JIRBDAI

Journal of Institutional Research, Big Data Analytics and Innovation

Systematic Review of Task Scheduling in Fog–Cloud Computing: Machine Learning and Metaheuristic Approaches

[1]Ayuba Liman, [2] Rafiu Mope Isiaka,[3] Nathaniel Akinbowale Babatunde, [4]Aminu Jafar, [5]Toyyib Olaitan Olanrewaju [6]Mustapha Abubakar Giro

[1,4,6] Department of Computer Science Abdullahi Fodio University of science and Technology, Aliero, Kebbi State
[2,3,5] Department of Computer Science, Kwara State University, Malete
*Corresponding Author email address: limanjega01@gmail.com

ABSTRACT

Fog–cloud computing has emerged as a hybrid infrastructure designed to support low-latency, scalable, and intelligent processing for modern Internet of Things (IoT) applications. Efficient task scheduling within this environment remains a complex and NP-hard challenge due to heterogeneous distributed resources, fluctuating workloads, and diverse Quality of Service (QoS) requirements. This study presents a systematic review of state-of-the-art task scheduling techniques published between 2018 and 2025, with a focus on machine learning, heuristic-based, and metaheuristic-based approaches. The reviewed literature was classified into a taxonomy highlighting methodological characteristics, evaluation metrics, and simulation tools used in fog–cloud environments. Analysis of the selected studies indicates that energy consumption and makespan time are the most prioritized performance metrics, while Python is the most frequently used simulation environment. The findings reveal that although metaheuristic algorithms dominate the field due to their adaptability and effectiveness in large search spaces, hybrid approaches integrating artificial intelligence show increasing potential for improving scheduling performance. Finally, research gaps, limitations, and future directions are discussed, emphasizing the need for real-world implementations, improved QoS handling, and advanced hybrid optimization frameworks

KEYWORDS: Fog–Cloud Computing, Task Scheduling, Metaheuristic Algorithms, Machine Learning, Heuristic-Based Techniques, Energy Optimization, Quality of Service (QoS).

I. INTRODUCTION

Cloud computing provides on-demand access to scalable computing resources through a networked infrastructure, enabling flexible, cost-effective, and pay-per-use service delivery models such as Infrastructure as a Service, Platform as a Service, and Software as a Service [1]. While cloud computing has been widely adopted for large-scale data storage and processing, its centralized nature introduces limitations in supporting latency-sensitive, mobility-aware, and real-time Internet of Things (IoT) application[2].. To address these limitations, fog computing was introduced as a complementary paradigm that extends cloud services closer to end devices by deploying computational and storage resources at the network edge[3]. By reducing communication latency and alleviating core network congestion, fog–cloud architectures enable

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

efficient support for emerging applications such as smart healthcare, intelligent transportation systems, and augmented reality[4][5]. However, the coexistence of heterogeneous fog and cloud resources introduces significant challenges in efficient task scheduling, particularly when balancing performance objectives such as latency, energy consumption, and resource utilization[6]. Task scheduling in fog–cloud environments aims to determine optimal task-to-resource mappings under dynamic and heterogeneous conditions while satisfying Quality of Service (QoS) requirements[7]. This problem is inherently NP-hard due to the large search space, conflicting optimization objectives, and variability in workload and resource availability[8]. As a result, conventional deterministic approaches are often insufficient to achieve satisfactory performance across diverse scenarios[9]. To cope with these challenges, researchers have increasingly adopted heuristic, metaheuristic, and machine learning–based scheduling approaches. Heuristic methods offer low-complexity solutions for predictable environments, metaheuristic algorithms provide near-optimal solutions for complex multi-objective optimization problems, and machine learning techniques enable adaptive decision-making under dynamic conditions[10]. Despite the growing body of literature, existing surveys often focus on a single class of approaches or limited performance objectives, providing fragmented insights[11][12]. Motivated by these limitations, this paper presents a systematic literature review of task scheduling in fog–cloud computing, offering a unified comparison of heuristic, metaheuristic, and machine learning approaches. The review further analyzes simulation tools, QoS metric frequency, and research trend evolution within a PRISMA-guided framework, providing a comprehensive and up-to-date synthesis of recent advances in fog–cloud task scheduling. Table 1 summarizes the distinctive characteristics of this review in comparison with representative existing surveys on fog–cloud task scheduling, highlighting differences in coverage period, scope, and analytical depth

