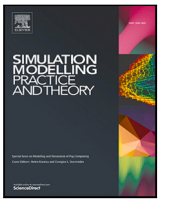
Simulation Modelling Practice and Theory 133 (2024) 102913

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat



Why it does not work? Metaheuristic task allocation approaches in Fog-enabled Internet of Drones

Saeed Javanmardi [a], Georgia Sakellari [b], Mohammad Shojafar [c], Antonio Caruso [a,*]

[a] Department of Mathematics and Physics 'Ennio De Giorgi', University of Salento, Lecce, Italy
[b] Faculty of Engineering and Science, University of Greenwich, London, United Kingdom
[c] 5G/6GIC, Information and Communication Systems, University of Surrey, Guildford, United Kingdom

ARTICLE INFO

Keywords:
IoT-Fog networks
Fog-enabled Internet of Drones
Task allocation
Metaheuristic algorithms
Execution time

ABSTRACT

Several scenarios that use the Internet of Drones (IoD) networks require a Fog paradigm, where the Fog devices, provide time-sensitive functionality such as task allocation, scheduling, and resource optimization. The problem of efficient task allocation/scheduling is critical for optimizing Fog-enabled Internet of Drones performance. In recent years, many articles have employed meta-heuristic approaches for task scheduling/allocation in Fog-enabled IoT-based scenarios, focusing on network usage and delay, but neglecting execution time. While promising in the academic area, metaheuristic have many limitations in real-time environments due to their high execution time, resource-intensive nature, increased time complexity, and inherent uncertainty in achieving optimal solutions, as supported by empirical studies, case studies, and benchmarking data. We propose a task allocation method named F-DTA that is used as the fitness function of two metaheuristic approaches: Particle Swarm Optimization (PSO) and The Krill Herd Algorithm (KHA). We compare our proposed method by simulation using the iFogSim2 simulator, keeping all the settings the same for a fair evaluation and only focus on the execution time. The results confirm its superior performance in execution time, compared to the metaheuristics.

1. Introduction

The proliferation of connected *things* (e.g., vehicles, drones, and wireless sensors) linked in a collaborative network has resulted in the evolution of Internet of Things (IoT). In such environments, intelligent *things* integrate with their surroundings, sharing data across media to interact with each other [1]. As a result, scientific knowledge managers, administrations, and investigators are working hard to invent approaches allowing widespread IoT deployment to sustain various case studies in multiple scenarios [2]. Drones have become widely used in various applications, including surveillance, object tracking, disaster investigation, and environmental monitoring. The Internet of Drones (IoD) refers to the integration of drones into IoT networks in which drones act as IoT devices [3].

IoD-Fog networks, in which we deploy Fog devices close to drones, make IoD application implementation more suitable for time-sensitive tasks. Fog devices are connected to Fog gateways to store servers at the network's edge and execute time-sensitive drones' requests to reduce execution time. Drones are part of IoD-Fog networks in an operation involving numerous computationally challenging duties. For example, a drone gathers data, renders tasks and allocates their processing to Fog devices for execution at corresponding Fog regions. The processed outcomes are then returned to the drones and delivered to the intended end users [4].

* Corresponding author.
E-mail addresses: saeed.javanmardi@unisalento.it (S. Javanmardi), g.sakellari@greenwich.ac.uk (G. Sakellari), m.shojafar@surrey.ac.uk (M. Shojafar), antonio.caruso@unisalento.it (A. Caruso).

https://doi.org/10.1016/j.simpat.2024.102913

Available online 20 February 2024

Efficient task allocation is an inseparable part of the IoD-Fog network. While metaheuristic algorithms have played a key role in computational problem-solving, their productivity in IoD-Fog time-sensitive applications is questionable due to high execution time [5,6]. These algorithms may not be practical in real environments because they are computationally expensive and require many resources. It can lead to poor performance in terms of execution or processing time, which is a critical metric in real-time IoD-Fog applications. As the number of tasks and resources increases, the optimization problem becomes more complex, and the algorithms may not find optimal solutions within a reasonable time.

1.1. Motivation

The primary motivation of this research article is to discuss challenges and considerations that limit the practicality of metaheuristic algorithms for Fog-enabled IoT applications like IoD-Fog task allocation. In the context of IoD, a proper task allocation algorithm should minimize execution time. While existing research has explored the use of metaheuristic algorithms, there is a need to reevaluate their practicality in execution time. Using such algorithms is practical in some environments with sufficient computational resources to run the algorithms. In IoT-Fog networks, we may implement these algorithms in Fog gateways with good computational capacities; however, metaheuristic algorithms do not guarantee an optimal answer, and their results may be semi-optimal. In other words, while metaheuristic algorithms have been used for IoT-Fog task allocation in some research studies, they may not be suitable for real-world applications owing to the time complexity (High execution time), uncertainty of real environments, and computational cost.

Metaheuristic algorithms are not inherently designed for time-sensitive applications with strict time constraints. They may require a considerable amount of time to converge to solutions, which is not suitable for applications that demand immediate responses [6–8]. Accordingly, they often have high execution times, which can be challenging in time-sensitive applications where quick decision-making is crucial. Here, we introduce the F-DTA method and compare it with metaheuristic algorithms such as PSO and KHA. We aim to illustrate why metaheuristic approaches are unsuitable for time-sensitive IoD-Fog applications; instead, a light method with low execution time is more practical.

1.2. Contribution

We devise a fuzzy-based method (F-DTA) that simultaneously uses the characteristics of resources and the drone's tasks. Then, we put it in the fitness function of two well-known metaheuristic approaches (PSO and KHA) to illustrate that they are impractical in the time-sensitive real environments for Fog-enabled IoD applications regarding execution time. These contributions can be summarized as follows:

- proposing a lightweight function and employing it in the two metaheuristic approaches' fitness functions, PSO and KHA;
- Comparing PSO and KHA algorithms in respect to their execution times and illustrating why these approaches are impractical for Fog-enabled IoD networks.
- Discussing some challenges in devising, evaluating and implementing metaheuristic approaches in IoD-Fog networks.

This study illustrates the practicality of lightweight methods such as F-DTA in addressing the multifaceted challenges of task allocation in Fog-enabled IoD networks. As a result, we hope that this paper serves as a valuable resource for researchers seeking to advance the practical approaches in IoT-Fog networks. The rest of the article is organized as follows: Section 2 studied related literature on Fog-enabled IoT and IoD task allocation employing metaheuristic methods. Section 3 presents an overview of the reference architecture, problem statement, and the F-DTA, PSO, and KHA task allocation approaches. Section 4 examines how the metaheuristic approaches evaluated. Section 5 discusses the challenges related to our work that may pave the way for future research directions. Finally, Section 6 concludes our study.

2. Related works

This section examines the state-of-the-art articles about task allocation in IoT and IoD. As drones act as IoT devices, we explain relevant articles about task allocation in IoT, IoD and UAV.

