Cluster Computing
https://doi.org/10.1007/s10586-023-03991-2

Task scheduling in the internet of things: challenges, solutions, and future trends

Tianqi Bu[1] · Zanyu Huang[2] · Kairui Zhang[3] · Yang Wang[4] · Haobin Song[5] · Jietong Zhou[6] · Zhangjun Ren[7] · Sen Liu[8]

Received: 11 December 2022 / Revised: 13 March 2023 / Accepted: 18 March 2023

Abstract
The Internet of Things (IoT) paradigm, by transforming physical devices into innovative items, affects every aspect of our daily lives. It has brought a slew of evolutionary and revolutionary services that were almost impossible to imagine until recently. The numerous services and applications offered by IoT cover numerous fields, such as personal healthcare, urban life, energy management, and manufacturing. A tremendous amount of data is produced to reach valuable information and meet users' needs. In addition, since the number of services and applications is increasing rapidly, an efficient method to fulfill the growing demands in different application domains becomes challenging. To handle the mentioned problems, task scheduling mechanisms have a significant influence. Despite the importance of these methods in the IoT, an in-depth and systematic review of the current works in this area is clearly lacking. Therefore, we aim to overcome this gap by adopting an organized manner. In fact, this paper aims to specify the challenging problems in IoT task scheduling, highlight the effective works, and outline some hints for upcoming studies.

Keywords Internet of Things · Meta-heuristic algorithms · Resource management · Systematic review · Task scheduling

1 Introduction

During the recent decade, scholars have gained much interest in the Internet of Things (IoT) because of its widespread applications in several real-life fields, particularly for critical environments such as smart homes [1], smart cities [2], and E-health [3]. The IoT brings out an environment in which the virtual and physical devices are equipped with processing, networking, detection, identification, and authentication functions to communicate with each other via the Internet to reach a common and meaningful purpose [4, 5]. The IoT permits home appliances to be more intelligent, everyday communication to be more meaningful, and routine processes to be more automated [6]. To extract useful information to enable intelligent and ubiquitous IoT services, different methods, such as data aggregation [7], service composition [8], load balancing [9], data fusion [10], service discovery [11], and data mining [12] are used. The IoT objects, employed in various applications, sense and gather information about

Tianqi Bu, Zanyu Huang, Kairui Zhang, Yang Wang, Haobin Song and Jietong Zhou have contributed equally to this paper.

✉ Zhangjun Ren
renzhangjun@njust.edu.cn

✉ Sen Liu
20203925@cqu.edu.cn

1 School of Mechanical and Power Engineering, NanJing Tech University, Jiangsu 211816, China

2 School of Economics and Management, Beijing University of Civil Engineering and Architecture, Beijing 102627, China

3 Department of Electrical and Computer Engineering, Queen's University, Ontario K7L 3N6, Canada

4 School of Public Administration, Huazhong Agricultural University, Hubei 430070, China

5 School of Urban Construction, Hainan Vocational University of Science and Technology, Hainan 571137, China

6 College of Computer Science and Engineering, Northwest Normal University, Gansu 730070, China

7 School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Jiangsu 210094, China

8 College of Optoelectronic Engineering, Chongqing University, Chongqing 400044, China

the physical world. In this regard, IoT applications are impacted by several factors, including users' requirements, energy limitations of devices, the heterogeneous nature of applications, the enormous need for communications, and limited computation power [13]. To tackle the mentioned problems, various types of algorithms have been introduced, such as Medium Access Control (MAC) scheduling [14], transmission power control [15], and task scheduling [16].

IoT applications produce numerous tasks of variable lengths that are often executed based on priority. Nevertheless, network endpoints possess heterogeneous characteristics and limited resources. These tasks compete for the limited resources available to heterogeneous devices in heterogeneous environments. Therefore, appropriate nodes should be assigned sufficient resources to execute these tasks in accordance with their resource requirements [17]. The optimal order of tasks on heterogeneous nodes with available resources can further enhance the performance of the task execution process and maximize resource utilization in terms of processor speed, bandwidth, memory, and minimize energy consumption, cost, and delay [18]. Different applications (delay-tolerant or latency-sensitive) submit heterogeneous tasks to IoT devices that are dynamic, vary in length, and are often prioritized. These tasks are queued to be executed on nodes with limited resources. In this regard, the need for an efficient, rapid, and convenient method of arranging these tasks in a manner that maximizes resource utilization and minimizes delay, cost, and energy consumption is of great importance [19].

As a matter of fact, task scheduling algorithms aim to schedule a broader range of tasks, such as transmission, processing, and sampling, on the IoT nodes considering the resources aiming to reach various purposes, including maximizing the operational lifetime of sensor nodes, reducing the communication costs among IoT objects, and improving resource utilization [20]. These approaches effectively distribute the existing tasks among computing resources to minimize inter-partition communication and computational latency [21]. Figure 1 illustrates the overall task scheduling process in the IoT environment. A centralized node receives requests from IoT end-users' applications in which a request evaluator assigns priorities to different tasks. Different parameters are considered when scheduling tasks, such as resource demand, communication demand, and computation size. The task scheduler schedules the tasks in a suitable sequence in which the tasks can be fulfilled under problem-specific constraints [22]. Figure 2 illustrates an overview of the task scheduling switcher that allows choosing the best scheduling method within the computation deadline for each task.

The IoT environment presents different scheduling challenges because of its heterogeneous nature, high scalability, and attributes associated with real-time computing, continuous processing, and shared sensing [23]. Traditional scheduling policies such as round-robin and First Come First Serve (FCFS) are inefficient enough to overcome this issue. In fact, these methods ignore the constraints mentioned above. Adopting a greedy scheme allows the efficient utilization of all available resources to obtain an assignment with minimal time constraints. Despite the most efficient results of this scheme, it remains impossible to perform all tasks on all devices simultaneously. The problem of discovering optimal solutions to these algorithms is NP-Hard since they suffer from high computational complexity [21]. Over the last decade, various efforts have been made and different task scheduling mechanisms have been introduced, including utilizing fog and cloud computing [24], meta-heuristic approaches [25], machine learning algorithms [26], etc.

Over the last decade, several review papers have been published to examine scheduling issues. Amalarethinam and Josphin [27] reviewed the existing task scheduling approaches in heterogeneous systems based on various factors. In addition, an in-depth evaluation of task scheduling methods in big data environments, such as Mesos, Storm, Spark, and Hadoop, is proposed by Soualhia, Khomh [28]. Energy-aware task scheduling mechanisms in cloud environments are reviewed by Hazra, Roy [29]. Ramezani, Naderpour [30] investigated meta-heuristic-based methods for cloud computing task scheduling. Furthermore, Amini Motlagh, Movaghar [31] systematically reviewed task scheduling methods in cloud computing. A detailed assessment of task scheduling techniques in fog computing is offered by Alizadeh, Khajehvand [32]. In addition, multi-objective scheduling methods designed for various cloud environments are reviewed by Hosseinzadeh, Ghafour [33]. Also, Yang and Rahmani [34] examined the task scheduling methods in fog computing in two main groups, heuristic and meta-heuristic. Matrouk and Alatoun [35] classified the present scheduling methods in fog computing into five classes, including workflow scheduling, job scheduling, resource allocation, resource scheduling, and task scheduling. Finally, Kaur, Kumar [36] categorized the task scheduling methods in fog computing into four categories: heuristic, meta-heuristic, deterministic, and hybrid.

Table 1 depicts a prominent picture of previous surveys, outlining the significant focuses and contributions of studies. The reviewed papers describe the task scheduling problem from different perspectives and provide a solid foundation for understanding various aspects of the problem. However, these studies have focused on fog, cloud, and big data environments; thus, IoT-based task scheduling

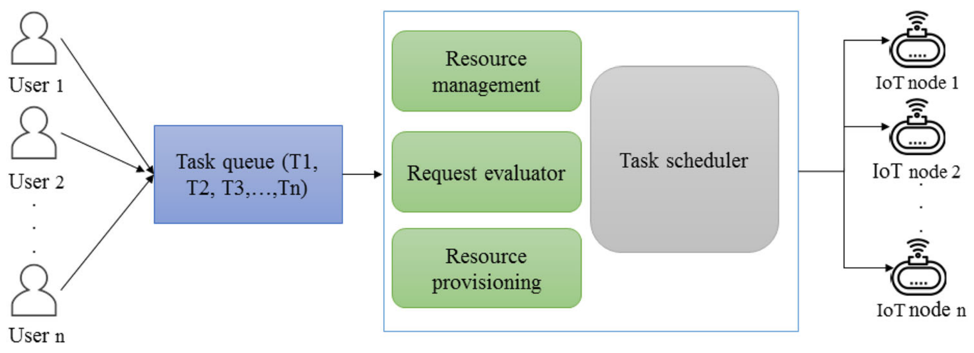

Fig. 1 Overview of task scheduling process in IoT

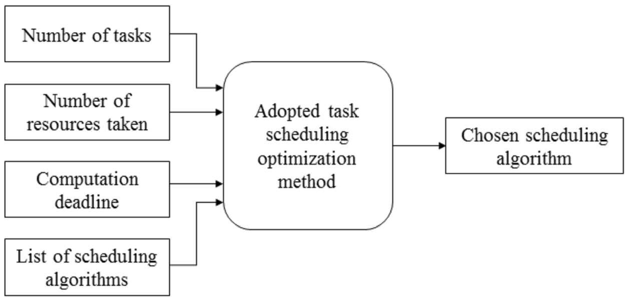
Number of tasks
Number of resources taken
Computation deadline
List of scheduling algorithms
Adopted task scheduling optimization method
Chosen scheduling algorithm

Fig. 2 Overview of the task scheduling switcher

approaches have been ignored. Therefore, we aim to cover this gap by adopting a systematic manner, aiming at reviewing the papers relevant to task scheduling in the IoT. A comprehensive assessment of existing task scheduling techniques in the IoT is presented in this paper. While conducting this systematic study, the primary methods are discovered, previous challenges are highlighted, state-of-the-art approaches are reviewed, and forthcoming research gaps are outlined. Concisely, this paper contributes to the following:

- Highlighting the current challenges in task scheduling in the IoT.
- Reviewing the proposed task scheduling methods in three groups, heuristic-based, non-heuristic-based, and machine learning-based, and determining the main strengths and weaknesses of methods.
- Discussing optimization models and describing their formulas.
- Demonstrating the impact of meta-heuristic algorithms in solving the task scheduling problem in the IoT and reviewing recent solutions.
- Determining the impact of IoT task scheduling on sustainable smart cities.
- Discussing open issues and outlining some potential directions for future research.

The paper is arranged into six sections. The next section discusses the research methodology. Section 3 reviews the current work in task scheduling in the IoT classifying them into four categories. Section 4 reports the obtained results regarding scheduling constraints, task-resource mapping, simulation tools and case studies, scheduling metrics, and the nature of tasks. In Sect. 5, open issues and directions for research are discussed. Finally, the conclusion is provided in Sect. 6.

2 Research methodology

In this section, we discuss the research methodology adopted to analyze existing task scheduling mechanisms in IoT. Our standards are based on the guidelines outlined by Kitchenham, Brereton [38], which many researchers have recently used to provide an accurate evaluation of the proposed methods related to various problems [39–41]. As shown in Fig. 3, the review procedure contains four major stages. The first stage involves defining the research objectives and questions. The relevant papers are searched and selected in the second stage, considering some criteria. We review the selected methods in the third stage based on some qualitative metrics. Finally, the last stage aims to report the obtained results, discuss unresolved issues, and suggest potential future research directions. To construct a Systematic Literature Review (SLR), fundamental research questions are required to drive the research methodology. Observing previous studies [9, 11], it is expected that the following research questions will be addressed during the methodological review.

Question 1: What are the significant challenging problems of task scheduling methods? This question specifies the issues addressed in previous methods and outlines the problems that have not been addressed.

Question 2: What are the evaluation parameters for analyzing and investigating task scheduling methods? This question is intended to identify the important qualitative metrics that can be used by researchers to measure their innovation performance.

Question 3: What is the significance of meta-heuristic algorithms in efficiently accomplishing task scheduling?

Table 1 Related surveys and their main contributions

Authors	Publication year	Main contributions
Amalarethinam and Josphin [27]	2015	Reviewing recent task scheduling techniques in heterogeneous systems considering various factors, such as efficiency, speedup, resource utilization, and makespan
Soualhia, Khomh [28]	2017	Reviewing and categorizing the proposed scheduling models for big data environments, such as Mesos, Storm, Spark, and Hadoop
Hazra, Roy [29]	2018	Discussing energy-efficient task scheduling methods in cloud computing in four main groups, including an energy-aware genetic algorithm, an energy-aware technique based on vacation queue, an online task scheduling technique based on DVFS, and scheduling high-performance computing tasks in decentralized cloud platforms
Ramezani, Naderpour [30]	2020	Highlighting the importance of evolutionary computation techniques in task scheduling in cloud environments
Amini Motlagh, Movaghar [31]	2020	Classifying task scheduling methods in cloud computing into three classes: a single cloud environment, multi-cloud environment, and mobile environment
Alizadeh, Khajehvand [32]	2020	Presenting a systematic study of task scheduling techniques in fog computing and reviewing existing works in four groups: heuristic, dynamic, static, and hybrid
Hosseinzadeh, Ghafour [33]	2020	Providing a comprehensive review of multi-objective workflow and task scheduling methods in cloud computing
Yang and Rahmani [34]	2020	Studying task scheduling methods in fog computing and classifying them into heuristic and meta-heuristic groups based on critical qualitative factors, including latency, performance, throughput, makespan, cost, efficiency, response time, and resource utilization
Matrouk and Alatoun [35]	2021	Reviewing task scheduling methods in fog computing in five classes, workflow scheduling, resource allocation, job scheduling, resource scheduling, and task scheduling
Kaur, Kumar [37]	2021	Categorizing the recent task scheduling methods in fog computing into four main classes, including heuristic, meta-heuristic, deterministic, and hybrid
Our study	2022	Discussing and reviewing the state-of-the-art task scheduling methods in the IoT, providing a comprehensive assessment, highlighting challenges, and outlining research gaps

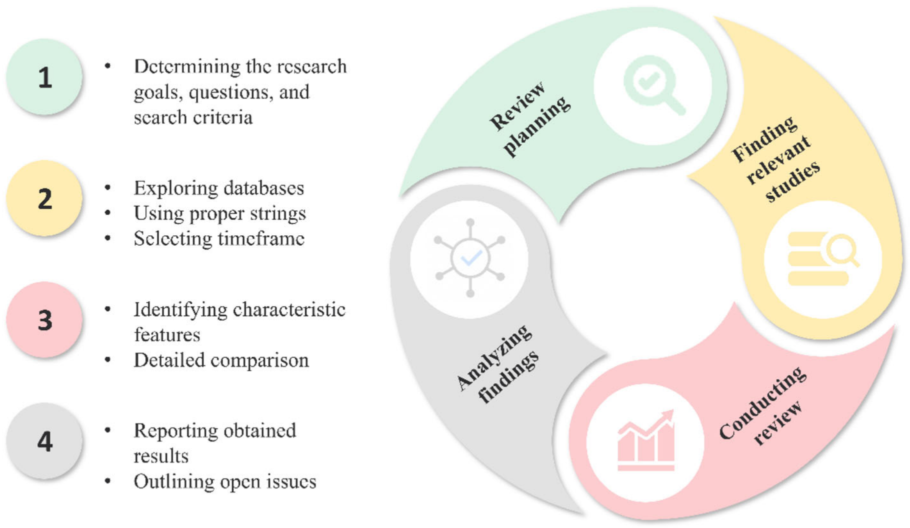

Fig. 3 The adopted research methodology

This study focuses on highlighting each meta-heuristic algorithm's importance in addressing the IoT task scheduling problem.