Table 1: Comparison of This Review with Existing Surveys on Fog–Cloud Task Scheduling

Feature / Aspect	Existing Surveys	This Review
Coverage Period	Mostly up to 2022–2023	Extends to 2024–2025
Review Type	Narrative or partial surveys	PRISMA-based systematic literature review
Scheduling Categories	Single or dual focus (heuristic or metaheuristic)	Tri-category comparison (heuristic, metaheuristic, ML)
Machine Learning Coverage	Limited or fragmented	Comprehensive and integrated
QoS Metrics Analysis	Listed metrics only	Frequency-based QoS analysis
Simulation Tools Analysis	Rarely discussed	Explicit comparison of simulators and tools
Trend Evolution Analysis	Not included	Temporal trend and research evolution analysis
Application Context Mapping	Limited	Mapped across fog–cloud use cases

The key contributions of this research include the following:

(a). Presenting a comprehensive systematic review on tasks scheduling techniques in fog – cloud environment from 2018 – 2025, assessing their advantages and limitations, along with the tools employed for their implementation.

(b). A comparative analysis of different quality-of-service (QoS) metrics, along with their usage and corresponding percentages, was conducted.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

(c). An extensive analysis of simulation tools used in fog –cloud computing
(d). Highlighting unresolved challenges to suggest potential directions for future research.

Review Questions

This study outlines the following research questions:
RQ1: *Which task scheduling Approaches have been extensively applied in fog–cloud computing environments between 2018 and 2025?*
RQ2: *What quality-of-service (QoS) metrics are commonly used in evaluating task scheduling techniques in fog–cloud computing?*
RQ3*: What simulation tools are currently being used in fog–cloud computing research, and how frequently are they adopted in existing studies?*
RQ4: *What future research directions have been proposed to overcome these challenges, and how feasible are they based on current technological trends?*

2. METHODOLOGY

A Systematic Literature Review (SLR) was carried out to investigate and classify task scheduling techniques within the fog–cloud computing domain. The SLR process followed a well-defined search strategy to ensure the retrieval of relevant and high-quality studies. As shown in Table 1**,** the literature search was conducted using multiple reputable scientific databases, including Google Scholar, ScienceDirect, SpringerLink, and IEEE Xplore**,** with their respective URLs provided. To capture temporal trends in research activity, Figure 2 illustrates the distribution of selected journal publications over time, with a concentration of studies published between 2018 and 2025. The filtration and selection process of the retrieved articles is also depicted in Figure 3**,** demonstrating the step-by-step screening used to refine the pool of research papers on task scheduling in online and distributed environments. Given the vast number of studies available, the review focused primarily on peer-reviewed journal articles**,** known for their rigorous publication standards. The literature search spanned five major databases**,** targeting works that addressed scheduling in fog–cloud architectures. Relevant studies were selected based on clearly defined inclusion and exclusion criteria**,** as detailed in Table 2 of Section 2. Although the primary analysis centered on publications from 2018 to 2025**,** earlier works from 2018 onwards were also considered to provide historical context and trace the evolution of key techniques and trends in task scheduling. To ensure the methodological rigor of the selected studies, a quality assessment framework was applied to all included articles. Each study was evaluated using five criteria: (i) clarity of problem formulation, (ii) appropriateness of the scheduling approach, (iii) adequacy of performance metrics, (iv) transparency of experimental setup and simulation tools, and (v) relevance to fog–cloud task scheduling. Each criterion was scored on a binary scale (1 = satisfied, 0 = not satisfied), resulting in a maximum quality score of 5. Studies with a score below 3 were excluded from the final analysis to maintain the overall quality and reliability of the review