In our former works, we employed metaheuristic approaches, PSO, MOPSO, and NSGA-III algorithms, for FPFTS [9], FUPE [10], and S-FoS [11] projects for resource management in IoT-Fog networks. In FPFTS, we employed a single objective optimization for resource efficiency, while we used a multi-objective optimization to balance security [12] and resource efficiency in FUPE. Finally, we used NSGA-III for performance optimization in a secure resource management system. Zhang et al. [13] developed a multi-UAV task assignment technique based on a clone selection algorithm in UAV clustering. The authors contended that metaheuristic methods, instead of thoroughly exploring every potential option, can swiftly identify nearly optimal solutions for intricate optimization issues, including multi-UAV job distribution. The problem of optimal task scheduling as also been studied in [14,15].

A PSO-based algorithm was put forth by Min Deng et al. [16] to address the task assignment issue in dynamic scenarios, including UAVs and digital twins. To increase the effectiveness and diversity of solutions, it integrates metaheuristic initialization, adaptive mutation, and subgroup techniques. Shufang Xu et al. [17] proposed a task allocation algorithm based on the Multi-Discrete Wolf

Pack Algorithm (MDWPA) for UAVs. As the authors used MDWPA, their approach can quickly find near-optimal solutions for multi-UAV task allocation in the simulation. Jie Zhu et al. [18] employed NSGA algorithm for the task scheduling in a UAV-enabled Mobile Edge-Computing. They also employed a simulated annealing local search in the crossover operation.

The authors in the article [19] proposed a metaheuristic strategy for IoT-Fog–Cloud networks based on Modified Harris Hawks Optimization (MHHO). This paper aims to improve resource usage in the Fog and Cloud layers while lowering makespan time (for Cloud computing), job execution costs, and power consumption. In multi-UAV-enabled mobile edge computing, Yong Wang et al.'s focus [20] was on optimizing joint deployment and work scheduling for large-scale mobile users. The authors suggest a joint deployment and job scheduling optimization greedy method to reduce mobile users' latency. Schwarzrock et al. [21] developed a Swarm intelligence solution from a generalized assignment problem algorithm to task allocation for multiple UAVs in a decentralized way. Their method scales well under conditions of high task density, making it appropriate for large-scale networks.

2.1. The comparison and the positioning of our work

In our former works, FPFTS, FUPE, and S-FoS [9–11], we used the simulators (iFogSim, Matlab, and IoTSim-Osmosis), which allowed us to explore and develop our approaches. While, these works are good regarding delay and network usage, they did not present any solutions for execution time.

The optimal solution depends on the algorithm's parameters and the problem's features; the metaheuristic approach article [13] utilized did not guarantee the discovery of the optimal solution. Evaluating the algorithm's performance in the article [16] is ultimately challenging because it needs to provide a thorough comparison with other cutting-edge algorithms regarding performance measures. Article [17] employed the MDWPA algorithm. In this work, various variables, like the problem's size, the constraints' difficulty, and the quantity of UAVs assigned to the tasks, affect how long MDWPA takes to execute. Consequently, the MDWPA's execution time may hinder its widespread use in real-world settings. Article [18] incorporated a Simulated Annealing algorithm within the NSGA for task scheduling in a UAV-enabled MEC (Multi-access Edge Computing) system. The authors evaluated their approach in Java without considering the network layers and IoT features. Accordingly, the metrics do not include the characteristics of computer networks. Moreover, While this approach may offer advantages in optimizing solutions, utilizing a Simulated Annealing algorithm can lead to extended execution times.

Article [19] presented a hybrid heuristic method and conducted comparisons with other heuristic approaches regarding execution cost and energy consumption. As stated in detail in article [20], metaheuristic techniques have the potential to outperform their greedy algorithm in solving task scheduling problems. But compared to their greedy algorithm, the metaheuristic approach has a much higher computational time complexity. This is because a metaheuristic method uses an iterative process to find the best solution. According to article [21], heuristic techniques often result in tasks not being completed, even when UAVs are equipped with sufficient resources to carry them out.

Even though the articles [20,21] mentioned the disadvantages of metaheuristic methods in task allocation in IoT, we aim to study the execution time of these approaches in IoD-Fog networks and illustrate why they cannot be practical in time-sensitive real-time IoD applications. The primary drawback of the mentioned articles is that they did not emphasize the importance of execution time, which is vital in determining the practicality of any approach, particularly in time-sensitive applications. Metaheuristic algorithms, as noted in the literature [6–8], often need extended convergence times, making them less suitable for real-time, time-sensitive tasks.

F-DTA, in contrast, places a strong emphasis on minimizing execution time to consider the productivity of the proposed approach. We have designed our approach to meet the requirements of time-sensitive applications through extensive optimization and efficiency enhancements. As a result, it demonstrates a marked advantage over metaheuristic methods in this critical aspect. By reducing execution time, we address a fundamental issue that can impact our approach's feasibility in real-world settings.

3. Task allocation approaches

This section explains the proposed function and the metaheuristic algorithms that employs it in their fitness functions. To that end, we describe the IoD-Fog architecture. Following that, we state the problem the presented approaches addressed, and finally, we explain the fuzzy function and metaheuristic approaches.

3.1. Fog-enabled IoD architecture

This section explains the reference architecture, which uses a Fog-enabled Internet of Drones to execute tasks via a broker approach. To meet drone requirements for IoD-Fog networks, we use a three-layers architecture as the typical architecture in IoT-Fog networks [9,12,22,23]. In the flying plane (i.e., device layer), a drone (IoT device in IoT) launches to finish an expedition over some device layer Fog regions. The Fog layer comprises a collection of Fog devices at the network's edge. The drones hover above the Fog regions, gather information (e.g., recording voice for audio translation service), and generate several tasks. It then sends them to the Fog gateways to assign them to the Fog devices to use their computing resources. When the drone moves to another Fog region, it repeats the same behaviour as before.

In the reference architecture, in case of overloading in a Fog device, it may offload tasks to the Fog devices inside the Fog region or the distant Cloud data centre [24]. We assume infinite computing resources are contained in data centres at the Cloud layer and complete the offloaded tasks. Fig. 1 depicts the architecture the presented scheduling approaches employ. The Fog devices in our

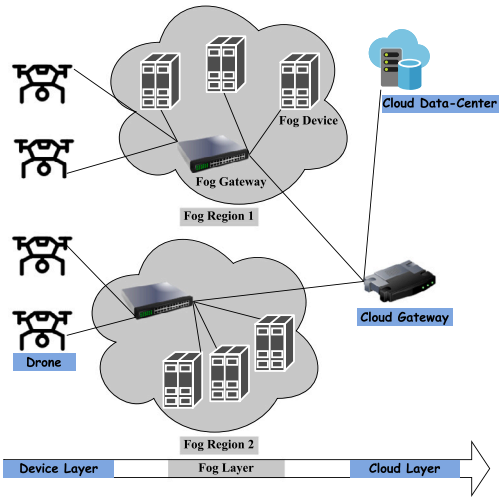


Fig. 1. The reference architecture with three layers: flying drones, fog devices, and the cloud. The fog devices are managed by a fog gateway in each zone.

scheme are grouped into Fog regions and connected by Fog gateways. Our scheme's task manager is located on the Fog gateway. For task allocation, we use a decentralized broker management strategy [25]. The task scheduler modules are in the Fog gateways, and they assign the drones' tasks to the Fog devices.