Question 4: What are the case studies adopted in task scheduling methods? Case studies are considered in this study to determine the feasibility of the task scheduling method in real-world situations.

Question 5: What are the most common simulation environments? This question aims to characterize the adopted simulation environments in the task scheduling methods.

We conducted an extensive and detailed search to review research papers of high repute. As shown in Table 2, four of the largest and most comprehensive databases in systems engineering, software engineering, computer engineering, and computer science were selected and searched. These electronic databases and indexing systems were chosen due to their high accessibility and their ability to export search results in standardized and computation-friendly formats. A further strength of these databases is that they are recognized as effective tools for conducting systematic literature reviews [42]. A search string was defined based on the keywords derived from the research questions. All databases were searched using the same search string according to necessary fields in digital libraries, including title, abstract, and keywords. Our search string is as follows:

("Internet of Things") AND ("Task scheduling").

An automatic search process for published papers between 2010 and 2022 based on the paper's title was done in November 2022, and 187 studies were found in journals, conferences, and books. Afterward, review papers, working reports, notes, and non-English studies were excluded from the review process to choose the highest-quality papers. In the final step, to select the proper studies for the review that are directly focused on the task scheduling problem, the authors carefully reviewed the full text of the remaining studies. Finally, 38 papers were selected. The search results included 11 papers in Web of Science, 7 in Scopus, 1 in ACM Digital Library, and 19 in IEEE Xplore Digital Library.

3 Review of IoT task scheduling mechanisms

This section presents an obvious trend of state-of-the-art works associated with task scheduling in the IoT by reviewing 38 studies in this field. The techniques selected in the previous section are classified into five key groups: traditional, heuristic-based, meta-heuristic-based, Reinforcement Learning (RL)-based, and Deep RL (DRL)-based. The innovation, strengths, and weaknesses of each method are outlined. Tables 3, 4, 5, 6, 7 show a brief investigation of methods involved in each group, in which the main idea, the research objective, the nature of tasks, the adopted simulation environment, the applied case study, and the weakness of each method are specified.

Figure 4 illustrates a taxonomy of task scheduling algorithms. The first type of algorithms are traditional algorithms reviewed in subSect. 3.1. The second type of algorithms are heuristic algorithms, which are used for solving optimization problems. An overview of existing heuristic algorithms is presented in subSect. 3.2. SubSection 3.3 discusses meta-heuristic algorithms for solving IoT task scheduling. Researchers have successfully applied RL algorithms to solve the task scheduling problem in dynamic and uncertain IoT environments. Algorithms based on RL are discussed in subSect. 3.4. Some researchers apply DRL-based algorithms in order to develop adaptive task scheduling algorithms suited to highly dynamic and unpredictable IoT environments. The DRL algorithm can also be classified as a value-iteration or policy-iteration-based algorithm, as discussed in subSect. 3.5.

As illustrated in Fig. 5, IoT scheduling targets can be categorized from two perspectives, end-user and provider. Service providers supply users with on-demand and leased resources. In contrast, these resources allow end-users to submit tasks for processing. Each party has its own motivation for participating in the IoT system. Providers are concerned about maximizing the efficiency of their resources, whereas end-users place a high priority on application performance. Service providers should pay more attention to parameters such as resource utilization, energy consumption, and load balancing. At the same time, users prefer to make their applications faster, so waiting time and makespan are more relevant for them.

- *Resource utilization:* It is calculated by dividing the number of computing units assigned to tasks by the total number of computing units requested. It can be calculated by Eq. 1, in which R_{avl} refers to available resources and R_{nu} denotes unused resources [43].

$$\mathrm{R_U = R_{avl} - R_{nu}} \tag{1}$$

- *Communication cost*: Communication costs refer to the expenses incurred in processing the requests of users. It can be calculated by multiplying the data set size by the network traffic price [44].

Table 2 Indexing systems and electronic databases

Name	Type	URL
Web of Science	Indexing system	www.webofknowledge.com
SCOPUS	Indexing system	www.scopus.com
ACM Digital Library	Electronic database	www.dl.acm.org
IEEE Xplore Digital Library	Electronic database	www.ieeexplore.ieee.org

Table 3 Overview of the discussed traditional task scheduling methods

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[52]	Proposing an efficient partial computation offloading and adaptive algorithm	Improving the average delay cost and offloading ratio	Independent	Anaconda 4.3	5G-enabled vehicular networks	Without evaluating resource utilization
[54]	Integrating unmanned aerial vehicles and mobile edge computing	Improving the system throughput while guaranteeing the fraction of served users	Independent	Real environment	5G-enabled unmanned aerial vehicles to community offloading	As the search space increases, trajectory design for the cooperation of multiple UAVs becomes more complex than for a single UAV
[55]	Developing a multi-objective distributed scheduling model based on multi-cloud and task schedulers	Minimizing energy consumption, improving resource utilization, and increasing cloud throughput	Workflow scheduling	N/A	Smart cities	It does not evaluate the communication cost
[56]	Developing an edge-enabled IoT system that incorporates cross-edge job processing	Improving the overall revenue and job completion ratio	Independent	Matlab	Wireless-powered mobile edge computing applications	Interconnections between tasks have been ignored
[57]	Providing a mixed-integer linear programming formulation	Maximizing system quality of security and minimizing system active energy consumption	Dependent	N/A	Heterogeneous multiprocessor system on a chip	Emerging dynamic tasks have been ignored
[58]	Sustainable energy harvesting for the IoT through adaptive task scheduling	Extending the battery life of IoT devices and lowering maintenance costs	Independent	Real-world prototype	Smart cities	Unable to schedule time-sensitive tasks

- *Reliability:* The term "reliability" refers to the guarantee of uninterrupted service for all users. This is an important indicator of performance, especially in applications such as Vehicle-to-Vehicle (V2V) communication and industrial automation, where an unreliable connection can lead to serious consequences [45].
- *Makespan:* The makespan refers to the total processing time of all tasks. In fact, it denotes the time span between the instant when the first task is scheduled and the instant when the last task completes execution [46]. It can be calculated by Eq. 2.

$$M = \max\{CT(t_i), t_i T\} \tag{2}$$

- *Load balancing:* Through load balancing, tasks are distributed uniformly among resources, ensuring both parallelization and utilization [47]. It can be calculated by Eq. 3 and Eq. 4. In these equations, n stands for the total number of resources, U_{av} refers to the average utilization of the resources, *U(R$_i$)* indicates the utilization of each resource, and *L(R$_i$)* denotes the load on the i^{th} resource.

$$U(R_i) = \frac{L(R_i)}{Makespan} \tag{3}$$

$$U_{av} = \frac{\sum_i U(R_i)}{n} \tag{4}$$

- *Average waiting time:* The average waiting time is the average amount of time a task spends waiting in the queue of each allocated resource as calculated by Eq. 5, in which n refers to the number of tasks and T_w represents the task waiting time function [48].

$$A_w = \frac{\sum T_w}{n} \tag{5}$$

- *Response time:* Response time is defined as the time required to complete a service request [49]. In fact, it is the sum of the service time and waiting time calculated by Eq. 6.

Table 4 Overview of the discussed heuristic-based task scheduling methods

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[51]	A dynamic programming algorithm for high-level task scheduling	Minimizing overhead	Independent	Real environment	Smart cities	It has not been tested on other IoT platforms outside of the three mentioned. Also, it has not been proven to be secure against potential attacks
[59]	Deadline and cost-aware scientific workflow	Improving success rate and makespan	Workflow scheduling	N/A	Wireless-powered mobile edge computing applications	More research is needed to test the algorithm on more complex workflows and to compare its performance against other algorithms
[60]	Online task scheduling and resource allocation for intelligent NOMA-based IoT	Cost reduction	Independent	Matlab	Fog enabled applications	It may not be feasible in practice due to the high computational complexity
[61]	Goal programming approach	Minimizing response time and cost	Independent	Matlab	Fog enabled applications	It requires a lot of computing power to run
[62]	Task scheduling for batteryless IoT devices	Reducing energy consumption	Independent	Real environment	Smart cities	It could add complexity to the system, and it may not be possible to achieve the same level of energy savings on all types of hardware

$$R_t = W_t + S_t \tag{6}$$

- *Throughput:* Throughput measures how much data is processed during a given period. This factor determines the reliability and performance of a system. It is calculated by the following formula, where δ refers to data size, N_d stands for the number of delivered task data, and t_s denotes the aggregate task simulation time [50].

$$T_p = \frac{\delta + N_d}{t_s} \tag{7}$$

- *Energy consumption:* It indicates the total power consumed by IoT resources when executing workflows, calculated by Eq. 8. T_k stands for the user's tasks allocated to a processing component (P_k) [51].

$$E_j = \sum_{k=1}^{N} (T_k \times P_k) \tag{8}$$

3.1 Traditional mechanisms

This subsection first discusses traditional scheduling algorithms and their strengths and weaknesses. Then, well-known traditional algorithms for scheduling IoT tasks are reviewed and compared. Static algorithms are based on the assumption that all information about a task and IoT resource is known in advance. These algorithms can be easily understood and implemented due to their simplicity. The most common traditional algorithms used to address the task scheduling problem include round-robin, FCFS, min-min, max–min, and Minimum Completion Time (MCT). FCFS and round-robin algorithms deal with a single machine while min–max and min-min algorithms handle multiple machines.

In FCFS, the order of tasks is determined by their arrival time. Tasks are placed at the tail of the task queue and then they are processed according to their entry order by the scheduler. A task with the earliest completion time is selected and executed by the MCT algorithm. The max–min algorithm executes large tasks on available machines first, so small tasks may suffer from starvation. The min-min algorithm selects and executes the smallest tasks on available machines. Consequently, large tasks are delayed for an extended period of time. After determining the shortest completion time for the task, the algorithm allocates resources to it. In round-robin, each task is assigned a small interval of time within a processor, called a time slot or time quantum. Prior to the expiration of the quantum, if a task has completed its CPU burst, it is preempted and the processor is assigned to the subsequent task in the queue. Tasks that fail to complete their burst are moved to the tail

Table 5 Overview of the discussed meta-heuristic-based task scheduling methods

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[63]	Combining genetic algorithm with several local search methods	Minimizing communication cost and response time	Independent	C	RFID devices	It has not been compared to previous works
[64]	Utilizing the genetic algorithm to model the multi-frame task scheduling problem in the heterogeneous embedded systems	Improving the success rate of tasks allocation	Independent	Matlab	Heterogeneous multiprocessor	It has ignored the interconnection between tasks
[65]	The adaptive double-fitness genetic task scheduling method	Decreasing communication cost and execution time	Independent	Not mentioned	Fog enabled applications	It suffers from high complexity and has neglected some vital metrics, such as resource utilization and energy consumption
[66]	Combining ACO and genetic algorithms	Reducing makespan and improving resource utilization	Dependent	C	Heterogeneous multiprocessor	It does not consider the prioritization of tasks
[67]	Introducing a deadline and mobility-aware task scheduling approach based on the ACO algorithm	Improving total profits and reducing delay	Independent	Not mentioned	Latency sensitive applications	Without considering dynamic changes of resources
[68]	Utilizing the max–min ant system to overcome the static task-graph scheduling	Minimizing makespan and improving resource utilization	Dependent	VB	Homogeneous multiprocessor	Focusing on small-sized task-graph input samples and ignoring the heterogeneous multiprocessor and many-core systems
[69]	Utilizing the PSO algorithm for task scheduling	Better search capability with high precision, improving resource utilization, and minimizing delay	Independent	Not mentioned	Survivability applications	It does not handle the interconnection between tasks
[70]	Introducing an ACO-based multi-task scheduling method for mobile crowdsensing service markets	Improving the workers' benefits in mobile crowdsensing service markets	Independent	Not mentioned	Mobile crowdsensing	It does not support the interconnection between tasks
[59]	Proposing a deadline and cost-aware task scheduling approach based on the genetic algorithm	Performance variation of VMs, acquisition delay, heterogeneous dynamics, and on-demand acquisition	Workflow scheduling	Not mentioned	Workflow applications	It does not evaluate the communication cost among VMs
[71]	Combining the ACO algorithm and priority non-preemptive method	Minimizing the average waiting time and turnaround time	Independent	Matlab	General	It does not address resource utilization
[72]	Utilizing the PSO algorithm to overcome the problem of resource management in both heterogeneous and homogeneous IoT-based environments	Minimizing makespan and improving resource utilization	Independent	Matlab	Logistics management applications	It has not addressed the prioritization of tasks

Table 5 (continued)

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[73]	Utilizing the PSO algorithm and fuzzy theory	Improving resource utilization	Independent	iFogSim	Smart cities	It does not consider energy consumption and communication cost
[74]	Introducing a novel method based on the marine predators algorithm	Minimizing carbon dioxide emission rate, flow time, makespan, and energy consumption	Independent	Java	Fog enabled applications	It has ignored the interconnection between tasks
[75]	Developing a novel mathematical formulation	Minimizing the energy consumption of fog nodes while meeting the QoS requirements of IoT	Independent	C + +	Fog enabled applications	IoT devices offload computation tasks to smart gateways without considering the decision-making process
[76]	Combining the salp swarm algorithm and modified manta-ray foraging optimizer	Improving the cloud throughput and reducing makespan time	Independent	Cloudsim	General	Unable to schedule time-sensitive tasks
[77]	Combining particle swarm optimization and genetic algorithms	Improving resource utilization and throughput, as well as decreasing request error rate and response delay	Independent	Matlab	General	It does not support the interconnection between tasks
[78]	Quasi-oppositional Aquila optimizer-based task scheduling technique	Reducing makespan and latency, as well as improving throughput and resource utilization	Independent	N/A	General	It has ignored the interconnection between tasks