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

Table 2: Database Table and their URL

Search database	Web address
ScienceDirect	https://www.sciencedirect.com
IEEE Xplore	https://ieeexplore-ieee-org
Springer	https://link.springer.com
Google scholar	https://scholar.google.com
ResearchGate	https://www.researchgate.net

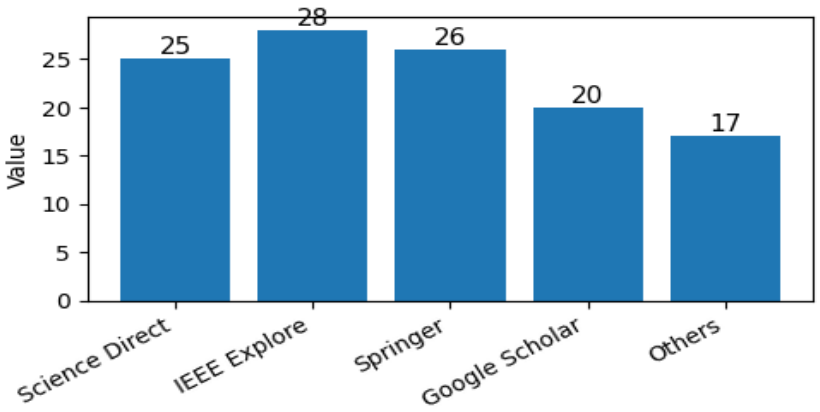

Figure 1: Database Distribution

The search strategy was instrumental in identifying relevant literature and deepening our understanding of the key concepts, terms, and keywords related to fog–cloud computing. It also addressed issues surrounding task scheduling techniques aimed at improving quality-of-service (QoS) metrics. To locate pertinent studies, specific keywords and phrases were strategically combined during the search process across research databases. As a result, the search string used in the library's search engine was: "((Task scheduling) AND Fog - cloud computing) AND (Machine learning Approach) AND (Heuristic Approach) AND (Meaheuristics pproaches)." This method effectively retrieved suitable publications, which were then downloaded for detailed analysis.

2.1 Process of Study Selection

In this section, we reviewed the titles, abstracts, and conclusions of all articles retrieved from scientific databases during the initial search phase, which yielded a total of 548 papers. Based on the relevance to our research questions, we identified 108 studies as highly relevant and excluded 440 papers that did not meet the criteria. The inclusion and exclusion criteria were systematically applied to the selected 108 papers. The PRISMA 2020 flow diagram, shown in Figure 3, illustrates the screening process and the number of records identified, included, and excluded, following the PRISMA framework. Each paper from the initial pool was carefully evaluated before being

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

considered for inclusion. The specific inclusion criteria used for this review are outlined in Table 2 Clear inclusion and exclusion criteria were applied at multiple screening stages to ensure transparency and reproducibility. Studies were excluded if they (i) fell outside the defined publication period (2018–2025), (ii) did not focus on task scheduling in fog–cloud or edge–cloud environments, (iii) lacked sufficient experimental or simulation-based evaluation, (iv) were not written in English, or (v) were non-peer-reviewed sources such as editorials or short abstracts. These exclusion decisions were consistently applied during title, abstract, and full-text screening phases, as illustrated in the PRISMA flow diagram.

Table 3: Inclusion and Exclusion Criteria

S/N	Inclusion	Exclusion
1.	The publication date of the selected article falls within the period from 2018 to 2025	At the outset, a filtering process was applied, targeting studies published between 2018 to present. After this step, the number of relevant papers was reduced to 196 from the original pool of 549 publications
2.	Research focused on task scheduling methods in fog–cloud computing environments	In the second stage of filtering, some papers were eliminated based on their titles and keywords. This step reduced the number of selected papers to 144**.**
3.	The publications were written in the English language.	During the third screening phase, papers were removed after evaluating their abstracts, reducing the selection from 144 to 119 studies.
4.	Journals, review articles, and conference proceedings	In the final filtering stage, only studies specifically focused on scheduling were retained for the proposed research, leading to a final selection of 108 papers.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