By defining this reference architecture, we create a common environment for evaluating the performance of metaheuristic techniques in the complex and dynamic environment of IoT-Fog task allocation within IoD scenarios. In subsequent sections, we will investigate the comparative analysis of these techniques within drone scenarios.

3.2. Problem statement

The task allocation problem in Fog-enabled IoD aims to assign drone tasks to Fog to enhance resource utilization. Given a set of drones and Fog devices, the goal is to allocate tasks to Fog devices to minimize execution time. While researchers have made significant progress using metaheuristic algorithms for IoD task allocation in academic areas, their productivity may fade due to high execution time. To illustrate the limitations of metaheuristic approaches, we conducted experiments using the PSO, KHA, and F-DTA methods. We used the F-DTA method in the PSO and KHA fitness functions and kept the settings the same for a fair comparison. The goal is to show that PSO and KHA have much higher execution times than the simple method (F-DTA), making them impractical in time-sensitive IoD applications. In real-world IoT scenarios, the capacity of Fog devices plays a crucial role in ensuring efficient task execution. Real-time and time-sensitive tasks necessitate careful consideration of Fog device limitations. Now, we define more formally the problem of task scheduling: we consider a set $\mathcal{T} = \{t_1, t_2, \dots, t_k\}$ of $k \in \mathbb{N}$ drones' tasks, and a set $\mathcal{F} = \{f_1, f_2, \dots, f_m\}$ of $m \in \mathbb{N}$ Fog devices; each task must be assigned to exactly one fog device. Each task require a time to be transmitted to the fog-device, and the solution computed, we denote with $T(t_i, f_j)$ an (expected) time required to receive, compute and get back the solution of task t_i when it is assigned to fog device f_j. Let x_{ij} be boolean variables, such that $x_{ij} = 1$ iff task i is selected to be scheduled in fog device j. Note that, tasks scheduled to different fog-devices runs in parallel, so the overall metric for delay, is similar to *makespan* in parallel job scheduling problems. So, the problem can be stated as the following mixed integer-linear program:

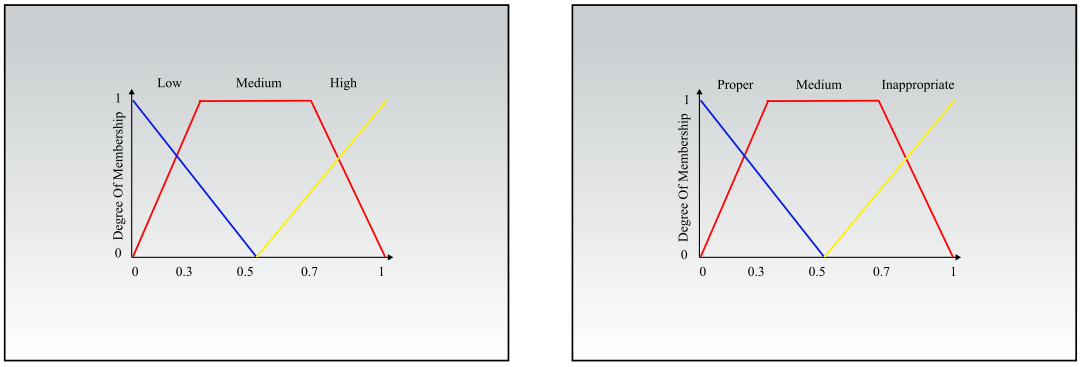
Problem 1 (*Task Scheduling*)**.**

$$minimize \quad z = \max_j \sum_{i=1}^{k} x_{ij} T(t_i, f_j) \tag{1}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \forall i \in [1, k] \tag{2}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in [1, k], j \in [1, m] \tag{3}$$

The optimization is performed, by minimizing the makespan of the schedule, i.e. the maximum of the load among the different fog-devices. Note that, in this mathematical formulation we need to know the exact values of $T(t_i, f_i)$, i.e. it is a deterministic and offline optimization problem. In practice we assume that each task has an estimated cost of execution t_i, and each fog-devices has a level of cpu-utilization f_j that is clearly related to the number of tasks in its execution queue. The F-DTA algorithm uses a fuzzy based optimization approach that is practical in most scenarios, since the exact determination of the values of $T(t_i, f_j)$ is difficult in most of the real cases.



(a) Input parameters for fuzzy function.

(b) Output parameter for fuzzy function.

Fig. 2. Fuzzy sets.

3.3. Fuzzy-based method (F-DTA):

As previously stated, the task manager's role inside the Fog gateways is to allocate the drones' tasks to the most suitable Fog devices. The Fog device assigned to the drones' tasks should minimize execution time. Our proposed method employs a lightweight algorithm to optimize resource utilization. It is worth mentioning that, like the former works, FPFTS, FUPE, and S-FoS [9–11], we employed a broker approach for node communications. As we discussed the architecture in Section 3.1, the task managers inside the fog gateways receive the drones' tasks and assign them to the most suitable Fog device inside the Fog region. As drones can travel inside the Fog regions, they send new requests to the Fog gateways that are responsible for task allocation inside the newly travelled Fog region. Accordingly, the assigned Fog devices are near the drones.

Resource utilization: IoT-Fog literature demonstrates that efficient task scheduling and allocation can be accomplished by simultaneously considering the resources' computing features and the tasks' CPU requirements [9–11,26–28]. Therefore, we use the CPU requirements of tasks and the available CPU of Fog devices as input values to fuzzy functions to optimize the performance. Mamdani is a popular fuzzy inference engine that employs sets and rules based on prior experience or predefined premises [9,11]. We create three overlapping fuzzy sets that allocate the input values in multiple sets simultaneously. As a result, each input value is located in at least two fuzzy sets with varying degrees of membership. For instance, consider Fig. 2(a). The blue line (low set) goes from (0,1) to (0.5, 0), and the yellow one (high set) goes from (0.5, 0) to (0,1). They overlapped with the medium set in red line. It means the input parameters between 0 and 0.5 are low and medium with two different degrees of membership. For instance, 0.3 is low and medium, with the degree of membership at 0.7. The point of intersection of two overlapped fuzzy sets indicates the degree of membership in the vertical axis.

For mindful readers, the question may arise that our proposed method is incomplete, as this would mean disregarding all aspects concerning the actual deployment of the tasks once they have been allocated and the delay and network usage. The contribution of this article is to focus on execution time and ignore delay, latency, energy consumption, and network delay to show the disadvantages of classic metaheuristic methods. As we used a simple Mamdani fuzzy function, we can add more input parameters, such as link bandwidth, nodes' energy consumption or other criteria, proposing a modified version of the proposed approach.