Table 6 Overview of the discussed RL-based task scheduling methods

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[81]	Proposing a Q-learning technique to overcome the joint computation mode selection and processing order problem	Minimizing computational and queuing delays of devices and energy consumption	Independent	Not mentioned	Wireless-powered mobile edge computing applications	Without considering the prioritization of tasks
[82]	A novel task scheduling technique based on Q-learning with a global view	Improving the lifetime of the IoT network, reducing delay, and extending task scheduling success rate	Independent	Not mentioned	General	It does not consider the prioritization of tasks
[21]	A task scheduling policy based on reinforcement learning	Minimizing communication costs and improving resource utilization	Dependent	Not mentioned	Fog enabled applications	It has not evaluated energy consumption
[83]	Imitation learning-enabled approach	Satisfying task latency constraints and minimizing system energy consumption	Dependent	Python	5G-enabled vehicular networks	Poor ability to balance the workload

Table 7 Overview of the discussed DRL-based task scheduling methods

References	Main idea	Research objective	Nature of tasks	Simulation environment	Case study	Weakness
[88]	Introducing a deep reinforcement learning-based task scheduling method considering mobile edge computing and mobile blockchain	Improving generalization and the rate of arrived data packages	Independent	Not mentioned	Mobile blockchain applications	The interconnection among tasks has been neglected
[89]	Task scheduling problem in space-air-ground integrated network for delay-oriented IoT services	Minimizing energy consumption and task processing delay	Independent	Python	Space-air-ground integrated networks	It ignores the interconnection between tasks
[90]	Two-phase scheduling based on deep learning	Reducing the missed rate of tasks	Independent	Python	Fog enabled applications	Without evaluating energy consumption
[91]	Formulating the task scheduling process as a partially observable stochastic game	Reducing energy consumption and delay	Independent	Python	Serverless edge computing networks	It does not evaluate the communication cost
[92]	Formulating the online task assignment and scheduling problem as an energy-constrained Deep Q-Learning process as a kickoff	Reducing energy consumption and delay	Independent	Python	Fog enabled applications	The interconnection among tasks has been neglected
[93]	Developing distributed deep reinforcement learning	Achieving higher task satisfaction ratio	Independent	Not mentioned	NOMA-MEC	Without considering the prioritization of tasks

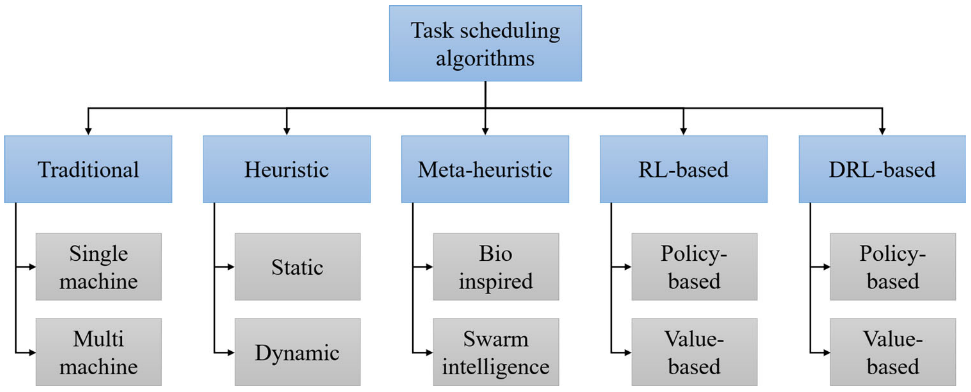

Fig. 4 Taxonomy of task scheduling algorithms

of the task queue. It is easy and simple to implement these static algorithms. A survey of the well-known traditional algorithms for scheduling IoT tasks is presented below.

Ning, Dong [52] have developed a partial computation offloading framework for vehicular networks equipped with Fifth Generation (5G). The framework enables vehicles to either process tasks locally or offloads them to Mobile Edge Computing (MEC) servers embedded within RoadSide Units (RSUs) through Non-Orthogonal Multiple Access (NOMA). NOMA represents one of the most significant air interfaces in IoT networks for maximizing spectrum utilization. It facilitates the connection of multiple IoT devices with a single frequency block, thereby enhancing IoT network connectivity [53]. In this paper, the task scheduling process is performed in three main phases, including channel resources allocation, offloading ratio decision, and offloading payoff policy for vehicles. A system-wide optimization problem that maximizes the profit of both the network operator and the vehicles is formulated by taking into account the mutual profits of the vehicle and the network operator. The original optimization problem is split into three subproblems due to the

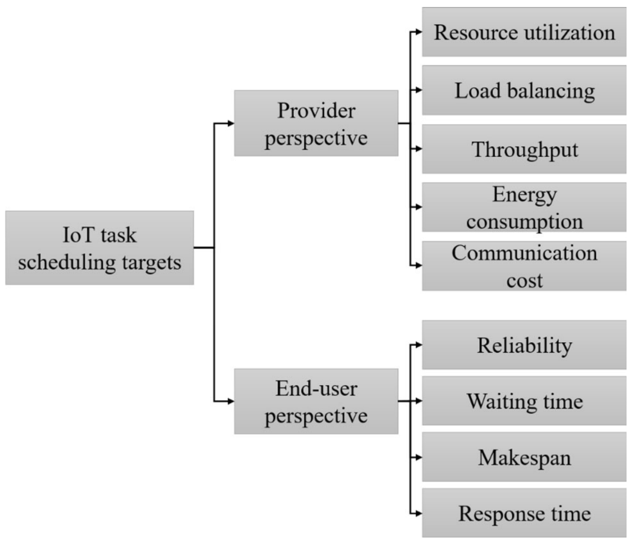

Fig. 5 IoT task scheduling targets

coupling of decision variables. First, a two-sided matching algorithm is developed to allocate channel resources by utilizing preference mapping and a list. Minimizing the cost of transmission delay is the sole objective of the first subproblem. A concave optimization problem is then formulated as the second subproblem. Karush–Kuhn–Tucker (KKT) conditions are applied to determine the offloading ratio. The Nash Equilibrium (NE) between vehicle profits and the profits of the network operator can be achieved since incentives are compatible and vehicles are rational. An adaptive non-cooperative game is therefore formulated to obtain an offloading payout policy. The real-world performance traces of taxis in Hangzhou (China) demonstrate that the proposed solution is effective from both system-wide and vehicle-level perspectives.

Diverse promising paradigms have brought remote servers closer to the edge of networks during the past two decades, such as MEC and Mobile Cloud Computing (MCC). MEC based on conventional base stations can accelerate complex computations, communications, and caching applications. However, these systems face three significant challenges, including the scarcity of wireless channel resources due to the increasing number of IoT devices, the limited coverage range of a fixed base station, and the latency sensitivity of application demands. Ning, Dong [54] propose the integration of UAVs and MEC to address these challenges by offering edge computing services to users through computation offloading. The deployment of 5G technology, along with several other effective spectrum-sharing technologies, enables cellular communications between UAVs and their users. The formulated problem is a Mixed Integer Nonlinear Program (MINLP) considering dependence within a community and independence among communities. By separating the variables of trajectory design and task scheduling decisions, the MINLP problem is divided into two subproblems. Relaxing the transmission rate constraint, a community-based latency approximation algorithm based on a piecewise function is proposed. The trajectory design subproblem is solved by developing an average throughput maximization-based auction algorithm. This paper also presents a method for solving the task scheduling subproblem in a community where transmission rates are constrained and tasks are atomic. The proposed algorithm maximizes the system throughput while guaranteeing a fraction of served users, and it can be used to achieve an appropriate trade-off between mobility efficiency and system throughput.

Cai, Geng [55] have developed a multi-objective distributed scheduling model based on multi-cloud and task schedulers in IoT, considering six objectives, including load balancing, resource utilization, energy consumption, cost, and response time. To implement the model, the authors developed a multi-objective intelligent algorithm based on the sine function, in which the variation tendency of diversity strategy in the population corresponds to the sine function. The sine function penalty strategy calculates the sine function value and multiplying it with the angle value to change the diversity proportion. Individuals with good performance are selected using the sind value as an evaluation criterion. Pareto-dominated and sine-function penalty selection strategies characterize this strategy. The authors discuss six objectives of task scheduling in a multi-cloud environment from both the user and provider perspectives. The total time and cost objectives include not only the calculation of the completion time and user costs of the task itself but also the transmission time and cost between multiple clouds and the data center. Throughput is a metric used to measure the efficiency of each cloud in terms of executing tasks and achieving more per unit of time. Energy consumption is influenced by the interests of providers. Resource utilization is the percentage of tasks that use the host resource, enabling tasks to be assigned to the appropriate resource. Finally, load balancing enables tasks to be distributed evenly among different virtual machines.

Huang, Li [56] have focused on the dynamic resource management and task scheduling problems in mobile edge computing to earn the best revenue for edge service providers. In contrast to most task scheduling and resource management algorithms formulated by an integer programming problem and solved in an NP-hard manner, this study investigates the problem structure in a novel way and identifies a promising property in the form of totally unimodular constraints. This property further helps design equivalent linear programming problems that can be solved efficiently and elegantly at polynomial computational complexity. The cross-edge job processing framework is formulated in an edge-enabled IoT system that adheres to a layered structure of edge computing. The mobility of devices, the overlapped signal coverage of edge base stations, the heterogeneous nature of tasks, and the computational resource capacity of edge servers have been considered. The proposed approach has been verified using simulations on a real-world dataset of base stations in the Melbourne CBD area and end-users.

Zhou, Sun [57] have studied the problem of scheduling tasks onto a heterogeneous multiprocessor system on a chip (MPSoC) deployed in IoT for optimizing the quality of security under task precedence, real-time, and energy constraints. To maximize system security, they propose a mixed-integer linear programming formulation to allocate and schedule tasks on a heterogeneous MPSoC system according to energy and real-time constraints. In order to efficiently solve the formulated problem, a two-stage analysis-based scheme is proposed, which determines the allocation, operating frequency, and security service of tasks to maximize system security qualities while satisfying design constraints. Simulator results show that the proposed two-stage scheme is more energy-efficient and secure than a number of existing representative approaches.

Energy harvesting-based IoT devices promise extended battery life, lowered maintenance costs, and a lower environmental impact. Despite this apparent potential, developing applications that ensure sustainable operation under variable energy availability and dynamic energy needs remains complex. An energy-aware task scheduler developed by Yang, Thangarajan [58] reduces this complexity by automatically adjusting the task execution rate to match the energy available in the environment. The proposed mechanism can prioritize tasks according to their importance, energy consumption, or a weighted combination of both. The approach differs from previous strategies in that it is autonomous and self-adaptive, requiring not a priori modeling of the environment or hardware platforms. This mechanism is more efficient than previous approaches in delivering sustainable operation for multiple tasks on heterogeneous platforms in dynamic environments. As calculated by simulations, the proposed mechanism has minimal performance overhead related to memory, computation, and energy, achieves Efficiency by allocating surplus resources according to the developer's priorities, keeps up with platform changes quickly, and ensures Sustainability by maintaining a user-defined optimal charging level. It is imperative to point out that the proposed approach can only be applied to tasks whose schedule can be independently controlled, such as sensing operations. The non-deterministic nature of its adaptive scheduler makes it unable to schedule time-sensitive tasks.

3.2 Heuristic-based mechanisms

This subsection presents a review and comparison of heuristic algorithms used by researchers to solve the IoT task scheduling problem along with their metrics. Scheduling problems can be optimized using heuristic algorithms, which aim to provide an optimal solution in a short period of time. These algorithms produce a sub-optimal but valid solution.

The majority of outdoor IoT applications rely on energy harvesting systems to ensure virtually uninterrupted operation. This is because these systems allow the devices to use energy from their environment, such as solar or wind energy, instead of relying on batteries. This reduces the need for maintenance and ensures that the devices can remain in operation for a long time. However, the use of energy harvesting raises issues regarding the efficacy of the application and the energy neutrality of the devices. In this context, Caruso, Chessa [51] present a novel dynamic programming algorithm suitable for optimizing the scheduling of tasks in IoT devices powered by solar panels. The problem is shown to be NP-Hard, and the algorithm obtains the optimal solution in pseudopolynomial time. The algorithm uses a two-dimensional array to store the optimal solutions for each subproblem. It then iterates through each subproblem to obtain the optimal solution for the entire problem. This makes it much more efficient than other existing algorithms, as it only needs to iterate a few times through the array instead of having to solve the entire problem from scratch each time. Furthermore, the algorithm is shown to be capable of being executed on three popular IoT platforms (namely TMote, Raspberry PI, and Arduino) with a small overhead. Moreover, the algorithm has been proven to be highly efficient and easily adaptable for use on multiple platforms, enabling its widespread adoption in IoT networks. However, the algorithm has not been tested on other IoT platforms outside of the three mentioned. Additionally, the algorithm has not been proven to be secure against potential attacks.

The deployment of large-scale IoT applications in cloud environments has become increasingly common in recent years. This has allowed for the development of highly efficient and powerful applications that can provide users with an improved user experience. Cloud computing features such as pay-as-you-go, unlimited expansion, and dynamic acquisition offer a number of conveniences for IoT applications. It is a challenging task to satisfy QoS requirements while allocating resources to tasks. This requires careful planning, efficient resource management, and a thorough understanding of the tasks and their respective QoS requirements. Ma, Gao [59] propose an algorithm that optimizes workflow execution while taking into accou1nt cost and deadline considerations. According to the topological structure of the virtual machine, they divide tasks into different levels in order to ensure that there is no dependency between tasks at a given level. This means that the tasks at any given level are independent and can be executed in any order without affecting the execution of the tasks at the other levels. This organization of tasks into different levels makes it easier to debug and maintain the virtual machine. The proposed algorithm uses three strings to code genes in order to better reflect the heterogeneous and resilient characteristics of cloud environments. The HEFT algorithm is then used to generate individuals in a timely and cost-effective manner. In order to increase the diversity of solutions, novel crossover and mutation schemes have been developed. The proposed algorithm has been tested on workflows that simulate the structured tasks of the IoT and performs well when compared with state-of-the-art algorithms. However, more research is needed to test the algorithm on more complex workflows and to compare its performance against other algorithms.

