fog computing using ant colony optimization. IEEE Access 8:37191–37201. https://doi.org/10.1109/ACCESS.2020.2975741
61. Al-Khafajiy M, Baker T, Al-Libawy H, Waraich A, Chalmers C, Alfandi O (2019) Fog computing framework for internet of things applications. In: Proceedings - International Conference on Developments in eSystems Engineering, DeSE, Institute of Electrical and Electronics Engineers Inc., pp 71–77. https://doi.org/10.1109/DeSE.2018.00017
62. Chen S, Zheng Y, Lu W, Varadarajan V, Wang K (2020) Energy-optimal dynamic computation offloading for industrial IoT in fog computing. IEEE Trans Green Commun Netw 4(2):566–576. https://doi.org/10.1109/TGCN.2019.2960767

63. Hou L, Zhang L, Kim J (2019) Energy modeling and power measurement for mobile robots. Energies (Basel). https://doi.org/10.3390/en12010027
64. Katoch S, Chauhan SS, Kumar V (2021) A review on genetic algorithm: past, present, and future. Multimed Tools Appl 80(5):8091–8126. https://doi.org/10.1007/s11042-020-10139-6
65. Jain M, Saihjpal V, Singh N, Singh SB (2022) An overview of variants and advancements of PSO algorithm. Appl Sci (Switzerland). https://doi.org/10.3390/app12178392
66. Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: whale optimization algorithm and its applications. Swarm Evol Comput 48:1–24. https://doi.org/10.1016/j.swevo.2019.03.004
67. Öztürk Ş, Ahmad R, Akhtar N (2020) Variants of Artificial Bee Colony algorithm and its applications in medical image processing. Appl Soft Comput J. https://doi.org/10.1016/j.asoc.2020.106799
68. Assiri AS, Hussien AG, Amin M (2020) Ant lion optimization: variants, hybrids, and applications. IEEE Access 8:77746–77764. https://doi.org/10.1109/ACCESS.2020.2990338
69. Mirjalili S, Aljarah I, Mafarja M, Heidari AA, Faris H (2020) Grey wolf optimizer: theory, literature review, and application in computational fluid dynamics problems. Studies in computational intelligence. Springer, Berlin, pp 87–105. https://doi.org/10.1007/978-3-030-12127-3_6
70. Abualigah L, Shehab M, Alshinwan M, Alabool H (2020) Salp swarm algorithm: a comprehensive survey. Neural Comput Appl 32(15):11195–11215. https://doi.org/10.1007/s00521-019-04629-4