The fuzzy sets for the parameters above and the result parameter are as follows:

- $Task$: ∈ {Low; Medium; High}
- $Fog\ Device$: ∈ {Low; Medium; High}
- $Result$: ∈ {Proper; Medium; Inappropriate}

We use *Task* and *Fog Device* in the above fuzzy sets to represent the tasks' CPU requirement and the Fog devices' available CPU, respectively. Moreover, the fuzzy rules and fuzzy sets used in the fuzzy function are shown in Table 1 and Fig. 2. In this work, we define nine fuzzy rules and based on the input values, some are fired. The input values for both input fuzzy sets must be within the specified range for triggering the fuzzy rule. The centroid approach builds a non-fuzzy value as the outcome of the fuzzy function in the aggregation step using all of the values of the fired rules. A non-fuzzy number is the outcome of the fuzzy function, a number between zero and one. The task manager assigns the task to the fog device that has the higher value. In our former work, S-FoS [11], we discussed the difference between the triangle fuzzy functions and the isosceles trapezoid fuzzy functions. Accordingly, with the lessons we learned from FPFTS and S-FoS projects, we implemented an isosceles trapezoid fuzzy function. Besides, we explained how the fuzzy function obtained the results in the FUPE project by some numerical examples [10]. Moreover, we discussed defining the fuzzy settings and fuzzy rules in the FPFTS and S-FoS projects [9,11].

Mamdani fuzzy inference consists of five stages: fuzzification of the input variables, carrying out fuzzy operations and fuzzy reasoning, carrying out fuzzy inferences, aggregation of the fired rules and defuzzification of the results. It uses Table 1 and Fig. 2 in all the mentioned five steps to obtain the output non-fuzzy number. As the scope of our article does not discuss the details of fuzzy

Table 1
Fuzzy rules.

Rule	Tasks cost	Fog CPU	Result
1	Low	Low	Proper
2	Low	Medium	Medium
3	Low	High	Inappropriate
4	Medium	Low	Medium
5	Medium	Medium	Proper
6	Medium	High	Medium
7	High	Low	Inappropriate
8	High	Medium	Medium
9	High	High	Proper

logic, we refer enthusiastic readers to the S-FoS article [11] for more information about how the Mamdani fuzzy inference engine works. Algorithm 1 indicates the algorithm of F-DTA method for IoD task allocation. In the algorithm, the values of the function $T(t_i, f_j)$ described in the model 1 is evaluated using the fuzzy inference, and denoted by v_{ij}, the schedule is produced by selecting for each task the fog devices with higher value of this function. Note that the algorithm, always produce a feasible schedule since: the set of tasks managed by the fog-gateway comprise also idle tasks (that can be scheduled when there are few tasks) or can be moved to the cloud (if the overall load of the fog-devices is at the maximum, and the remaining tasks if scheduled would produce a makespan that is over a maximum threshold), eventually increasing the delay for some tasks. We can observe that in the latter case, the problem is more easily defined as a problem of setting the right SLA requirements of the entire architecture, and maybe solved by, for example, increasing the number of fog-devices deployed.

Algorithm 1 F-DTA

INPUT: m, k, $cpu(f_j)$, $load(t_i)$

m	Number of Fog devices
k	Number of Tasks
$cpu(f_j)$	Cpu level of Fog Device f_j
$load(t_i)$	Cost of Task t_i

1: Define isosceles trapezoid fuzzy function as the membership function
2: Define the rules for the fuzzy inference system
3: Define the centroid method as defuzzification to obtain the crisp output value.
4: **for** $i = 1$ to k **do**
5: **for** $j = 1$ to m **do**
6: Use the fuzzy system to compute the values of v_{ij}
7: based on the values of $load(t_i)$ and $cpu(f_j)$.
8: **end for**
9: $x_{ij} \leftarrow 1$ iff $j = argmax_j v_{ij}$, i.e. Assign task i to the fog device j with the max value of v_{ij}
10: Update the values of $cpu(f_j)$ to reflect the assignment.
11: **end for**
12: Output the schedule, i.e. variables x_{ij}.

How to use F-DTA method for Task allocation: We have a set of drone tasks and a set of Fog devices. When assigning a task to a Fog device, we run F-DTA for the task and each Fog devices to find the most proper one. We assign a Fog device to the task with the highest number, similar to the previous work, FPFTS [9]. Finally, as we aim to illustrate a simple, lightweight approach that considers the features of tasks and the resources much more practical and has less execution time than metaheuristic approaches such as PSO and KHA, we employed the F-DTA method in the PSO and KHA fitness functions.

3.4. Metaheuristic approaches

In the literature, researchers have employed metaheuristic approaches such as NSGA [29], PSO [30], Firework algorithm (FWA) [31], Whale optimization algorithm (WOA) [32], Bee life algorithm (BLA) [33], Harmony Search (HS) [34], and Tabu Search (TS) [35] for IoT applications. PSO is one of the most acceptable ones, and KHA is almost the newest one that reduced the iterations, so we chose them as the employed metaheuristic ones.

Particle Swarm Optimization (PSO): PSO is a popular optimization algorithm inspired by the collective behaviour of birds. In a PSO algorithm, a population of potential solutions, known as "particles", explores a solution space in search of the optimal solution. Each particle represents a potential solution and adjusts its position based on its own experience and the experiences of its neighbouring particles [36]. The algorithm refines the particle positions by moving them towards the best solutions found so far. This process employs how a flock of birds cooperatively adapts to find the most suitable direction. While PSO is a popular optimization algorithm with various applications, its suitability for online, real-time, and time-sensitive tasks is limited due to its inherent characteristics and computational needs.

Table 2
Entity requirements.

Requirements	Cloud	Fog device	Task
RAM	0	4 GB	10 MB
CPU	0	44800 MIPS	200 MIPS
Bandwidth	0	200 Mbps	100 Mbps

Krill Herd Algorithm (KHA): KHA is a bio-inspired optimization algorithm inspired by the collective behaviour of krill, small marine crustaceans. KHA is a population-based optimization technique that simulates krill's social interactions and movements in search of optimal solutions. In KHA, each potential solution is represented as a krill, and these individuals collectively explore the solution space by adjusting their positions. Krill update their positions based on various factors, including their experience and other neighbouring krill's influence [37]. The algorithm aims to improve the quality of solutions over time by employing the swarming behaviour of krill, where individuals cooperate to find more suitable solutions. KHA offers certain advantages over PSO. One key advantage is KHA's ability to adapt to dynamic and complex solution spaces. KHA's collective behaviour, inspired by krill swarming, allows it to better handle problems with varying environments. Additionally, KHA often requires fewer tuning parameters than PSO, making it more suitable for practical applications.
PSO and KHA fitness functions: In the PSO and KHA algorithms, the fitness function is a crucial component that determines the suitability of a Fog device to run the drone's tasks. The primary purpose of the fitness function is to evaluate how well a particular Fog device satisfies the optimization objectives and constraints. As our aim is to study the execution time of the metaheuristic algorithms, we used the proposed F-DTA method in their fitness function to have a fair comparison. We refer the enthusiastic readers to read the article [38] for more information about the Metaheuristic algorithms and fitness functions.