Fog computing is a distributed computing architecture that allows tasks to be processed closer to the data source, allowing for faster response times and decreased latency. This makes it an ideal solution for IoT tasks that require high levels of computation, since it can process data faster and more efficiently. The development of non-orthogonal multiple access (NOMA) has been recognized as a promising technique to improve spectrum efficiency in parallel with the development of fog computing. NOMA allows multiple users to access the same radio frequency resources simultaneously, thereby increasing the spectrum efficiency. This makes it possible for multiple users to access the same resources at the same time, which can reduce the latency of fog computing tasks and improve the response time. Additionally, the use of NOMA technology can also reduce the amount of energy consumed by fog computing nodes, thus making the system more efficient. Wang, Zhou [60] propose a NOMA-based fog computing framework for IoT systems, in which multiple task nodes offload their tasks to multiple nearby helper nodes. This allowed for the optimization of the total energy consumed by the system, as well as the overall completion time of the tasks. Additionally, this approach enabled the system to handle more task nodes than could be handled by traditional fog computing architectures. An online learning approach is used to solve the task scheduling and subcarrier allocation problems. An iterative algorithm is proposed as part of the online learning process to optimize both subcarrier allocation and task scheduling simultaneously. The results of the simulation indicate that the proposed scheme has the potential to reduce the sum cost significantly when compared to the baseline scheme. However, the proposed scheme may not be feasible in practice due to the high computational complexity. In addition, the proposed scheme may not be able to achieve the same performance when the channel conditions are not known in advance.

Fog computing can be an essential processing resource for IoT devices that have limited processing power. The goal of task scheduling in fog computing is to efficiently distribute tasks among the limited computing resources available, while taking into account the time constraints associated with tasks and the cost of running the tasks. This is done by analyzing the computing resources available, the type of task to be executed, and the cost of executing the task. Najafizadeh, Salajegheh [61] present a privacy-preserving architecture for scheduling IoT tasks. A multi-objective algorithm based on this architecture is presented in order to minimize service time and service costs. This multi-objective algorithm balances the trade-off between the two objectives and ensures that the privacy of the IoT tasks is guaranteed. The results showed that their proposed algorithm outperformed the existing algorithms in terms of both solution quality and computational efficiency when dealing with the four different scenarios. The goal programming approach (GPA) allows the algorithm to consider multiple objectives and find the optimal solution that satisfies all constraints. GPA uses a set of objectives and constraints to generate a set of feasible solutions, and then it uses a mathematical optimization technique to select the best one. Simulation results indicate that this algorithm has a much shorter runtime than other algorithms, meaning it is able to produce accurate results more quickly. Furthermore, it is also able to satisfy the privacy requirements of IoT devices, ensuring that user data is kept secure and private. However, there are some potential drawbacks to using this algorithm. One is that it is resource-intensive, meaning it requires a lot of computing power to run. This could make it impractical for use on devices with limited resources, such as IoT devices. Another potential drawback is that the algorithm is designed to work with a specific type of data, so it might not be able to handle other types of data that might be encountered in IoT devices.

As the demand for IoT devices increases, the need for safe and reliable energy sources grows. Batteries offer a stable energy source, but they can contain harmful chemicals, making them difficult to dispose of safely. Additionally, batteries are limited in their capacity and may need to be replaced or recharged regularly. This increases the cost of running and maintaining these devices, as well as the environmental impacts arising from the disposal of used batteries. Energy harvesters capture energy from their environment and convert it into electricity. This electricity is stored in capacitors and used to power battery less devices which can last longer and require less maintenance than those powered by batteries. A traditional computing scheduler cannot handle intermittent computing, so Delgado and Famaey [62] propose a first step toward the development of an energy-aware battery less device scheduler. An energy-aware task scheduling algorithm is presented that can optimize the scheduling of application tasks to avoid power failures and provides insight into the optimal look-ahead time for energy prediction. According to the results of the study, making the task scheduler energy aware prevents power failures and enables more tasks to be completed successfully. However, there are some potential drawbacks for implementing an energy-aware task scheduler. For example, it could add complexity to the system, and it may not be possible to achieve the same level of energy savings on all types of hardware.

3.3 Meta-heuristic-based mechanisms

Meta-heuristic algorithms are discussed in this subsection and are further classified into bio-inspired and swarm intelligence algorithms. Over the past few years, it has become increasingly popular to use meta-heuristic algorithms to solve complex computational problems. In particular, their adaptive nature makes them suitable for solving hard optimization problems that require creative solutions. Meta-heuristic algorithms can make the process of task scheduling under distributed computing conditions more efficient and reliable by identifying the optimal solution. In recent years, bio-inspired algorithms have gained increasing importance in solving engineering optimization problems. The simplicity of their implementation and superior performance have recently attracted the attention of researchers. They are robust and adaptive as they are inspired by the processes of nature. As global optimization tools, these algorithms have been applied to a variety of optimization engineering problems.

A variety of telecommunication applications have been explored using bio-inspired algorithms in recent years. Swarm intelligence-based computation is concerned with the collective behavior of self-organizing, decentralized systems. It is primarily derived from the behavior of certain animals and insects such as ants, termites, birds, and fish. By studying the behavior of these animals and insects, scientists have developed algorithms that mimic their behavior and can be used to solve complex problems. In swarm intelligence-based computation, multiple agents work together in a decentralized system to solve a problem or optimize a process. This robustness and adaptability enable swarm intelligence to be an ideal design paradigm when dealing with complex problems, such as IoT-based systems. Therefore, swarm intelligence is an extremely useful source of inspiration for the development of IoT-based systems that can be modeled as a swarm of simple devices or integrated with swarm intelligence-based algorithms in order to achieve a number of global objectives.

Given the NP-Hard nature of the task scheduling problem, various meta-heuristics have been developed to solve it over the last few years. For instance, to provide a solution to the task scheduling issue in the IoT, Li, Wang [63] have combined the Genetic Algorithm (GA) with several local search methods. Each solution of the proposed algorithm contains two vectors, the scheduling vector, which refers to the task scheduling sequence, and the RFID device assignment vector, which specifies the RFID device for processing the task. To reach the initial population with a high level of quality, some initial methods such as random rule, high workload first rule, least workload first rule, and lowest processing first are utilized. Generally, the proposed algorithm is done in the following manner. Several solutions are generated as an initial population. Each solution is evaluated and two solutions are selected randomly. Then, the crossover operator is performed on the two solutions and two new-generated solutions are replaced with the selected solutions. The mutation operator is performed on the two solutions, the newly generated solutions are assessed, and the best one is recorded. Using the C + + programming language, they tested the performance of their innovation. The most appropriate tasks are determined by applying the initial population, mutation, and crossover operator and considering the communication cost and task sequence constraints.

Task optimization problem in IoT-based embedded systems has been investigated by Wu [64]. The multi-frame task scheduling issue in heterogeneous embedded systems has been modeled to evaluate the availability of tasks using GA. The following assumptions have been adopted in the proposed model. To solve the task scheduling problem at runtime, tasks are assigned to processors and then used in a single processor. The non-divided execution mode enables executing tasks on one processor and another part on other processors. There is no dependency on tasks. This means that resource sharing is not possible. The periodic task model assigns tasks with varied execution times to multiple processors without considering resource sharing. Communication among tasks has been ignored as well. The interconnection among tasks should be considered when scheduling tasks and assigning them to multiple processors to obtain an optimal allocation scheme.

Utilizing fog computing, Liu, Wei [65] have proposed a scheduling algorithm called Adaptive Double-fitness Genetic Task Scheduling (ADGTS). They aim to diminish communication cost and execution time by operating IoT objects with higher communication, memory, and processing capacities as fog nodes. Task allocation to fog nodes is performed considering multiple parameters, such as communication capacities, delay requirements, and computing power. They use an adaptive mechanism for the mutation operator, a traditional single-point algorithm for the crossover operator, and the roulette selection method for the selection operator. The fitness function contains the communication cost and the makespan of the task execution. Compared to the conventional Min–Min methodology, the suggested approach outperforms in terms of communication cost and makespan. However, its high complexity remains a problem. Moreover, some important indicators, such as resource utilization and energy consumption, have not been investigated.

To obtain an efficient task scheduling solution in an IoT-based heterogeneous multiprocessor cloud environment, Basu, Karuppiah [66] have proposed an intelligent bio-inspired method. Combining GA and ACO algorithm, they implemented a hybrid algorithm that selects the most effective combination of tasks at each stage. The proposed approach ensures the appropriate convergence to reach an optimal solution. It considers the dependency among tasks to minimize the total execution time (makespan) of the tasks. The proposed technique outperforms ACO and GA in heterogeneous multiprocessor environments, but it has not specified a method for task prioritization.

To address the application deadlines and IoT device mobility, Fan, Liu [67] have proposed a deadline and mobility-aware task scheduling approach that uses the prediction of location based on the Dynamic Pattern Tree (DPT). An improved ACO with links' priorities has been developed to solve the scheduling problem as a multidimensional 0–1 knapsack problem. Several simulations utilizing real-world mobility trace were performed to demonstrate the effectiveness of the proposed mechanism, and the obtained results indicate it is more efficient than previous works regarding deadline-guarantee ratio and total profits. However, the dynamic changes of resources have been neglected.

To overcome the static task-graph scheduling in homogeneous multiprocessor environments and offer a high-performance method, Boveiri, Khayami [68] have proposed a novel method based on Max–Min Ant System (MMAS). To optimize the scheduling efficiency and robustness of multiprocessor task graphs and obtain the most optimal task order, they principally aim to manipulate the priority values of requests. Using background knowledge of the issue as heuristic values, the proposed approach has become an efficient and robust method. To verify the efficiency of the method, they utilized various random task graphs with several shape parameters, and the obtained results confirm its better performance than traditional methods. However, they have just focused on small-sized task-graph input samples in the cloud. Moreover, they have ignored the heterogeneous multiprocessor and many-core systems.

To optimize queue management while guaranteeing the QoS levels, Al-Turjman, Hasan [69] have presented an efficient scheduling approach developed based on Particle Swarm Optimization (PSO) algorithm. The suggested approach integrates cloud computing and IoT, where users' applications in various real-world domains are scheduled optimally. They have used the PSO algorithm to handle massive incoming tasks suitable for Dynamic Dedicated Server Scheduling (DDSS) and Heterogeneous DDSS (h-DDSS). Fully Informed PSO (FIPSO) and Canonical PSO (CPSO) algorithms have been utilized to ensure QoS in survivability applications. These algorithms prioritize the available cloud resources and incoming requests, considering different data traffic classes. Three strategies have been proposed to update particle velocities and improve convergence: service rate, arrival rates, and system utilization ratio. The simulation results prove that the FIPSO algorithm is superior to the CPSO algorithm with regard to delay and throughput, but it does not handle the interconnection among tasks.

Li, Jia [70] have utilized the ACO algorithm to provide a multi-task scheduling approach aiming to improve the workers' benefits in mobile crowdsensing service markets with IoT. Mobile crowdsensing is a novel sensing mode for IoT that contains three main components platform, workers, and requesters. Since each component in the crowdsensing market attempts to reach more benefits, the need for different task assignment methods to fulfill the different needs of each component becomes a challenging problem. The authors of the paper have introduced a novel task scheduling method from the workers' perspective. A theoretical examination of the benefits calculation model was performed to explore the factors affecting workers' income. Moreover, by utilizing the STSP dataset available online in multiple experiments, the proposed approach was tested to improve workers' productivity and reduce the cost of completing multiple tasks. Nevertheless, the suggested technique does not support the interconnection among tasks.

A deadline and cost-aware task scheduling approach based on GA has been proposed by Ma, Gao [59]. This method relies on key cloud characteristics, such as performance variation of Virtual Machines (VMs), acquisition delay, heterogeneous dynamics, and on-demand acquisition, to reduce costs under time constraints. The tasks are assigned to proper VMs using heuristic operations. The simulation outcomes show that the suggested algorithm obtains low execution costs and provides the highest success rate under various deadline constraints. However, it has not evaluated the communication cost among VMs.

Combining the ACO algorithm and priority non-preemptive method, Prasanth, George [71] have proposed a hybrid task scheduling approach to prioritize the tasks and discover the most effective path to assign them to the relevant IoT node. The suggested mechanism consists of two main modules, task pre-processor and task path selection. The task pre-processor module, available inside the IoT gateway, handles the preprocessing of individual tasks based on the user's query. This module separates the user queries into multiple tasks and finds the proper sensor nodes to execute them. In this regard, there are two main types of tasks, spatial and temporal tasks. The spatial tasks include the tasks that need information about a specific location, and the temporal tasks contain the tasks that have limited time to complete. The second module, task path selection, is responsible for forwarding the task packets among the task group and scheduling them by considering their delay parameter. Regarding average waiting time and turnaround time, the proposed algorithm outperforms the traditional ACO algorithm but does not improve resource utilization.

To overcome the resource management problem in homogeneous and heterogeneous IoT-based environments and provide an effective task scheduling approach in IoT-enabled smart job-shop, Hasan and Al-Rizzo [72] have utilized the CPSO algorithm. The proposed method aims to reduce the makespan and improve resource utilization compared to previous works. The decision mode used in the scheduling algorithm is based on dynamic scheduling rules. There are several unrelated machines that control the defined framework comprising VMs. According to their assumptions, each task in any VM has a unique processing time. The users' requests are sorted based on the processing time, which is estimated or specified. Then, the objective function is evaluated aiming to select the best solution with a minimum makespan. However, there is no method adopted for task prioritization.

Utilizing the PSO algorithm and fuzzy theory, Javanmardi, Shojafar [73] have proposed a fog-based task scheduler in which the observations relevant to application loop delay and network utilization have been considered. The proposed approach is suitable for both delay-sensitive and delay-tolerant applications. They aim to introduce a novel method for optimally fog-based resources to enhance network utilization and decrease the application loop delay. In this regard, the computing features of the fog nodes, such as bandwidth, RAM size, and CPU processing capacity, together with the tasks' features, such as CPU need, have been considered. In the case of fog-layer overloading, the class of each application is taken into account when making scheduling decisions. The proposed task scheduling method optimally uses the fog resources, solves fog task scheduling problems, and overcomes the local minimum problem, but it does not consider energy consumption and communication cost.

Abdel-Basset, Mohamed [74] have suggested an energy-aware task scheduling method based on Marine Predators Algorithm (MPA). Besides, the other two versions of MPA have been proposed. In the first version, Modified MPA (MMPA), MPA is modified to use the most recently updated locations rather than the most recent best locations to improve exploitation capabilities. The second version improves the MMPA by re-initializing and mutating toward the best strategy based on the ranking strategy. The proposed approach outperforms the previous regards carbon dioxide emission rate, flow time, makespan, and energy consumption, but it has ignored the interconnection among tasks.