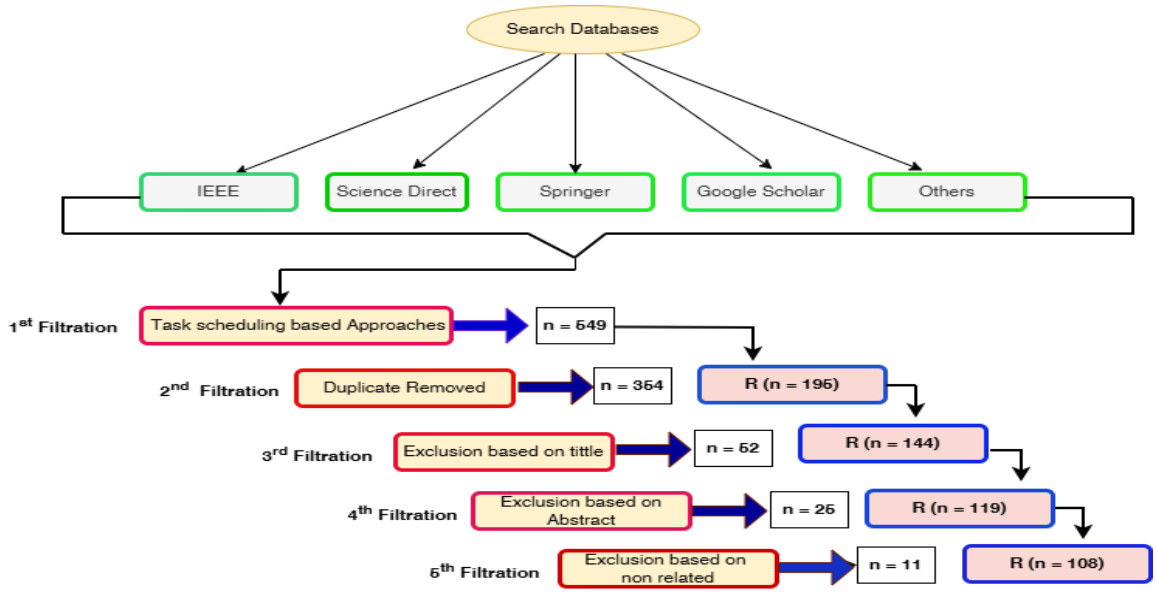

Figure 2: Filtration Process

2.2. Performance Metrics

Performance evaluation metrics for task offloading, scheduling, and allocation are essential for assessing the efficiency and effectiveness of various methods under different conditions[13] Especially in distributed computing environments such as fog and cloud computing, these metrics play a crucial role in measuring system performance and enhancing the Quality of Service (QoS) by meeting specific requirements and constraints[14][15] While numerous metrics are employed to evaluate QoS in task management, the most commonly referenced ones in the literature are discussed in the following subsections.

(a). Makespan

Makespan represents the total duration needed to complete a workflow, starting from the submission of the first task to the generation of the final output. It is the time interval between a user's workflow submission and the delivery of the results. Most scheduling approaches in existing literature emphasize minimizing this makespan[16]. By decreasing the overall execution time, tasks can be assigned to available resources more effectively, leading to reduced execution costs. Consequently, minimizing the makespan becomes a primary goal in workflow scheduling, as it helps shorten the total schedule duration while satisfying user requirements. The mathematical representation is given as follows

$$M = max_{i\in\{1,2....n\}}\{Ci\} \quad (1)$$

(b). Energy Consumption

Energy consumption serves as an essential performance indicator in computing systems, as it directly affects both operational expenses and environmental sustainability. It is defined as the total amount of energy utilized during the execution of computational tasks[17]. Enhancing energy efficiency is especially important in power-limited systems such as mobile devices and remote sensors, where optimal energy utilization is crucial for ensuring sustainability and reliability. In fog computing environments, managing energy consumption poses a significant challenge, as fog nodes often depend on limited battery power. This makes it necessary to develop effective