4. Performance evaluation

In this section, we clarify the evaluations we performed to assess the performance of the approaches. The performance evaluation depicts the impact of devising a task manager that considers resource utilization by employing metaheuristic techniques and a simple method (i.e., `F-DTA`). To implement our approach, we used iFogSim2 [39], a java-based simulator built on the CloudSim framework. We use this simulator because it supports mobility and clustering and is quite acceptable for task allocation approaches for Fog-enabled IoT applications. Moreover, it integrates real-world datasets into the simulation.

4.1. Simulation setup

We used the EUA dataset of iFogSim2 for the simulation environment because it supports mobility of the drones. We also used `XFuzzy` tool [40] to define the fuzzy rules and fuzzy sets. We employed six Fog gateways spread across six Fog regions. Besides, we used a broker approach and put the algorithms in the Fog gateways. The Fog gateways are the broker between the drones (things in the IoT) and the Fog devices in the Fog regions. Drones have a path, and the simulation will allow them to create several tasks at random points so they offload to the region brokers, and finally, Fog gateways assign them to the Fog devices. The scenarios that are actually running in the simulation contain drones flying to multiple Fog regions and also the simulation grid if formed by multiple Fog devices. We used *random_usersLocation_melbCBD_1.csv*, which contains the drone's mobility traces and represents the drone's path during the simulation and *edgeResources-melbCBD.csv* for fog layer's nodes configurations.

We used the particular features of the scenarios listed in Table 2 in order to simulate the methods, the configuration setup used in the simulation is shown in this Table. The computational capability and bandwidth (i.e., the amount of CPU, RAM, and bandwidth) of the cloud data centre and Fog devices are displayed. As we do not use Fog device to cloud data-centre data offloading in our simulation, we put zero for its features in the table. Each fog device is equipped with a reasonable amount of RAM and with a multi-core processor, we took the MIPS of a processor like ARM CORTEX-A73.[1] The Task column shows how much RAM, CPU, and bandwidth they require to run. We use the iFogSim2 entities to implement this functionality. Furthermore, we fixed the range of fuzzy sets for the security portion of our proposed task manager between zero and one. Next, we took the information out of the EUA dataset and divided it by the maximum value to normalize it. As a result, the fuzzy algorithm's actual value range falls between zero and one. After that, we defined the fuzzy sets and fuzzy rules using XFuzzy. Finally, we assessed the approaches using the iFogSim2.

4.1.1. Simulation metrics

The primary metric for the evaluation of the task scheduling and management approach is *execution time*. It is the time a task management unit takes to complete a task once it has received a request. It includes processing time but not the time for the response to travel back to the drone. For the execution time, we should consider the optimization result (the execution time of the tasks on the Fog devices) and the time it takes to execute the optimization itself. In our experiments, we considered both of them, but, it is worth mentioning that we kept all the settings the same for all three methods. We used a star topology, and the distance between the Fog devices and the Fog gateways and their configurations (links and Fog devices) are the same; the differences between the execution times for the three algorithms are mainly due to the second one.

[1] https://wiki2.org/en/Instructions_per_second

Table 3
Comparison of the Execution Time (in sec.) of F-DTA, PSO, and KHA with an increasing number of tasks.

Tasks	F-DTA	PSO	KHA
30	0.82	1402	720
50	1.36	2039	1102
70	1.53	3807	1613
90	2.05	4234	1956
110	2.38	4944	2408
150	2.98	5450	2717
200	4.18	6031	3119

4.1.2. Implementation scenarios

Metaheuristic approaches require a ready workload to choose the best answer among the many answers for the various drones. In the real environment, we cannot wait to have a huge workload. We should assign the most available and proper fog device to the request. That is why some real-world IoT applications still use First-Come-First-Serve (FCFS) scheduling algorithms. For instance, article [41] uses FCFS priority scheduling to detect and allocate empty parking spaces in a real-time manner. We implement a small network scale. The main reason is that we want to show that metaheuristic approaches do not have any merits when the size of the network is small or when we do not have a ready huge workload. Over the following various drones, and Fog devices scenarios, we compare our proposed approach to PSO and KHA approaches:

- *Scenario-1: Comparing approaches based on various number of drones:* The number of Fog devices in this scenario is 12 in 6 Fog regions. We vary the number of drones to adjust the number of tasks (requests) to study the effects of the number of requesters on PSO and KHA and compare the execution time with the F-DTA method we use in their fitness functions.
- *Scenario-2: Comparing approaches based on various number of Fog devices:* The number of drones in this scenario is one with 10 tasks. This scenario aims to illustrate the effects of the number of Fog devices on PSO and KHA and compare the execution time with the F-DTA method that we use in their fitness functions.

4.2. Evaluation results

This section shows the implementation results for the metaheuristic approaches and the F−DTA method. As this research aims to illustrate that metaheuristics are unsuitable for time-sensitive Fog-enabled IoD applications, we keep the settings the same for the three approaches. We used the F−DTA method in the PSO and KHA fitness functions. We just show the result of the execution time and discuss the network utilization and delay, stating that their results are sufficient for our research report.

First, we discuss the evaluation for the first scenario, comparing approaches based on various numbers of drones. In this scenario, we have 12 Fog devices, and we test the result of the methods by varying the drone numbers from 1 to 11. Table 3 summarizes the execution time of the F−DTA, PSO, and KHA methods. As they illustrate, there is a clear upward trend in execution time across all methods. There are more requests than there are drones, which means that the task management should distribute more work to the Fog devices.

The significantly lower execution time of the F−DTA method in the context of IoD task allocation is due to its simplicity. F−DTA, as a custom-designed method, likely involves straightforward calculations and minimal data processing, making it faster to evaluate. On the other hand, the KHA and PSO have more execution time. However, KHA showcases a lower execution time than PSO due to using the K-means algorithm. KHA is a nature-inspired optimization algorithm that emphasizes a more streamlined approach to population movement, often leading to quicker convergence towards a solution. In contrast, PSO involves more complicated mechanisms for particle update, global best position tracking, and computational overhead, contributing to a relatively longer execution time [42].

As we observed from simulation, The *network usage* of the three approaches was the same, because it is primarily determined by the communication requirements of the tasks and the data transfer between the Fog gateways and the drones. All three approaches allocate the same tasks to the same Fog device (due to using F-DTA method in the PSO and KHA fitness functions), resulting in similar network usage across these approaches. In other words, the allocation of tasks to drones is the same, so the network usage, which depends on these task assignments, remains unchanged. Moreover, when we have more drone tasks, the links should transfer more data, and accordingly, we see an increase in network usage.