The high distribution level, dynamic nature, and resource limitations of fog computing make it challenging to deploy fog computing for heterogeneous and delay-sensitive IoT tasks. To meet the QoS requirements of IoT and reduce the overall energy consumption of fog nodes, Azizi, Shojafar [75] have developed a novel mathematical formulation. The proposed model also attempts to minimize deadline violation time. Two semi-greedy-based algorithms have also been introduced to efficiently map IoT tasks to fog nodes, including priority-aware semi-greedy and priority-aware semi-greedy with the multi-start procedure. In contrast to greedy heuristic algorithms, semi-greedy approaches avoid local optima due to randomness, which is better for achieving quality results. Experimental results indicate that the suggested algorithms meet the deadline requirement for most IoT tasks, and the rest continue to receive responses with a little time.

Attiya, Abd Elaziz [76] have introduced a novel way of organizing and scheduling IoT applications within a cloud computing environment. To solve the problem of scheduling IoT tasks in cloud computing, they have proposed a hybrid swarm intelligence approach, combining the Salp Swarm Algorithm (SSA) and a modified Manta-Ray Foraging Optimizer (MRFO). The exploitation ability of MRFO is improved through the use of SSA operators. The proposed approach includes two main phases, namely, the initial phase and updating phase. In the first phase, a population of initial values is formed, representing a solution to the task scheduling problem regarded as a discrete optimization problem. MRFO was originally developed to address continuous optimization problems, while task scheduling is a discrete optimization problem. This limitation is tackled by initially producing a population. The second phase aims to update solutions in the following manner. First, the fitness values for solutions are computed. Then, a solution with the best fit (smallest makespan value) is determined. In the next step, operators of MRFO are used to update the solution during the exploration stage. The experimental results show that the proposed method outperforms its competitors on various performance metrics, including makespan time and cloud throughput.

Bu [77] has proposed a multi-task equilibrium scheduling method combining genetic and PSO algorithms. Server node load is measured using CPU, memory, network bandwidth, and disk input and output occupancy rates. These measures establish a resource balance model that measures the server node load. Weight adjustment is based on the fitness function used to quantify the influence. The PSO algorithm then applies the contraction operator and disturbance factor. The optimal algorithm aids in calculating the optimal fitness function and determining the optimal weight. The simulation results confirm that the proposed hybrid algorithm outperforms the traditional method in terms of resource utilization, request error rate, throughput, and response delay by more than 5%. The proposed mechanism considers various request types but does not mix them together.

A novel quasi-oppositional Aquila optimizer-based task scheduling (QOAO-TS) method has been proposed by Kandan, Krishnamurthy [78], in which an Aquila optimizer (AO) and quasi-oppositional-based learning have been integrated. Aquila's behavior while catching prey stimulates the traditional AO, and the QOAO was developed to enhance that behavior. The QOAO-TS approach satisfies the makespan requirements by achieving the best task scheduling process. It considers the relationship between tasks and minimizes the makespan to meet the client's needs. The results of multiple simulations have been investigated in terms of utilization ratio, latency, flow time, throughput, and makespan.

3.4 RL-based mechanisms

The RL algorithm is a machine intelligence algorithm designed for learning under highly dynamic and uncertain conditions. RL algorithms allow machines to learn from their experiences and make decisions based on the information they have gathered. This helps machines better adapt to changing environments and makes them more efficient and accurate at their tasks [79]. By interacting with a stochastic environment, RL proposes a computational approach for an agent to learn the appropriate behavior to accomplish its objective. Through experimentation and learning, the agent develops a policy that maximizes the expected long-term reward, which is known as the optimal policy. As the agent interacts with the environment and the environment changes, the agent is able to adapt its policy to achieve the best possible outcome. As a consequence, the purpose of RL is to enable the agent to learn appropriate policies by mapping states to actions in order to maximize long-term benefits [80]. It aims to facilitate the self-decision of IoT nodes for several networking functions, including routing, scheduling, resource allocation, dynamic spectrum access, energy management, mobility, and caching. By using RL, the agent can learn from the environment, form a model of the environment, and then optimize its decisions accordingly. This allows it to make decisions that optimize for the networking goal, such as minimizing energy usage or improving security levels.

Xie, Wang [81] have proposed a Q-learning technique to overcome the joint computation mode selection and processing order problem in a wireless-powered mobile edge computing IoT network. They have formulated the problem as an optimization problem aiming to reduce computational delays in devices and energy consumption. The proposed method can be implemented in multi-user systems and dynamically modify scheduling processes by considering diverse network setups. The results confirm that it performs better than the conventional benchmark methods and achieves up to 15.9% and 55.1% gain compared to local-computing and offloading methods, respectively.

Furthermore, to improve the lifetime of IoT networks, reduce delay, and extend task scheduling success rate, Ge, Liu [82] have proposed a novel task scheduling mechanism based on Q-learning with a global view named (QFTS-GV). At first, they define the Q-learning framework, including the rewards function, action set, and state set in a global view that forms the basis of the proposed approach. Then, they establish a task scheduling policy with determining rewards for nodes in different areas of the network. The transmission power of energy-relaxed nodes can be increased to promote the benefits of the whole network. The energy-strained nodes can be protected to improve the network lifetime. The proposed approach saves energy consumption, reduces delay, and improves the task scheduling success rate compared to previous Q-learning-based task scheduling schemes.

Pandit, Mir [21] have developed a task scheduling policy based on RL. They aim to decrease communication costs during distributed execution and obtain an efficient method with the minimum time to execute tasks as well as optimal resource utilization. In this regard, a two-level Neural Network (NN)-based task scheduling system has been introduced, in which the first-level NN specifies whether the data stream could be directly forwarded to the cloud or analyzed in the resource-constrained environment. The second-level NN is responsible for scheduling all the tasks sent by level 1 NN to the fog layer. The suggested mechanism handles task scheduling in real-time fog-based IoT with the least communication cost, minimum makespan, and best resource utilization.

In Vehicle Edge Computing (VEC), computing-intensive tasks can be moved to network edges and enhanced to overcome the challenges associated with the Internet of

Vehicles (IoV). Wang, Ning [83] conducted research on energy-efficient policies for vehicles and network operators in VEC networks because of the huge energy demands. The authors propose a task scheduling algorithm for online VEC that incorporates imitation learning. In imitation learning, the agent mimics the expert's demonstration (or reference policies) in order to solve the original problem effectively. The VEC model is established, and the task scheduling issue is formulated as an optimization problem. By acting as routers or gateways, RSUs facilitate the scheduling of tasks among different servers. The proposed solution is an infrastructure-free approach, as it does not require computational tasks to be cached or handled by RSUs. The service abilities of Service Demanding Vehicles (SPVs) are analyzed in the coverage of RSUs and are clustered by integrating their idle resources. Each RSU maintains cluster-level information, while the SPV-level information is located within its own cluster. This reduces the overhead in communications between the SPVs and the RSUs. The designed method conquers the disadvantages of traditional algorithms, including the slow convergence speed and the low searching efficiency. To solve the formulated task scheduling problem, it is decomposed into two sub-problems, resource aggregation and task scheduling, to assign tasks to suitable SPV clusters. The first subproblem is solved by analyzing the service abilities of local SPVs in the coverage of RSUs. The second subproblem is handled by imitating the expert's policy for cluster-based task scheduling, in which the convergence time can be largely reduced. A theoretical analysis of the performance of the algorithm has shown it to be able to achieve an acceptable performance gap from that provided by the expert.

3.5 DRL-based mechanisms

Due to the high dimensionality of IoT states and actions, traditional RL techniques are limited in terms of computation complexity and convergence to poor policies [84]. So DRL techniques, a combination of RL and Deep Learning (DL) approaches based on Artificial Neural Networks (ANN), are developed to overcome these limitations and improve learning and decision operations [85]. In recent years, DL techniques have been successfully applied to data analysis, natural language processing, sequence prediction, and image processing [86, 87].

Besides, to improve the computational ability of IoT devices, Gao, Wu [88] have considered mobile edge computing and mobile blockchain, which have been deployed at the Small-cell Base Station (SBS) as a supplement. They have considered the long-term revenue of the SBS to encourage SBS involvement in mobile blockchain networks. They have formulated the problem of task scheduling as a Markov Decision Process (MDP) to minimize the resource cost of the SBS and maximize the long-term mining reward. Moreover, they have proposed a deep reinforcement learning-based solution named Policy Gradient-based Computing Tasks Scheduling (PG-CTS) algorithm to obtain an efficient intelligent strategy. Utilizing a deep neural network, the policy mapping from the system state to the task scheduling decision is specified. To train the policy network, episodic simulations have been built, and the REINFORCE algorithm with the baseline has been utilized. The simulation results prove the generalization ability and effectiveness of the proposed approach.

Moreover, Zhou, Wu [89] have investigated the task scheduling problem in the space-air-ground integrated network for delay-oriented IoT services. Utilizing an Unmanned Aerial Vehicle (UAV), the computing tasks are collected from IoT devices and then online offloading decisions are made. They aim to introduce a task scheduling policy to reduce the computing and offloading delay of all tasks given the UAV energy capacity constraint. In this regard, the online scheduling problem has been formulated as an energy-constrained MDP. Then, a novel deep risk-sensitive reinforcement learning algorithm has been developed according to the dynamic nature of tasks. Compared to probabilistic configuration techniques, the proposed mechanism reduces the task processing delay by up to 30% and overcomes the UAV energy capacity constraint.

The proposed task scheduling method by Shadroo, Rahmani [90] includes two main phases. A clustering approach is used to specify the location of task execution. The second phase contains task scheduling based on the execution location. The clustering part involves three main concepts based on the Self-Organizing Map (SOM) clustering approach. The first and second concepts utilize the SOM and hierarchical SOM to separate the tasks' features from the IoT layer into various clusters. In the third concept, the features of tasks are extracted and their dimensions are decreased using the Autoencoder. The simulation results confirm that the proposed approach has reduced the missed rate of tasks in the fog and cloud and their costs.

Tang, Xie [91] have discussed the distributed task scheduling for the IoT in serverless edge computing networks in which the nodes are heterogeneous and rational individuals who desire to optimize their own scheduling utility. In contrast, the nodes only have local observations available. As a partially observable stochastic game (POSG), the task scheduling competition process enables serverless edge computing nodes to schedule tasks and allocate computing resources based on their locally observed system state. This system state considers the associated task generation state, data queue state, communication channel state, and the previous resource allocation state. An algorithm for multi-agent task scheduling, based on the dueling double deep recurrent Q-network (D3RQN) method, has been developed to approximate the optimal task scheduling and resource allocation solution for the proposed POSG.

Sellami, Hakiri [92] have investigated a low-latency and energy-aware oriented computing task scheduling problem in a Software-Defined Fog-IoT Network. First, the task scheduling and assignment problems are formulated as an energy-constrained Deep Q-Learning process. Under the constraints of application dependence, the latter seeks to minimize network latency while ensuring energy efficiency. A DRL approach is then proposed for dynamic task scheduling and assignment in SDN-enabled edge networks. Experimentally, the introduced algorithm has been compared to three pioneering deep learning algorithms, namely A3C agents, random, and deterministic. The proposed design is characterized by its emphasis on energy efficiency, as it reduces energy consumption by as much as 87% compared to other approaches.

In the Industrial IoT (IIoT) network, MEC and Non-Orthogonal Multiple Access (NOMA) are promising technologies. Multiple MEC servers must cooperate to increase their processing capacity. The rapidly changing IIoT environment with undefined scenarios, such as dynamic wireless channels, complex task demands, changing wireless loads, and multiple MEC servers, can critically affect the task offloading decision and NOMA user pairing. This greatly affects resource management in NOMA-MEC-based IIoT networks. In order to deal with this issue, Lin, Zhou [93] developed a distributed DRL-based solution for optimizing task offloading decisions and sub-channel assignments to provide binary computing offloading. Recurrent Neural Networks (RNN) are employed for each IIoT device agent to address the partial state observability problem by predicting the load states of sub-channels and MEC servers, which are further used to make RL decisions. According to simulation results, the proposed prediction-based-DRL method (P-DRL) achieves higher task satisfaction rates than existing methods.

4 Discussion and analytical comparison

This section aims to provide an analytical examination of the reviewed task scheduling methods. The obtained results are reported based on mentioned research questions in Sect. 2. In the previous section, the existing task scheduling approaches in the IoT were ordered into three groups, heuristic-based methods, non-heuristic-based methods, and machine learning-based methods, and assessed considering important qualitative metrics, such as resource utilization, communication cost, reliability, makespan, load balancing, waiting time, response time, throughput, and energy consumption. When solving the task scheduling problem, non-heuristic algorithms generally rely on gradient-based search methods to find optimal local solutions within a reasonable timeframe. Gradient information, however, is necessary for finding the search directions. Since these approaches cannot differentiate between objective functions and constraints, they may be inefficient when solving the task scheduling problem.

Consequently, heuristic or meta-heuristic algorithms have become increasingly popular for solving task scheduling problems. These algorithms derive their mathematical basis from natural phenomena. Despite the effectiveness of these algorithms in exploring the search space, they take a relatively long time to identify the local optimum. Machine learning algorithms represent a new method for resolving the task scheduling problem in the IoT environment, which are employed to predict the resources based on the task size and features. In large-scale networks, the search efficiency of these algorithms is low, and they have slow convergence speeds. The methods mentioned above were concisely described, considering their main advantages and drawbacks. For instance, the suggested task scheduling technique by Hasan and Al-Rizzo [72] provides an effective manner in terms of minimum makespan and improves resource utilization, but it does not address the prioritization of tasks. Besides, the method proposed by Pandit, Mir [21] aims to reduce communication costs and improve resource utilization, but it does not evaluate energy consumption.

4.1 Comparison based on scheduling constraints

Memory, power, computational capability, and communication resource constraints are significant factors in the IoT field and are key performance indicators in task scheduling. These factors might negatively influence the scheduling performance if a large number of applications are unable to meet these constraints. Researchers have examined one or more indicators regarding the IoT task scheduling problem in the reviewed papers. As shown in Fig. 6, computational capability is the constraint most emphasized by task scheduling techniques, especially heuristic techniques.