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

strategies that balance system performance with energy usage[18]. The mathematical expression is as follows

$$\mathbf{E_{total}} = \sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{P_i} \times \mathbf{T_i} \quad (2)$$

(c). Completion Time

In fog computing, completion time refers to the total period needed for a task to be processed and its results returned to the originating device. It encompasses the entire workflow from task generation on the user device to the final transmission of the processed output ensuring efficient task execution across the network[19]. Completion time is a vital performance metric because it directly influences system efficiency and user satisfaction in fog computing environments, especially for applications that demand real-time or low-latency responses[20]. Mathematically, the completion time can be expressed as follows:

$$CT_i = T_{wait,i} + T_{trans,i} + T_{exec,i} + T_{return,i} \quad (3)$$

(d). Execution time

Execution time represents the total period needed to complete all tasks assigned within a system. In this study, minimizing execution time is a primary goal aimed at improving task processing speed, which is crucial for latency-sensitive applications in fog computing environments[21][22] This approach ensures efficient utilization of fog node resources, minimizes delays, and enhances overall system performance through optimized task allocation and the implementation of the improved Particle Swarm Optimization (PSO) algorithm [23].

(e). Latency

Latency is an essential performance metric in distributed computing, especially in fog computing, where data processing is performed closer to the source to significantly reduce delays[24]. In task scheduling, latency denotes the time interval between the initiation of a task and the commencement of its execution, along with the duration needed to complete the task[25]. This metric is crucial for applications that demand real-time processing such as mobile devices, real-time analytics, and interactive applications where any delay can adversely affect user experience or system performance. Mathematically, latency can be represented as follows:

$$L_i = T_{queue,i} + T_{trans,i} + T_{prop,i} + T_{exec,i} + T_{return,i} \quad (4)$$

(f). Reliability

Reliability is a crucial objective in fog computing task scheduling, focused on ensuring consistent and dependable execution of tasks and services[26]. It is essential for meeting user expectations, maintaining service-level agreements (SLAs), and building trust in fog and cloud infrastructures. As a scheduling objective, reliability highlights the significance of fault tolerance, system stability, and uninterrupted service availability. Scheduling algorithms strengthen reliability by incorporating techniques such as redundancy, fault recovery, and proactive error handling to ensure seamless and continuous edge operations.

(g). Availability

In fog computing, availability as a scheduling objective emphasizes ensuring that computing resources and services remain consistently accessible and operational whenever needed. It is a key

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

factor in maintaining system dependability and delivering a smooth, uninterrupted user experience[27] Availability is generally measured by the percentage of time a system or service remains operational and accessible. Scheduling strategies aim to maximize system uptime, ensuring that tasks are executed with minimal disruption. This objective is closely related to reliability, as it represents the system's ability to deliver continuous and consistent service performance.

(h).Security

As a scheduling objective, security entails the application of strategies and protective measures designed to safeguard resources, data, and services against unauthorized access, cyberattacks, and data breaches[28]. This is particularly important in shared environments where sensitive data is stored and processed across multiple users. Security-oriented scheduling ensures that protective measures are integrated into the task execution process to maintain data integrity and confidentiality[29]. This is achieved by incorporating technical controls, security policies, and best practices. Such measures help preserve user trust, promote regulatory compliance, and ensure a secure, resilient, and reliable computing environment.

(i). Throughput

In distributed computing (including fog - cloud and IoT systems), throughput refers to the rate at which tasks are successfully processed and completed within a given period of time[30].It measures system efficiency how much useful work (tasks, jobs, or data) is accomplished per unit time. To achieve maximum throughput, [31] introduced an innovative hybrid multi-objective optimization algorithm designed to address the complex task scheduling challenges in edge computing environments. This approach aims to balance the objectives of both service providers and users, thereby improving overall system efficiency and resource utilization. The algorithm concurrently optimizes several critical factors, including minimizing makespan, reducing workload imbalance, and maximizing resource utilization as well as system throughput