Finally, we discuss the evaluation for the second scenario, comparing approaches based on various numbers of Fog devices. In this scenario, we have a drone with 10 tasks, and we test the result of the methods by varying the Fog device numbers from 18 to 180. Table 4 summarizes the execution time of the F-DTA, PSO, and KHA methods.

Some notable results emerged when we employed the F-DTA, PSO, and KHA methods with varying numbers of Fog devices. As the number of Fog devices increases from 18 to 180, the execution time for all three methods experiences an increase. The main reason is that, as the algorithms should consider more resources, it takes more time to select the proper one. However, comparing the execution times reveals that F-DTA consistently outperforms PSO and KHA across all scenarios. F-DTA exhibits significantly shorter execution times, in contrast, PSO and KHA, while effective optimization methods show notably longer execution times. This

Table 4
Comparison of the Execution Time (in sec.) of F-DTA, PSO, and KHA with an increasing number of Fog devices.

Fog devices	F-DTA	PSO	KHA
18	0.96	908	414
36	1.15	1696	857
60	1.69	2224	1060
90	2.37	3731	1544
120	2.60	3980	2244
150	3.50	6002	2702
180	4.45	7736	3074

outcome indicates the advantage of using the F-DTA method for IoT task allocation, as it consistently provides quicker results even as the scale of the problem increases, making it a valuable choice for time-sensitive and resource-constrained IoD-Fog environments.

In the evaluation scenario where the F-DTA, PSO, and KHA methods are employed for drone task allocation with varying numbers of Fog devices distributed across 6 Fog regions, a consistency emerges regarding *network usage*. Despite the increase in Fog devices, ranging from 18 to 180 across these regions, the network usage for all three methods remained constant. This result indicates that when we have very few requests (IoD tasks) when assigning to the resources (Fog devices), adding the Fog devices does not affect the network utilization. The reason is that we have six Fog regions, and each of them has a Fog gateway. We use the broker approach, and the instances of the algorithms are in the Fog gateways. More importantly, we used a star topology and the distance between the Fog devices, and the Fog gateways and their configurations (links and Fog devices) are the same. As a result, network usage remains the same.

In our experimental results, we evaluated the execution time and discussed network usage. As the result of the *delay* in a common simulator could be far from reality in the real-world, we did not assess it. The main reason is that the simulators simplify real-world requirements such as processing delay, transmission delay, propagation delay, queuing delay, sensing delay, task queuing delay, task execution delay, and load balancing delay to make them computationally feasible. These simplifications can lead to a significant difference between the simulation results and the real-world results. In the discussion Section 5.1, we discuss the problems of obtaining delay in common simulators. Still, to pave the way for the other researchers, we anticipate that the outcomes of the three approaches are almost the same in terms of delay. We anticipate delay remains nearly identical across the three methods, despite variations in execution times. The reason why we anticipate equivalent delays across these methods is due to employing F-DTA in the PSO and KHA fitness functions [43].

4.2.1. Time complexity

We use metaheuristics if the problem is so complex that precise optimization takes too much time. It is usually proved by showing that the problem is NP-Hard. If the problem is simpler and an efficient polynomial solution exists, metaheuristics are unmotivated. For the time-sensitive Fog-enabled IoD applications, the time complexity should be very low, and Fog gateways do not have enough capacity to run metaheuristics. The task manager should assign the tasks to the most available Fog device, and cannot wait to gather a huge amount of tasks, and then repeat the iterations and find the best Fog device. Accordingly, their practicality fades away in real scenarios. As the primary goal of our article is to show the execution time of the mentioned algorithms, to wrap up this section, we discuss the time complexity of the PSO, KHA, and F-DTA methods. The time complexity of a Mamdani fuzzy method is $\mathcal{O}(1)$ due to performing all the steps in a parallel manner [44]. But as the F-DTA assigns each task to a Fog device, by iterating among k tasks and m fog devices, the overall complexity is $\mathcal{O}(k \cdot m)$. The time complexity of the PSO and KHA methods are $\mathcal{O}(i \cdot m \cdot f)$, where i, m, and f are the number of iterations, the population size (i.e., number of Fog devices), and fitness function, respectively. When F-DTA is used as the fitness function, the time complexity of the PSO and KHA methods are $\mathcal{O}(i \cdot k \cdot m^2)$.

5. Discussion

In this section, we discuss some challenges in devising, evaluating and implementing metaheuristic approaches in IoD-Fog networks that inspire researchers and pave the way for future research directions for developing practical applications.

5.1. The limitations of IoT simulators

In our study, the utilization of metaheuristic methods for IoD-Fog task allocation was explored, and we encountered particular challenges inherent in using common simulators focusing on execution time. We acknowledge that, at times, the choice of simulator is constrained, and in our case, we opted for a Java-based simulator. However, it is imperative to highlight that the results obtained from such simulators may not truly mirror real-world scenarios, thus potentially impacting the validity of our proposed approach [45]. The unreliability of common IoT simulators can be attributed to several factors: Firstly, these simulators often need the incorporation of essential IoT features and network layers, which makes it challenging to replicate the characteristics of real-world settings; Additionally, the parameters used in these simulators are frequently based on assumptions and approximations, potentially failing to capture the nuanced conditions of the network accurately [46].

To take an example, consider the delay metric. Many elements that are difficult to accurately model in a simulator, such as packet loss, and network congestion, affect the delay parameter. Common simulators frequently obtain delay based on approximations and assumptions that may not accurately represent the actual conditions of the network. Besides, the simulators do not consider the computation delay of a Fog gateway, but in reality, it tends to be far different due to caching in processors. In a nutshell, the absence of network layers, network-level features, and the simplified IoT and network models in the simulator significantly limits its ability to model and evaluate delays in real-world scenarios accurately. The simulator's architecture does not capture the characteristics of communication protocols and networking dynamics, which are critical factors in determining the delay in IoT task allocation.

5.2. IoD simulators

In addition to the commonly used simulators, there are several simulation tools, namely IoD-Sim, AirSim, Gazebo, and RealFlight's drone simulator. These simulators do not have abilities to implement task allocation approaches and evaluate drone task allocation metrics, as their primary purpose is to simulate drone movement and interactions within an environment. For instance, IoD-Sim creates realistic simulations of the UAV by extending the features of ns-3 to cover various aspects of IoD, including mission design, trajectory planning, hardware and application configuration, mobile wireless communication, and energy consumption models [47,48]. A well-liked simulator for testing and simulating robotic systems, including drones, is the Gazebo. Gazebo, however, is primarily concerned with mimicking automated systems' sensors and physical components. It is particularly good at simulating drone dynamics, sensor data, and environmental interactions [49].