4.2 Comparison based on task-resource mapping approaches

Generally, the IoT resources are mapped to the incoming tasks in four basic methods, namely, prediction-based, AI-based, dynamic, and static, in order to efficiently use the available resources depending on the submitted workload and the condition of the IoT environment. Allocating resources based on prediction is associated with the

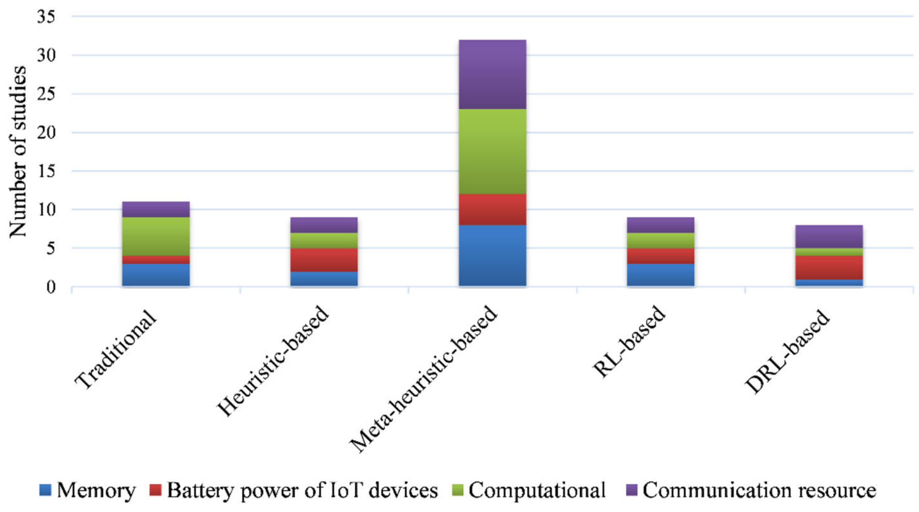

Fig. 6 Usage ratio of the constrained approaches

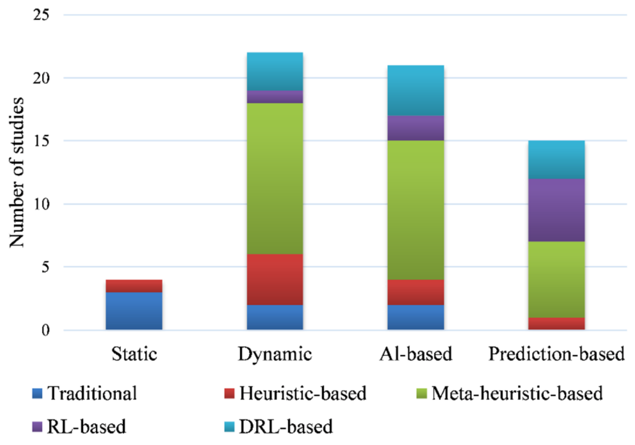

Fig. 7 Usage ratio of the task-resource mapping schemes

behavior of methods and various measures. To effectively schedule tasks and optimize resource allocation in the IoT environment, it is significant to utilize techniques for automatic resource allocation or reservation that predict user requirements for the future. AI-based scheduling involves the creation of a highly technical and specialized process that can schedule and assign resources to various aspects, including expert systems, ML, neural networks, agent-based systems, nature-inspired intelligent systems, and autonomous systems. Task scheduling can be conducted dynamically without requiring knowledge of all task properties. This can be beneficial for IoT users to accommodate variable demands, particularly when optimizing resource utilization is more important than lowering execution time. Static scheduling relies on prior information about the tasks to make a scheduling decision. As depicted in Fig. 7, AI-based scheduling and dynamic scheduling are more commonly used strategies due to their effectiveness and efficiency.

4.3 Comparison based on the significance of scheduling metrics

Table 8 provides a side-by-side comparison of methods based on considered metrics. This table specifies effective methods, considering the mentioned metrics. As specified in Fig. 8, most of the researchers have attempted to provide a method with minimum delay and energy consumption as well as maximum resource utilization.

4.4 Comparison based on the nature of the scheduling problem

Generally, optimization algorithms can be ordered into two groups, including single-objective and multi-objective optimization algorithms. Multi-objective algorithms require multiple objectives to optimize for an optimal solution, whereas single-objective algorithms can optimize only one objective function. The multi-objective task scheduling methods simultaneously optimize multiple objectives, while single-objective task scheduling methods determine only one fitness value. We checked the discussed task scheduling methods based on their adopted fitness function type (multi-objective and single-objective). As shown in Fig. 9, 87% of papers belong to multi-objective methods, and the remaining 13% belong to single-objective methods.

4.5 Comparison based on adopted simulation tools and case studies

Researchers have implemented their innovation in well-known environments such as Matlab, C, VB, iFogSim, Java, and Python. Besides, as indicated in Fig. 10, various case studies have been adopted in the discussed task scheduling methods. It is obvious that the fog-enabled

Table 8 A side-by-side comparison of task scheduling methods

Category	Technique	References	Task scheduling qualitative metrics								
			Response time	Makespan	Resource utilization	Waiting time	Energy consumption	Communication cost	Reliability	Load balancing	Throughput
Traditional	NOMA	Ning, Dong [52]				●					●
	MINLP	Ning, Dong [54]			●		●				●
	Pareto-dominated	Cai, Geng [55]		●	●		●			●	●
	Integer programming	Huang, Li [56]	●		●						●
	MINLP	Zhou, Sun [57]					●			●	
	Asymmetric task adaptation rate scheduler	Yang, Thangarajan [58]					●	●			
Heuristic-based	Dynamic programming	Caruso, Chessa [51]		●			●				●
	HEFT	Ma, Gao [59]		●							●
	NOMA	Wang, Zhou [60]			●						●
	GPA	Najafizadeh, Salajegheh [61]	●								●
	MILP	Delgado and Famaey [62]				●	●				
Meta-heuristic-based	GA	Li, Wang [63]	●					●			
	GA	Wu [64]						●		●	
	GA	Liu, Wei [65]		●		●		●		●	
	GA-ACO	Basu, Karuppiah [66]		●	●					●	●
	DPT-ACO	Fan, Liu [67]			●	●	●				●
	MMAS	Boveiri, Khayami [68]		●	●			●			
	PSO	Al-Turjman, Hasan [69]			●	●			●		●
	ACO	Li, Jia [70]	●					●	●		
	GA	Ma, Gao [59]	●	●		●					
	ACO	Prasanth, George [71]				●					●
	CPSO	Hasan and Al-Rizzo [72]		●	●	●					●
	PSO	Javanmardi, Shojafar [73]			●	●			●		
	MPA	Abdel-Basset, Mohamed [74]	●	●			●	●			

Table 8 (continued)

Category	Technique	References	Task scheduling qualitative metrics								
			Response time	Makespan	Resource utilization	Waiting time	Energy consumption	Communication cost	Reliability	Load balancing	Throughput
	PSG	Azizi, Shojafar [75]	●	●		●	●				
	SSA- MRFO	Attiya, Abd Elaziz [76]		●				●			●
	GA-PSO	Bu [77]			●	●		●		●	●
	AO	Kandan, Krishnamurthy [78]		●	●	●					●
RL-based	Q-learning	Xie, Wang [81]			●	●	●				
	Q-learning	Ge, Liu [82]				●	●				●
	Two-level neural network	Pandit, Mir [21]		●	●	●		●			
	Imitation learning	Wang, Ning [83]			●	●	●				
DRL-based	DNN	Gao, Wu [88]			●	●		●			●
	MDP	Zhou, Wu [89]				●	●				
	SOM	Shadroo, Rahmani [90]			●			●			
	D3RQN	Tang, Xie [91]				●	●				
	Deep Q-Learning	Sellami, Hakiri [92]	●		●	●	●			●	
	RNN	Lin, Zhou [93]				●			●		●

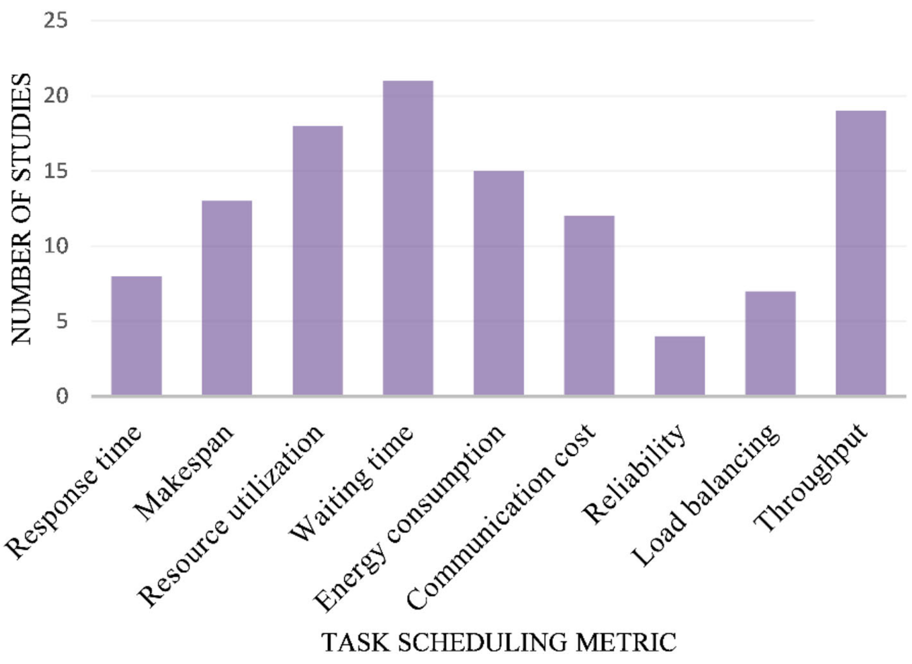

Fig. 8 Percentage of considered performance metrics in the task scheduling methods

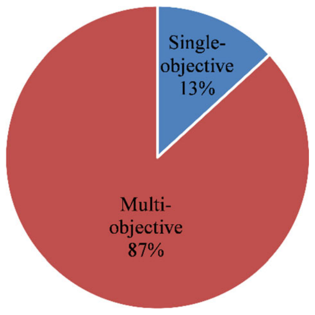

Fig. 9 Type of fitness function in discussed methods

application with eight usages is the most applied case study among existing case studies.

5 Open issues

The current study organized a detailed and systematic paper in this field by discussing the state-of-the-art methods in the field of task scheduling in the IoT and highlighting their main features. The obtained results confirm that the recent approaches have not covered all available challenges and scheduling metrics. In order to answer question 1 mentioned in Sect. 2, this section aims to specify open issues and some considerable challenges to help future researchers develop effective works. As a matter of fact, this section provides a roadmap toward efficient task scheduling methods.

It is estimated that the number of IoT devices will exceed one trillion by 2030 [94]. These IoT devices produce a significant amount of data that can be utilized for a variety of purposes. Cloud data centers by themselves cannot handle the vast amount of data efficiently. Also, cloud data centers are multi-hop away from the end-user. Consequently, data transmission to a remote cloud is accompanied by substantial delays and congestion on networks, which are incompatible with time-sensitive applications such as telemedicine, vehicle-to-vehicle communications, e-health, and so on. Thus, conventional centralized IoT infrastructure may be unable to cope with the challenges associated with security, network congestion, long delays, and bandwidth consumption. To address these problems, various technologies have been proposed, such as mist computing, osmotic computing, mobile cloud computing, multi-access edge computing, volunteer computing, and fog computing. Among these technologies, fog computing, developed by Cisco, is of particular interest as it offers enhanced capabilities for geographical distribution, heterogeneity, low-energy consumption, mobility, and application processing, thereby achieving significant improvements in QoS requirements.

- *Fog computing:* Fog computing technology is a form of decentralized computing that facilitates the extension of cloud computing infrastructure to the edge of a network. It has recently received considerable interest in handling IoT applications. Fog computing provides computing resources within close proximity to IoT devices, which simplifies the process involved in accessing computing resources and enables IoT requests to be processed more quickly. Although fog computing offers significant advantages for processing IoT applications, it also presents several challenges

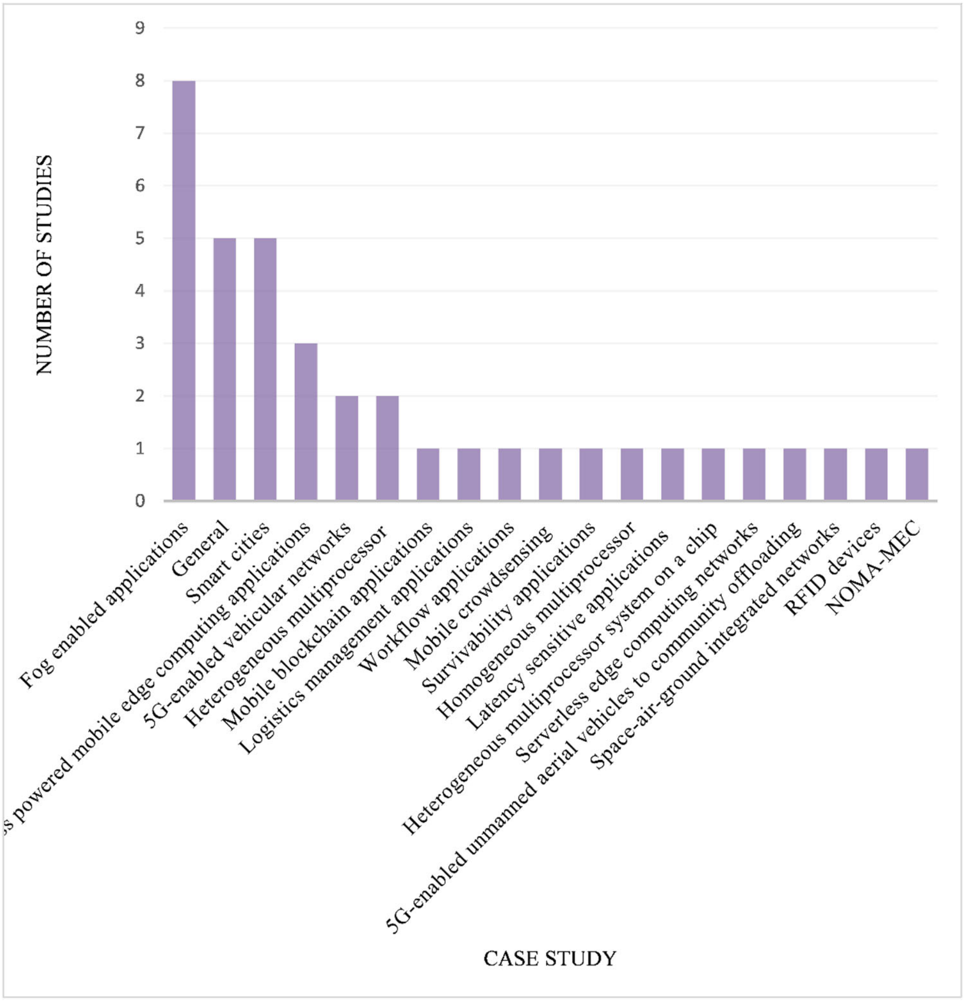

Fig. 10 Percentage of the applied case studies in the task scheduling methods

regarding the provisioning and management of fog resources. Fog computing resources comprise distributed, heterogeneous, dynamic, and capacity-limited components. Furthermore, energy consumption is one of the most important factors to consider when utilizing fog computing. Thus, fog resource provisioning and management play a crucial role in leveraging fog computing for the efficient execution of IoT applications. Besides, how to allocate heterogeneous and dynamic fog resources to IoT tasks while minimizing the energy consumption of fog nodes remains a fundamental issue.