2.3. Fog Computing Architecture

Fog computing is a decentralized architecture positioned close to end users to handle and process large volumes of data efficiently. The typical fog computing environment consists of three layers, as illustrated in Figure 4. This architecture is particularly well-suited for IoT applications, where numerous distributed devices require collaboration, data storage, and computational processing [32]. As shown in Figure 4, various devices such as smart phones, sensors, and embedded systems are located within the IoT device layer**.** This layer is responsible for collecting data from sensor devices and transmitting it to the fog layer. The fog layer includes routers, gateways, switches, workstations, and access points. Positioned between the IoT and cloud layers, it provides computing, networking, and storage capabilities closer to the data source, thereby reducing latency and improving responsiveness[33]. Finally, the cloud layer comprises powerful cloud servers that offer large-scale data storage, intensive computation, and long-term analytics capabilities [34], [35], [36]

3. TASK SCHEDULING IN CLOUD-FOG ENVIRONMENTS

The integration of cloud and fog computing has led to the development of various methods aimed at efficiently managing these interconnected platforms. These methods help reduce the workload

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

on cloud servers by distributing computational tasks across both fog and cloud layers. In this architecture, network edge devices send their processing requests to nearby fog nodes or cloud servers. The fog layer first reviews incoming requests and decides whether to execute them locally or forward them to the cloud, based on task characteristics such as computational complexity, latency sensitivity, and resource requirements. For instance, within the Internet of Vehicles (IoV) ecosystem, task scheduling plays a crucial role due to the vast number of connected vehicles continuously generating large volumes of data [37], [38]. Implementing efficient scheduling techniques in such environments minimizes latency, ensures timely task processing, and ultimately enhances vehicle safety and traffic management. Task scheduling in fog computing presents a complex challenge due to the dynamic and heterogeneous nature of the fog environment. This environment comprises diverse computational resources, fluctuating workloads, and mobile devices, all of which influence the efficiency and stability of task execution. As a result, developing efficient task scheduling and resource management techniques is essential to ensure optimal performance within the fog-cloud continuum [39]. Researchers have proposed several approaches to address this challenge, including heuristic-based**,** optimization-based**,** and machine learning-based methods. Heuristic-based methods rely on predefined rules or heuristics to assign tasks to fog nodes according to parameters such as proximity, node availability, and current workload [40]. Although these methods are relatively simple and computationally efficient, they often fail to achieve globally optimal results due to their reliance on local decisions. To overcome these limitations, many researchers have proposed improved heuristic algorithms that enhance scheduling performance by integrating adaptive decision mechanisms or combining heuristic strategies with optimization concepts[29].In some approaches [41], [32], [42], [43], [44], [45], IoT nodes are clustered at the network edge to process data closer to the source. This clustering strategy helps reduce communication delays, improves resource utilization, and enhances overall system responsiveness in fog-based task scheduling

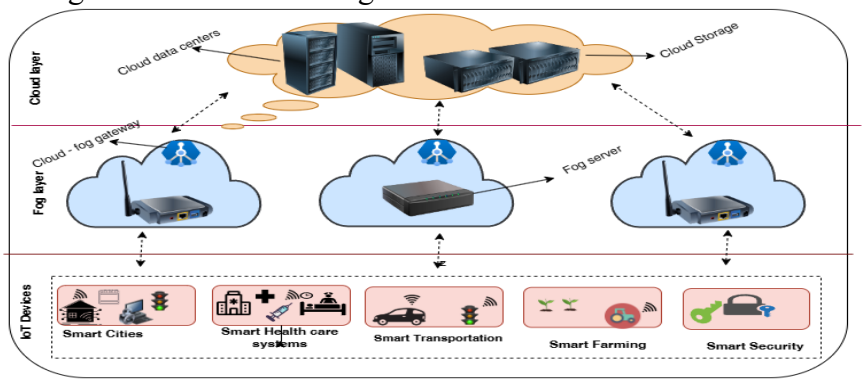

Figure 1: The Three-layer Fog – Cloud Computing Architecture.