5.3. The productivity of IoD task allocation approaches

Metaheuristic approaches such as PSO and KHA have been widely explored and refined by researchers seeking to optimize task allocation in IoT applications. However, their practical productivity often needs to improve when applied to time-sensitive applications. Although they may yield excellent solutions in controlled, non-real-time environments, the time required for these methods to converge can be prohibitive in time-sensitive scenarios such as drone operations. Consequently, the high execution time of metaheuristic approaches can significantly hinder their productivity and practical utility in time-sensitive IoD applications.

5.4. Challenges in bridging academic theory and practical productivity

This academic research helps build a foundation of knowledge that can eventually lead to more practical and efficient solutions. Although metaheuristic approaches may not be immediately suitable for time-sensitive Fog-enabled IoD applications because of their high execution time, research in academia serves as a stepping stone for future work. The theoretical insights gained from studying metaheuristics can inform the development of hybrid methods or improved algorithms that better balance theoretical considerations and practical applications. While these approaches might not have direct productivity in real-world applications, they contribute to the academic area, enabling a deeper understanding of optimization methods and inspiring further innovation that may eventually bridge the gap between academic theory and practical productivity in time-sensitive IoD applications, such as drone operations.

6. Conclusions and future directions

Practical task allocation and scheduling are essential for achieving optimal performance in the fog-enabled Internet of Drones and numerous researchers have developed metaheuristic-based task allocation techniques. Empirical studies, case studies, and benchmarking data support the notion that metaheuristic algorithms while promising in the academic realm, may have limitations in real-time environments because of their high execution times, resource-intensive nature, increased temporal complexity, and inherent uncertainty in reaching optimal solutions. This paper presented `F-DTA`, a lightweight and straightforward task allocation method that serves as the fitness function for two metaheuristic methods: the Krill Herd Algorithm (KHA) and Particle Swarm Optimization (PSO). We used the iFogSim2 simulator to simulate our suggested methodology, and the findings validate that F-DTA performs better than metaheuristic methods in terms of execution time.

CRediT authorship contribution statement

Saeed Javanmardi: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Georgia Sakellari:** Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Mohammad Shojafar:** Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Antonio Caruso:** Conceptualization, Methodology, Validation, Writing – review & editing, Supervision, Project administration.

Data availability

No data was used for the research described in the article.

Acknowledgements

Partial funding for this project has been provided by Puglia Region, Italy, with the iniziative "*RIPARTI-assegni di RIcerca per riPARTire con le Imprese*". RIPARTI supports the professional development of researchers and their integration into the regional production ecosystem through research grants.

References

[1] M. Chiang, T. Zhang, Fog and IoT: An overview of Research Opportunities, IEEE Internet Things J. 3 (6) (2016) 854–864.
[2] K. Shafique, B.A. Khawaja, F. Sabir, S. Qazi, M. Mustaqim, Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios, IEEE Access 8 (2020) 23022–23040.
[3] J. Yao, N. Ansari, Qos-aware power control in internet of drones for data collection service, IEEE Trans. Veh. Technol. 68 (7) (2019) 6649–6656.
[4] J. Yao, N. Ansari, Online task allocation and flying control in fog-aided internet of drones, IEEE Trans. Veh. Technol. 69 (5) (2020) 5562–5569.
[5] D. Rahbari, Analyzing meta-heuristic algorithms for task scheduling in a fog-based IoT application, Algorithms 15 (11) (2022) 397.
[6] K. Rajwar, K. Deep, S. Das, An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges, Artif. Intell. Rev. (2023) 1–71.
[7] J. Min, M. Oh, W. Kim, H. Seo, J. Paek, Evaluation of metaheuristic algorithms for TAS scheduling in time-sensitive networking, in: 2022 13th International Conference on Information and Communication Technology Convergence, ICTC, IEEE, 2022, pp. 809–812.
[8] O. Kebriyaii, A. Heidari, M. Khalilzadeh, J. Antucheviciene, M. Pavlovskis, Application of three metaheuristic algorithms to time-cost-quality trade-off project scheduling problem for construction projects considering time value of money, Symmetry 13 (12) (2021) 2402.
[9] S. Javanmardi, M. Shojafar, V. Persico, A. Pescapè, FPFTS: a joint fuzzy particle swarm optimization mobility-aware approach to fog task scheduling algorithm for internet of things devices, Softw. - Pract. Exp. 51 (12) (2021) 2519–2539.
[10] S. Javanmardi, M. Shojafar, R. Mohammadi, A. Nazari, V. Persico, A. Pescapè, FUPE: A security driven task scheduling approach for SDN-based IoT–fog networks, J. Inf. Secur. Appl. 60 (2021) 102853.
[11] S. Javanmardi, M. Shojafar, R. Mohammadi, V. Persico, A. Pescapè, S-fos: A secure workflow scheduling approach for performance optimization in SDN-based IoT-fog networks, J. Inf. Secur. Appl. 72 (2023) 103404.
[12] S. Javanmardi, M. Shojafar, R. Mohammadi, M. Alazab, A.M. Caruso, An SDN perspective IoT-fog security: A survey, Comput. Netw. 229 (2023) 109732, Publisher: Elsevier.
[13] X. Zhang, X. Chen, UAV task allocation based on clone selection algorithm, Wirel. Commun. Mob. Comput. 2021 (2021).
[14] A. Caruso, S. Chessa, S. Escolar, F. Rincón, J.C. López, Task scheduling stabilization for solar energy harvesting internet of things devices, in: 2022 IEEE Symposium on Computers and Communications, ISCC, 2022, pp. 1–6, http://dx.doi.org/10.1109/ISCC55528.2022.9913061.
[15] M. Kuzman, X.d. Toro García, S. Escolar, A. Caruso, S. Chessa, J.C. López, A testbed and an experimental public dataset for energy-harvested IoT solutions, in: 2019 IEEE 17th International Conference on Industrial Informatics, Vol. 1, INDIN, 2019, pp. 869–876, http://dx.doi.org/10.1109/INDIN41052.2019.8972219.
[16] M. Deng, Z. Yao, X. Li, H. Wang, A. Nallanathan, Z. Zhang, Dynamic multi-objective AWPSO in DT-assisted UAV cooperative task assignment, IEEE J. Sel. Areas Commun. (2023).
[17] S. Xu, L. Li, Z. Zhou, Y. Mao, J. Huang, A task allocation strategy of the UAV swarm based on multi-discrete wolf pack algorithm, Appl. Sci. 12 (3) (2022) 1331.
[18] J. Zhu, X. Wang, H. Huang, S. Cheng, M. Wu, A NSGA-II algorithm for task scheduling in UAV-enabled MEC system, IEEE Trans. Intell. Transp. Syst. 23 (7) (2021) 9414–9429.
[19] I.Z. Yakubu, M. Murali, An efficient meta-heuristic resource allocation with load balancing in IoT-fog-cloud computing environment, J. Ambient Intell. Humaniz. Comput. 14 (3) (2023) 2981–2992.
[20] Y. Wang, Z.-Y. Ru, K. Wang, P.-Q. Huang, Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing, IEEE Trans. Cybern. 50 (9) (2019) 3984–3997.
[21] J. Schwarzrock, I. Zacarias, A.L. Bazzan, R.Q. de Araujo Fernandes, L.H. Moreira, E.P. de Freitas, Solving task allocation problem in multi unmanned aerial vehicles systems using swarm intelligence, Eng. Appl. Artif. Intell. 72 (2018) 10–20.
[22] P. Hu, S. Dhelim, H. Ning, T. Qiu, Survey on fog computing: architecture, key technologies, applications and open issues, J. Netw. Comput. Appl. 98 (2017) 27–42.
[23] P.G.V. Naranjo, Z. Pooranian, M. Shojafar, M. Conti, R. Buyya, FOCAN: A fog-supported smart city network architecture for management of applications in the internet of everything environments, J. Parallel Distrib. Comput. 132 (2019) 274–283.
[24] M. Shojafar, S. Javanmardi, S. Abolfazli, N. Cordeschi, FUGE: A joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and a genetic method, Cluster Comput. 18 (2015) 829–844.
[25] M. Ghobaei-Arani, A. Souri, A.A. Rahmanian, Resource management approaches in fog computing: a comprehensive review, J. Grid Comput. 18 (1) (2020) 1–42.
[26] S. Javanmardi, S. Shariatmadari, M. Mosleh, A novel decentralized fuzzy based approach for grid resource discovery, Int. J. Innovat. Comput. 3 (1) (2013).
[27] H. Ben Alla, S. Ben Alla, A. Touhafi, A. Ezzati, A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment, Cluster Comput. 21 (4) (2018) 1797–1820.
[28] S.P. Singh, A. Nayyar, H. Kaur, A. Singla, Dynamic task scheduling using balanced VM allocation policy for fog computing platforms, Scalable Comput. Practice Exper. 20 (2) (2019) 433–456.
[29] K. Peng, H. Huang, B. Zhao, A. Jolfaei, X. Xu, M. Bilal, Intelligent computation offloading and resource allocation in iIoT with end-edge-cloud computing using NSGA-III, IEEE Trans. Netw. Sci. Eng. (2022).
[30] T. Salehnia, A. Seyfollahi, S. Raziani, A. Noori, A. Ghaffari, A.R. Alsoud, L. Abualigah, An optimal task scheduling method in IoT-fog-cloud network using multi-objective moth-flame algorithm, Multimedia Tools Appl. (2023) 1–22.
[31] A.M. Yadav, K.N. Tripathi, S. Sharma, A bi-objective task scheduling approach in fog computing using hybrid fireworks algorithm, J. Supercomput. 78 (3) (2022) 4236–4260.
[32] S. Chakraborty, A.K. Saha, A. Chhabra, Improving whale optimization algorithm with elite strategy and its application to engineering-design and cloud task scheduling problems, Cogn. Comput. (2023) 1–29.
[33] P. Narayana, C. Jatoth, P. Paravataneni, G. Rekha, An efficient optimizing energy consumption using modified BEE colony optimization in fog and IoT networks, Big Data Anal. Fog-Enabled IoT Netw. Towards Privacy Secur. Perspect. (2023) 197.
[34] F. Xu, Z. Yin, G. Han, Y. Li, F. Zhang, Y. Bi, Multi-objective fog node placement strategy based on heuristic algorithms for smart factories, Wirel. Netw. (2023) 1–18.
[35] P. Memari, S.S. Mohammadi, F. Jolai, R. Tavakkoli-Moghaddam, A latency-aware task scheduling algorithm for allocating virtual machines in a cost-effective and time-sensitive fog-cloud architecture, J. Supercomput. 78 (1) (2022) 93–122.