- *Edge computing and mobile edge computing:* These technologies have a similar principle to fog computing, which is aimed at bringing computation closer to the user. Fog and edge computing often refer to the same thing. However, several researchers have defined edge computing as a paradigm restricted to the edge network, which is composed of devices like mobile phones and access points that are directly connected to IoT devices. In contrast, fog expands this idea by incorporating core networks as well as cloud data centres in order to provide services close to data sources. Mobile edge computing emphasizes mobile end-users and brings storage and computational functions to the edge of the mobile network through enhancing the capabilities of the base stations of the 5G and 6G networks. Nevertheless, all these paradigms are characterized by similar features, including distributed, constrained, and heterogeneous resources, in addition to a number of key objectives, including mobility support, improved network efficiency, and lowered latency.
- *Mist computing:* As a layer between the fog and IoT layers, mist computing attempts to minimize latency by bringing fog computing even closer to IoT sensors and devices. This paradigm incorporates several aspects of cloud computing albeit on a smaller scale, namely resource pooling and virtualization. Mist resources have limited capacity compared to fog resources, which are sparse when compared to cloud resources. Mist computing has sometimes been mistaken for edge computing. However, they differ in several important respects.

In particular, edge computing typically occurs at the IoT layer, by utilizing local computing capabilities embedded in the IoT sensors and devices themselves (therefore, edge computing is device-specific). Hence, IoT sensors and devices are known as edge computing devices. On the other hand, fog/mist computing offers end-device independence and enables hierarchical and scalable architectures between IoT and cloud environments. Moreover, edge computing rarely supports virtualization and resource pooling capabilities, compared to fog/mist computing.

- *Osmotic computing:* Osmotic computing is a recently emerging paradigm that focuses on the scheduling of microservice-based applications across cloud and fog environments. Osmotic computing provides a balanced deployment of microservices by integrating fog/edge and cloud resources in order to meet the varying expectations of microservices-based IoT applications. Osmotic computing defines a microservice architecture as an appropriate model to develop IoT applications and deployment them in an integrated fog/edge and cloud environment due to their fine granularity, fast deployment and elasticity. Therefore, this paradigm attempts to address the specific challenges associated with microservices, including dynamic edge-cloud placement, elasticity control, monitoring, networking, and microservice orchestration.
- *Mobile cloud computing:* Mobile cloud computing enhances the ability of mobile devices to process and schedule tasks. Mobile cloud computing enables individuals to execute resource-intensive tasks in the cloud near mobile devices. Although task scheduling is an integral part of mobile devices, due to the size and power consumption of modern applications, it poses a significant challenge for mobile applications. As a result of technological advances, mobile devices are now capable of collecting, handling, and transmitting data without interruption to the concerned intelligent devices and their effective environment. However, intelligent mobile devices still face limitations concerning bandwidth utilization, battery capacity, CPU speed, and power consumption and intensive operation.
- *Volunteer computing:* Volunteer computing is a form of network-based distributed computing in which individuals can donate idle computing resources on their personal computers for the purpose of scientific research computing. Grid and volunteer share the idea of providers across organizations contributing to a pool of resources that appears to consumers as a transparent whole. The motivation of providers to participate is a key difference between the two paradigms. In grid computing, providers anticipate profiting from the grid itself in the future, i.e., harvesting computing power as consumers. In volunteer computing, service providers contribute their time and resources to a worthy cause. In this way, the group of consumers (typically the members of a scientific project) is clearly defined from the outset. Furthermore, volunteer computing resources are provided by private end-users, whereas grid resources are often owned by an organization, for example, a university. As private individuals participate in volunteer computing as providers, research on volunteer computing has a strong focus on security, fault tolerance, usability, and incentive mechanisms.
- *Real IoT datasets:* The use of a suitable real-world dataset is a significant issue in the reviewed task scheduling methods. As discussed in previous sections, most of the methods have not been verified based on a real IoT dataset. The researchers can obtain more effective methods and gain real-time properties by implementing their innovations based on the recent datasets.
- *Mobility-aware:* As another open issue, the mobility of IoT devices has not been considered in most of the reviewed papers. The obtained results confirm that the increasing number of real-time and mobile tasks has a significant impact on the performance of the task scheduling method and significantly increases the total cost. The mobility of IoT devices faces various problems, such as tackling the dynamic IoT environment and the limited energy of devices.
- *Multiprocessing capabilities:* Since embedded systems are utilized to gather and investigate different types of IoT data, the multiprocessing capabilities of embedded systems have become more important. Embedded systems for IoT objects with various sensors aim to accomplish varied types of lightweight data processing.
- *Task migration:* Task migration refers to selecting suitable processing servers to process the tasks, separating tasks into several sub-tasks that can be scheduled on various processors while complying with the feasible order of executing the parts of the main task. In mobile edge computing, different edge servers need to adjust the allocation of task loads and edge server resources at all levels to fulfill the reliability of different tasks demand, execution energy consumption, and the heterogeneity of processing delay.
- *Replication of multiple fog broker nodes:* The proposed task scheduling approaches can be extended by replicating multiple fog broker nodes to cover fault tolerance and scalability issues and support high mobility. In this regard, more scalable methods with minimum response and execution time can be obtained by developing the approaches in a distributed manner.
- *Prioritization of tasks:* Most discussed task scheduling methods have not considered prioritizing tasks based on

context information. Developing models of prioritized applications from a recognition context can be considered another great matter for future studies. By specifying the information relevant to the contexts and device of the application, such as application QoS, network signal strength, battery level, etc., the scheduling methods can gain better performance.

- *Subdivision and offloading:* Nodes with limited resources may not be able to execute tasks. The solution is to divide a task into multiple loosely coupled subtasks and then schedule them on multiple devices. Task offloading refers to the case that the task is offloaded to be executed by a nearby helper node. This allows devices with limited resources to take advantage of the resources of nearby nodes that have more computing power and can execute tasks more efficiently. Furthermore, by dividing tasks into smaller subtasks, it reduces the amount of data that needs to be transferred, which reduces communication overhead. Offloading subtasks to multiple nodes requires effective synchronization policies. The offloading of tasks to different nodes may be caused by mobility, excessive workload, and node failures. To address these issues, Wang, Tan [95] proposed the use of DRL and Kubernetes approaches for node orchestration. The subdivision mechanism can be further improved by increasing the independence and loose coupling of each task.
- *Network state observations:* The change in critical levels for distinct information in mobile edge networks impacts users' decision-making, especially when partial observations are available. This issue has not yet been addressed in existing research. An effective way to overcome these challenges is to quantify the freshness of information according to its critical levels.
- *Self-adaptive scheduling:* The majority of scheduling models ignore the learning aspect, which is a challenge in IoT task scheduling. While some research studies have considered self-adaptive scheduling, all these efforts have been limited to the experimental realm. In this regard, task scheduling algorithms are necessary to optimally schedule tasks arising from unexpected events in dynamic environments.
- *Energy consumption:* As IoT devices suffer from energy constraints resulting from low-power batteries, energy-aware approaches remain an open issue. Researchers have focused on energy optimization, while efficient bandwidth usage in transmission of data, energy loss, and battery drainage issues remain challenges to be addressed.
- *Context-aware service provisioning:* The context constitutes a set of factors that can affect the performance of applications. The existing methods of provisioning context-aware services have limited flexibility, scalability, and cannot accommodate a wide range of IoT applications. Therefore, it is necessary to develop more techniques for context-aware service provisioning to address the mentioned shortcomings.
- *Security:* Security of IoT nodes presents a significant concern since they are resource-constrained and installed in unsafe environments, making them vulnerable to a variety of attacks. Thus, developing a reliable, high-speed, and lightweight safety mechanism remains a challenge.
- *Privacy:* IoT nodes obtain a significant amount of personal data through a variety of IoT applications. As a consequence, protecting the privacy of such information is of paramount importance to users. While some privacy-preserving strategies have been applied to IoT nodes, no satisfactory authentication mechanism has been developed. IoT nodes are more likely to suffer from vulnerabilities that complicate the authentication process.
- *Parallel scheduling:* Parallel processing involves dividing a task into several subtasks and executing them simultaneously, which reduces delays via distributed computing is another interesting direction for upcoming studies.
- *Optimal resource allocation:* Due to the mobile nature of IoT nodes, available resources may not be accessible at other times, making it challenging to allocate resources. Latency issues for real-time services, the absence of generalization, and the rapid adaptation of current techniques require further investigation.
- *Resource utilization:* IoT devices face resource constraints regarding energy, computation, and storage. They experience dynamic workloads associated with both delay-tolerant and latency-sensitive applications. Consequently, it is challenging to arrange the unpredictable arrival of tasks on these nodes for maximum efficiency.

6 Conclusions

The current paper has systematically reviewed the recent task scheduling approaches in the IoT. The available methods have been ordered into five key groups, traditional, heuristic-based, meta-heuristic-based, RL-based, and DRL-based methods, and checked out based on important evaluation metrics. In this regard, the main features, strengths, and weaknesses of each method have been described. To specify the efficient approaches, all discussed methods have been compared side-by-side. As a matter of fact, an effort has been made to provide a strong basis for comprehending diverse aspects of the task scheduling problem in the IoT by examining and analyzing recent works and offering an up-to-date comparison of them. The evaluation comparison shows that most researchers have attempted to provide a method with minimum communication cost and maximum resource utilization, but response time and energy consumption have been ignored by most of them. Moreover, the obtained results prove that none of the discussed methods simultaneously considers all task scheduling evaluation metrics. The outcome of this research can be precious and helpful for future researchers and it can act as a roadmap to assist them in improving and enhancing their approaches.

Author contributions TB, ZH, KZ, YW, HS, and JZ: Conceptualization, Formal Analysis, Investigation, Methodology, Software, Writing-Original Draft. ZR and SL: Conceptualization, Supervision, Writing-Review & Editing.

Funding The authors have not disclosed any funding.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Kumar, A., et al.: Smart power consumption management and alert system using IoT on big data. Sustain Energy Technol Assess **53**, 102555 (2022)
2. He, P., et al.: Towards green smart cities using Internet of Things and optimization algorithms: A systematic and bibliometric review. Sustain Comput: Info Syst **36**, 100822 (2022)
3. Meisami, S., Beheshti-Atashgah, M., Aref, M. R.: *Using blockchain to achieve decentralized privacy in IoT healthcare.* arXiv preprint arXiv:2109.14812 (2021)
4. Liu, X., et al.: The method of Internet of Things access and network communication based on MQTT. Comput. Commun. **153**, 169–176 (2020)
5. Mehbodniya, A., et al.: Modified Lamport Merkle digital signature blockchain framework for authentication of internet of things healthcare data. Expert. Syst. **39**(10), e12978 (2022)
6. Lin, Y., et al.: Optimal caching scheme in D2D networks with multiple robot helpers. Comput. Commun. **181**, 132–142 (2022)
7. Mohseni, M., Amirghafouri, F., Pourghebleh, B.: CEDAR: A cluster-based energy-aware data aggregation routing protocol in the internet of things using capuchin search algorithm and fuzzy logic. Peer-to- Peer Netw App 1–21 (2022).
8. Hayyolalam, V., et al.: Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques. Int J Adv Manuf Technol **105**(1–4), 471–498 (2019)
9. Pourghebleh, B., Hayyolalam, V. A comprehensive and systematic review of the load balancing mechanisms in the Internet of Things. Cluster Comput., 1–21 (2019).
10. Akhavan, J., Manoochehri, S. Sensory data fusion using machine learning methods for in-situ defect registration in additive manufacturing: A review. In 2022 IEEE international IOT, electronics and mechatronics conference (IEMTRONICS). IEEE (2022).
11. Pourghebleh, B., Hayyolalam, V., Anvigh, A.A.: Service discovery in the Internet of Things: Review of current trends and research challenges. Wireless Netw. **26**(7), 5371–5391 (2020)
12. Shadroo, S., Rahmani, A.M.: Systematic survey of big data and data mining in internet of things. Comput. Netw. **139**, 19–47 (2018)
13. Tsai, C.-W.: SEIRA: An effective algorithm for IoT resource allocation problem. Comput. Commun. **119**, 156–166 (2018)
14. Chen, Y., et al.: Channel-reserved medium access control for edge computing based IoT. J. Netw. Comput. Appl. **150**, 102500 (2020)
15. Sodhro, A.H., et al.: 5G-based transmission power control mechanism in fog computing for Internet of Things devices. Sustainability **10**(4), 1258 (2018)
16. Nikoui, T.S., et al.: Cost-aware task scheduling in fog-cloud environment. In 2020 CSI/CPSSI international symposium on real-time and embedded systems and technologies (RTEST). IEEE (2020).
17. Ataie, I., et al.: D 2 FO: Distributed dynamic offloading mechanism for time-sensitive tasks in fog-cloud IoT-based systems. In 2022 IEEE international performance, computing, and communications conference (IPCCC). IEEE (2022).
18. Seyfollahi, A., Taami, T., Ghaffari, A.: Towards developing a machine learning-metaheuristic-enhanced energy-sensitive routing framework for the internet of things. Microprocess. Microsyst. **96**, 104747 (2023)
19. Pourghebleh, B., Navimipour, N.J.: Data aggregation mechanisms in the Internet of things: A systematic review of the literature and recommendations for future research. J. Netw. Comput. Appl. **97**, 23–34 (2017)
20. Sandhu, M.M., et al.: Task scheduling for simultaneous IoT sensing and energy harvesting: A survey and critical analysis. arXiv preprint arXiv:2004.05728 (2020)
21. Pandit, M.K., Mir, R.N., Chishti, M.A.: Adaptive task scheduling in IoT using reinforcement learning. Int. J Intel Comput Cybern (2020).
22. Zhang, Y., Fu, J.: Energy-efficient computation offloading strategy with tasks scheduling in edge computing. Wireless Netw. **27**(1), 609–620 (2021)
23. Taami, T., Azizi, S., Yarinezhad, R. An efficient route selection mechanism based on network topology in battery-powered internet of things networks. Peer-to-Peer Netw App, 1–16 (2022).
24. Stavrinides, G.L., Karatza, H.D.: A hybrid approach to scheduling real-time IoT workflows in fog and cloud environments. Multimedia Tools App **78**(17), 24639–24655 (2019)
25. Hosseinioun, P., et al.: A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm. J Parallel Distributed Comput **143**, 88–96 (2020)
26. Zhou, J.: Real-time task scheduling and network device security for complex embedded systems based on deep learning networks. Microprocess. Microsyst. **79**, 103282 (2020)
27. Amalarethinam, D.G. Josphin, A. M.: Dynamic task scheduling methods in heterogeneous systems: A survey. Int J Comput App **110**(6) (2015).
28. Soualhia, M., Khomh, F., Tahar, S.: Task scheduling in big data platforms: a systematic literature review. J. Syst. Softw. **134**, 170–189 (2017)
29. Hazra, D., et al.: Energy aware task scheduling algorithms in cloud environment: A survey. In: Smart Computing and Informatics, pp. 631–639. Springer (2018)