Table 3 presents a comparative summary of recent review on task scheduling techniques**.** Various researchers have investigated the characteristics of different task scheduling methods within fog–cloud environments. In table1. The Author [46], This study reviewed a wide range of task scheduling algorithms in fog–cloud computing. The existing scheduling techniques were categorized into four major groups: learning-based, heuristic, metaheuristic, and deterministic approaches. Additionally, key qualitative performance parameters including response time, cost, and energy consumption were examined for each method. Finally, the review highlights existing

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

research gaps and outlines potential future research directions. The author [47], This study reviewed a broad spectrum of task scheduling algorithms in fog–cloud computing, with a particular focus on those developed using metaheuristic approaches. Key performance metrics including response time, cost, and energy consumption were analyzed to assess the effectiveness of each method. The findings also reveal several research gaps, which provide valuable insight and direction for future advancements in scheduling techniques within fog–cloud environments. The author [48], Several studies have reviewed various task scheduling algorithms in fog–cloud computing environments. Existing techniques in this domain are broadly classified into three groups: heuristic approaches, metaheuristic approaches, and hybrid methods. These techniques are evaluated based on qualitative performance metrics such as response time, execution cost, and energy consumption. In addition, the literature highlights several open research challenges and proposes potential future directions to guide further advancements in fog–cloud task scheduling. The author [13], A number of studies have reviewed task scheduling algorithms within edge–cloud computing environments. The existing techniques in this field have been primarily examined based on metaheuristic approaches. These approaches are evaluated using key qualitative performance metrics, including response time, execution cost, and energy consumption. In addition, the reviewed literature identifies several open issues and proposes promising directions for future research in fog–cloud task scheduling. The author[49], presented a systematic review of task scheduling algorithms in fog environments. The authors examined both dynamic, static, heuristic and hybrid scheduling approaches and discussed the strengths and weaknesses of the reviewed algorithms. In addition, the study outlined various task scheduling tools applicable to cloud–fog computing. The survey also identified several open challenges related to task scheduling and proposed potential future research directions to address these limitations. The author [50], The authors presented a survey article on fog-based task scheduling covering the period from 2015 to 2018.

The task scheduling techniques were categorized into two main groups: dynamic and static approaches. However, the review lacked a systematic structure, and the article selection methodology was not clearly outlined. The author [51], the study presented a systematic review of task scheduling algorithms in cloud–fog environments. It analyzed both dynamic and static scheduling approaches and highlighted the strengths and limitations of each method. In addition, the review identified various task scheduling tools relevant to cloud–fog computing. The study also outlined existing research challenges and proposed future research directions to address unresolved issues in task scheduling. The author [40], The study reviewed various task scheduling algorithms in fog–cloud computing. The existing techniques were examined based on heuristic and metaheuristic approaches. Key qualitative performance parameters, such as response time, cost, and energy consumption, were evaluated for each method. Additionally, the review identified open research challenges and proposed potential directions for future work. The author[52] The study reviewed various task scheduling algorithms in fog–cloud computing, categorizing the existing techniques into static and dynamic approaches. Key qualitative performance parameters, including response time, cost, and energy consumption, were assessed for each method. In addition, the review highlighted significant open research challenges and proposed potential directions for future work to advance scheduling efficiency within fog–cloud environments.

Based on the studies presented in Table 1, the existing surveys on cloud–fog task scheduling exhibit several limitations, including:

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

(a). All the studies fail to include the most recent scheduling methods, especially those from 2024 and 2025
(b). Certain surveys failed to evaluate the performance metrics aspects of the techniques
(c). The process for selecting articles is not well-defined, and certain studies do not present the surveys in an organized manner

As far as we know, this survey is the first of its kind that provides a complete Systematic Literature Review (SLR) on the current task scheduling techniques in fog computing and their comparison in terms of relevant parameters. The main purpose of this study is to review the task scheduling algorithms presented in different articles, categorize them, and analyze their benefits and drawbacks. Four categories are presented to study these techniques, including machine learning, heuristic-based, metaheuristic-based, and determin- istic mechanisms. Furthermore, several scheduling criteria are employed to evaluate the presented techniques, including energy consumption, makespan, execution time, delay, response time, cost, resource utilization, and throughput