[36] T.M. Shami, A.A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M.A. Summakieh, S. Mirjalili, Particle swarm optimization: A comprehensive survey, IEEE Access 10 (2022) 10031–10061.
[37] G.-G. Wang, A.H. Gandomi, A.H. Alavi, D. Gong, A comprehensive review of krill herd algorithm: variants, hybrids and applications, Artif. Intell. Rev. 51 (2019) 119–148.
[38] M. Abdel-Basset, L. Abdel-Fatah, A.K. Sangaiah, Metaheuristic algorithms: A comprehensive review, Comput. Intell. Multimedia Big Data Cloud Eng. Appl. (2018) 185–231.
[39] R. Mahmud, S. Pallewatta, M. Goudarzi, R. Buyya, iFogSim2: An extended iFogSim simulator for mobility, clustering, and microservice management in edge and fog computing environments, J. Syst. Softw. 190 (2022) 111351.
[40] Xfuzzy, 2017, Fuzzy Logic Design Tools, Instituto De MicroElectronica De Sevilla, IMSE-CNM, http://www2.imse-cnm.csic.es/Xfuzzy/.
[41] M. Veeramanickam, B. Venkatesh, L.A. Bewoor, Y.W. Bhowte, K. Moholkar, J.L. Bangare, IoT based smart parking model using arduino UNO with FCFS priority scheduling, Measurement: Sensors 24 (2022) 100524.
[42] H. Kumar, et al., A new hybrid particle swarm optimizationalgorithm for optimal tasks scheduling in distributed computing system, Intell. Syst. Appl. 18 (2023) 200219.
[43] A. Yousefpour, G. Ishigaki, J.P. Jue, Fog computing: Towards minimizing delay in the internet of things, in: 2017 IEEE International Conference on Edge Computing, EDGE, IEEE, 2017, pp. 17–24.
[44] S. Javanmardi, M. Shojafar, S. Shariatmadari, S.S. Ahrabi, FR trust: a fuzzy reputation–based model for trust management in semantic P2P grids, Int. J. Grid Utility Comput. 6 (1) (2015) 57–66.
[45] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, J.P. Jue, All one needs to know about fog computing and related edge computing paradigms: A complete survey, J. Syst. Archit. 98 (2019) 289–330.
[46] M. Chernyshev, Z. Baig, O. Bello, S. Zeadally, Internet of things (IoT): Research, simulators, and testbeds, IEEE Internet Things J. 5 (3) (2017) 1637–1647.
[47] J. Yao, N. Ansari, Wireless power and energy harvesting control in iod by deep reinforcement learning, IEEE Trans. Green Commun. Netw. 5 (2) (2021) 980–989.
[48] G. Grieco, G. Iacovelli, P. Boccadoro, L.A. Grieco, Internet of drones simulator: Design, implementation, and performance evaluation, IEEE Internet Things J. 10 (2) (2022) 1476–1498.
[49] A. Farley, J. Wang, J.A. Marshall, How to pick a mobile robot simulator: A quantitative comparison of CoppeliaSim, gazebo, MORSE and webots with a focus on accuracy of motion, Simul. Model. Pract. Theory 120 (2022) 102629.