30. Ramezani, F., et al.: Task scheduling in cloud environments: A survey of population-based evolutionary algorithms. Evol. Comput. Scheduling, 213–255 (2020).
31. AminiMotlagh, A., Movaghar, A., Rahmani, A.M.: Task scheduling mechanisms in cloud computing: A systematic review. Int. J. Commun. Syst. **33**(6), e4302 (2020)
32. Alizadeh, M.R., et al.: Task scheduling approaches in fog computing: A systematic review. Int. J. Commun Syst **33**(16), e4583 (2020)
33. Hosseinzadeh, M., et al.: Multi-objective task and workflow scheduling approaches in cloud computing: A comprehensive review. J. Grid Comput., 1–30 (2020:).
34. Yang, X., Rahmani, N.: Task scheduling mechanisms in fog computing: Review, trends, and perspectives. Kybernetes (2020)
35. Matrouk, K., Alatoun, K.: Scheduling algorithms in fog computing: A survey. Int J Netw Distributed Comput **9**(1), 59–74 (2021)
36. Kaur, N., Kumar, A., Kumar, R. A systematic review on task scheduling in Fog computing: Taxonomy, tools, challenges, and future directions. *Concurrency Comput.*, e6432.
37. Kaur, N., Kumar, A., Kumar, R.: A systematic review on task scheduling in Fog computing: Taxonomy, tools, challenges, and future directions. Concurrency Comput **33**(21), e6432 (2021)
38. Kitchenham, B., et al.: Systematic literature reviews in software engineering–a systematic literature review. Inf. Softw. Technol. **51**(1), 7–15 (2009)
39. Hayyolalam, V., Pourghebleh, B., PourhajiKazem, A.A.: Trust management of services (TMoS): Investigating the current mechanisms. Trans Emerg Telecommun Technol **31**(10), e4063 (2020)
40. Pourghebleh, B., et al.: A roadmap towards energy-efficient data fusion methods in the Internet of Things. Concurrency Comput (2022). https://doi.org/10.1002/CPE.6959
41. Hayyolalam, V., et al.: Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends. Concurrency Comput **34**(5), e6698 (2022)
42. Kamalov, F., et al.: Internet of medical things privacy and security: Challenges, solutions, and future trends from a new perspective. Sustainability **15**(4), 3317 (2023)
43. Praveenchandar, J., Tamilarasi, A.: Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing. J. Ambient. Intell. Humaniz. Comput. **12**(3), 4147–4159 (2021)
44. Mosleh, M.A., et al.: Adaptive cost-based task scheduling in cloud environment. Sci Program **2016**, 1–9 (2016)
45. Siddiqi, M.A., Yu, H., Joung, J.: 5G ultra-reliable low-latency communication implementation challenges and operational issues with IoT devices. Electronics **8**(9), 981 (2019)
46. Abdelmoneem, R.M., Benslimane, A., Shaaban, E.: Mobility-aware task scheduling in cloud-fog IoT-based healthcare architectures. Comput. Netw. **179**, 107348 (2020)
47. Kanbar, A.B., Faraj, K.H.A.: Region aware dynamic task scheduling and resource virtualization for load balancing in IoT-fog multi-cloud environment. Future Gen Comput Syst **137**, 70–86 (2022)
48. Aladwani, T.: Scheduling IoT healthcare tasks in fog computing based on their importance. Procedia Comput Sci **163**, 560–569 (2019)
49. Wadhwa, H., Aron, R.: Optimized task scheduling and preemption for distributed resource management in fog-assisted IoT environment. J Supercomput **79**, 1–39 (2022)
50. Abd Elaziz, M., Abualigah, L., Attiya, I.: Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments. Future Gen Comput Syst **124**, 142–154 (2021)
51. Caruso, A., et al.: A dynamic programming algorithm for high-level task scheduling in energy harvesting IoT. IEEE Internet Things J. **5**(3), 2234–2248 (2018)
52. Ning, Z., et al.: Partial computation offloading and adaptive task scheduling for 5G-enabled vehicular networks. IEEE Trans Mobile Comput (2020). https://doi.org/10.1109/TMC.2020.3025116
53. Cao, K., et al.: Enhancing physical layer security for IoT with non-orthogonal multiple access assisted semi-grant-free transmission. IEEE Internet Things J **9**, 24669–24681 (2022)
54. Ning, Z., et al.: 5G-enabled UAV-to-community offloading: joint trajectory design and task scheduling. IEEE J. Sel. Areas Commun. **39**(11), 3306–3320 (2021)
55. Cai, X., et al.: A multicloud-model-based many-objective intelligent algorithm for efficient task scheduling in internet of things. IEEE Internet Things J. **8**(12), 9645–9653 (2020)
56. Huang, J., Li, S., Chen, Y.: Revenue-optimal task scheduling and resource management for IoT batch jobs in mobile edge computing. Peer-to-Peer Netw App **13**(5), 1776–1787 (2020)
57. Zhou, J., et al.: Security-critical energy-aware task scheduling for heterogeneous real-time MPSoCs in IoT. IEEE Trans. Serv. Comput. **13**(4), 745–758 (2019)
58. Yang, F., et al.: AsTAR: Sustainable energy harvesting for the Internet of Things through adaptive task scheduling. ACM Trans Sens Netw **18**(1), 1–34 (2021)
59. Ma, X., et al.: An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing. EURASIP J. Wirel. Commun. Netw. **2019**(1), 1–19 (2019)
60. Wang, K., et al.: Online task scheduling and resource allocation for intelligent NOMA-based industrial Internet of Things. IEEE J. Sel. Areas Commun. **38**(5), 803–815 (2020)
61. Najafizadeh, A., et al.: Privacy-preserving for the internet of things in multi-objective task scheduling in cloud-fog computing using goal programming approach. Peer-to-Peer Netw App **14**, 3865–3890 (2021)
62. Delgado, C., Famaey, J.: Optimal energy-aware task scheduling for batteryless IoT devices. IEEE Trans. Emerg. Top. Comput. **10**(3), 1374–1387 (2021)
63. Li, J., Wang, Y., Sun, T.: A hybrid genetic algorithm for task scheduling in internet of things. In ICIT 2013 the 6th international conference on information technology. Amman, Jordan (2013).
64. Wu, D.H.: Task optimization scheduling algorithm in embedded system based on internet of things. Appl Mech Mater. (2014). https://doi.org/10.4028/www.scientific.net/AMM.513-517.2398
65. Liu, Q., et al.: Task scheduling in fog enabled Internet of Things for smart cities. In 2017 IEEE 17th international conference on communication technology (ICCT). IEEE (2017).
66. Basu, S., et al.: An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment. Futur. Gener. Comput. Syst. **88**, 254–261 (2018)
67. Fan, J., et al.: LPDC: Mobility-and deadline-aware task scheduling in tiered IoT. In 2018 IEEE 4th international conference on computer and communications (ICCC). IEEE (2018).
68. Boveiri, H.R., et al.: An efficient Swarm-Intelligence approach for task scheduling in cloud-based internet of things applications. J. Ambient. Intell. Humaniz. Comput. **10**(9), 3469–3479 (2019)
69. Al-Turjman, F., Hasan, M.Z., Al-Rizzo, H.: Task scheduling in cloud-based survivability applications using swarm optimization in IoT. Trans Emerg Telecommun Technol **30**(8), e3539 (2019)
70. Li, W., et al.: A multi-task scheduling mechanism based on ACO for maximizing workers' benefits in mobile crowdsensing service markets with the Internet of Things. IEEE Access **7**, 41463–41469 (2019)

71. Prasanth, A., George, J.A., Surendram, P.: Optimal resource and task scheduling for IoT. In 2019 international conference on innovation and intelligence for informatics, computing, and technologies (3ICT). IEEE (2019).
72. Hasan, M.Z., Al-Rizzo, H.: Task scheduling in Internet of Things cloud environment using a robust particle swarm optimization. Concurrency Comput **32**(2), e5442 (2020)
73. Javanmardi, S., et al.: FPFTS: A joint fuzzy particle swarm optimization mobility-aware approach to fog task scheduling algorithm for Internet of Things devices. Software: Practice and Experience (2020).
74. Abdel-Basset, M., et al.: Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications. IEEE Trans Indust Info (2020). https://doi.org/10.1109/TII.2020.3001067
75. Azizi, S., et al.: Deadline-aware and energy-efficient IoT task scheduling in fog computing systems: A semi-greedy approach. J Netw Comput App (2022). https://doi.org/10.1016/j.jnca.2022.103333
76. Attiya, I., et al.: An improved hybrid swarm intelligence for scheduling IoT application tasks in the cloud. IEEE Trans Indust Info (2022). https://doi.org/10.1109/TII.2022.3148288
77. Bu, B.: Multi-task equilibrium scheduling of Internet of Things: A rough set genetic algorithm. Comput. Commun. **184**, 42–55 (2022)
78. Kandan, M., et al.: Quasi oppositional Aquila optimizer-based task scheduling approach in an IoT enabled cloud environment. J Supercomput (2022). https://doi.org/10.1007/s11227-022-04311-y
79. Xu, J., et al.: Human-factors-in-driving-loop: driver identification and verification via a deep learning approach using psychological behavioral data. IEEE Trans Intel Transp Syst (2022). https://doi.org/10.1109/tits.2022.3225782
80. Lei, W., et al.: Optimal remanufacturing service resource allocation for generalized growth of retired mechanical products: Maximizing matching efficiency. IEEE Access **9**, 89655–89674 (2021)
81. Xie, J., Wang, S., Yin, C.: Machine learning based task scheduling for wireless powered mobile edge computing IoT networks. In 2019 11th international conference on wireless communications and signal processing (WCSP). IEEE (2019).
82. Ge, J., et al.: Q-learning based flexible task scheduling in a global view for the Internet of Things. Trans Emerg Telecommun Technol **32**, e4111 (2020)
83. Wang, X., et al.: Imitation learning enabled task scheduling for online vehicular edge computing. IEEE Trans Mobile Comput (2020). https://doi.org/10.1109/TMC.2020.3012509
84. Saeidi, S.A., et al.: A novel neuromorphic processors realization of spiking deep reinforcement learning for portfolio management. In 2022 design, automation & test in Europe conference & exhibition (DATE). IEEE (2022).
85. Haghshenas, S.H., Hasnat, M.A., Naeini, M. A temporal graph neural network for cyber attack detection and localization in smart grids. arXiv preprint arXiv:2212.03390 (2022).
86. Qin, X., et al.: User OCEAN personality model construction method using a BP neural network. Electronics **11**(19), 3022 (2022)
87. Zhang, X., et al.: A hybrid-convolution spatial-temporal recurrent network for traffic flow prediction. Comput J (2022). https://doi.org/10.1093/comjnl/bxac171
88. Gao, Y., et al.: Deep reinforcement learning based task scheduling in mobile blockchain for IoT applications. In ICC 2020–2020 IEEE international conference on communications (ICC). IEEE (2020).
89. Zhou, C., et al.: Deep reinforcement learning for delay-oriented IoT task scheduling in space-air-ground integrated network. IEEE Trans Wireless Commun (2020). https://doi.org/10.1109/TWC.2020.3029143
90. Shadroo, S., Rahmani, A.M., Rezaee, A.: The two-phase scheduling based on deep learning in the Internet of Things. Comput. Netw. **185**, 107684 (2021)
91. Tang, Q., et al.: Distributed task scheduling in serverless edge computing networks for the Internet of Things: A learning approach. IEEE Internet Things J (2022). https://doi.org/10.1109/JIOT.2022.3167417
92. Sellami, B., et al.: Energy-aware task scheduling and offloading using deep reinforcement learning in SDN-enabled IoT network. Comput. Netw. **210**, 108957 (2022)
93. Lin, L., et al.: Deep reinforcement learning-based task scheduling and resource allocation for NOMA-MEC in Industrial Internet of Things. Peer-to-Peer Netw App **16**, 1–19 (2022)
94. Quadar, N., et al.: Cybersecurity Issues of IoT in ambient intelligence (Am I) environment. IEEE Internet Things Magaz **5**(3), 140–145 (2022)
95. Wang, K., et al.: Learning-based task offloading for delay-sensitive applications in dynamic fog networks. IEEE Trans. Veh. Technol. **68**(11), 11399–11403 (2019)

Tianqi Bu is a current student in the New Energy Science and Engineering program at the School of Mechanical and Power Engineering, Nanjing Tech University. His main research interests are wind turbines, renewable energy, and the application of Internet of Things technology to the sustainable and intelligent development of cities.

Zanyu Huang is pursuing her BS degree at Beijing University of Civil Engineering and Architecture, Beijing, China.

Kairui Zhang is pursuing his BSc degree at Department of Electrical and Computer Engineering , Queen’s University, Kingston, Canada.

Jietong Zhou is pursuing her BS degree at Northwest Normal University, Gansu, China.

Yang Wang is an undergraduate student majoring in public administration at Huazhong Agricultural University. His research interests include Internet and big data, public policy, and emergency management.

Zhangjun Ren received B.E degree in Nanjing University of Science and Technology in 2023.His rescarch interests include distributed system and Internet of Things.

Haobin Song is currently studying at Hainan Vocational University of Science and Technology. His major is construction engineering.

Sen Liu is an undergraduate majoring in optoelectronic information science and engineering at Chongqing University in China. His research interests include computer vision, image processing, natural language processing and the Internet of Things.