4. AN OVERVIEW OF MACHINE LEARNING APPROACHES

Conventional scheduling methods are not efficient when dealing with large-scale data sets[53]. Machine learning-based approaches can be applied in fog task scheduling to enhance decision accuracy and overall performance. By learning from historical workloads and recognizing behavioral patterns, machine learning models can forecast future traffic conditions and allocate tasks more intelligently. This capability helps minimize latency, enhance system responsiveness, and improve resource utilization[54]. Furthermore, machine learning can detect abnormal activities or potential failures and initiate preventive actions before they affect the system. In summary, integrating machine learning techniques into fog task scheduling can significantly improve scheduling efficiency and boost the performance of fog computing environments.

In this section, the machine learning-based techniques are classified into three groups. Additionally, their respective strengths and limitations are examined and discussed

4.1. Deep Reinforcement Learning (DRL) Based Methods

Deep Reinforcement Learning (DRL)-based approaches show strong potential for enhancing task scheduling in fog computing systems by lowering energy consumption and improving overall performance. These techniques continuously learn from previous interactions and leverage real-time information to make optimal decisions, which helps minimize latency. However, a key drawback is the high implementation complexity and computational overhead, as DRL models often require substantial processing power and resources. For example [55] uses Deep Reinforcement Learning-based scheduling for optimizing system load and response time in edge and fog computing environments. The authors employed DRLIS to dynamically and efficiently optimize the response time of diverse IoT applications while also ensuring effective load balancing across edge and fog servers. The approach significantly reduces the execution cost of IoT applications by up to 55%, 37%, and 50% in terms of load balancing, response time, and weighted cost, respectively. However, as the system scale increases, the model fails to achieve convergence, resulting in unstable outcomes.

The author [56] uses Deep Reinforcement Learning Approach for Energy Efficient Task Scheduling in Fog Environment which focuses on scheduling IoT tasks in a fog computing environment with the goal of minimizing energy consumption, service delay, and operational cost.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

The approach effectively optimizes time and energy usage, even under strict resource limitations and deadline requirements. However the radeoff between energy and delay need to take into account The author [57] present Deep Reinforcement Learning approach for saving time and cost on scheduling of fog- based IoT application. The proposed approach employs a Double Deep Q-Learning (DDQL)–based scheduling mechanism that integrates target networks and experience replay strategies to minimize service delay, cost, and energy consumption. The experimental results demonstrate that the method not only improves energy efficiency but also achieves an effective balance of workload across virtual machines. However the Average waiting time need to be considered. The author [58], proposes a two-phase scalable scheduling algorithm based on deep learning models to optimize energy consumption in IoT environments. In the first phase, a clustering method determines the execution location of each task, followed by scheduling based on the identified location. Three clustering methods are explored: Self-Organizing Map (SOM), Hierarchical SOM (H-SOM), and Autoencoder-based dimensionality reduction. In the first two methods, tasks from the IoT layer are clustered according to their attributes using SOM and H-SOM, while in the third method, an Autoencoder reduces attribute dimensions before clustering. Simulation results indicate that the Autoencoder-based approach reduces the number of missed tasks compared to SOM and H-SOM, demonstrating improved energy efficiency. The proposed method is scalable, capable of handling large numbers of IoT devices and fog nodes. However, it may require substantial computational resources and training time, which could affect the overall scheduling execution time and may not always satisfy the execution deadlines of certain IoT applications, potentially impacting performance and reliability. The author [53] The study proposes an efficient algorithm for rescheduling preempted tasks in fog nodes, named Brain-Inspired Rescheduling Decision-making (BIRD)**.** BIRD leverages an Actor-Critic reinforcement learning model to emulate human brain decision-making, aiming to generate a rescheduling list that meets task deadlines while ensuring optimal fog node performance through load balancing. The algorithm was evaluated against other scheduling policies, including First Come First Served (FCFS), Greedy Task Allocation, Least Laxity, Shortest Job First (SJF), and Earliest Deadline First (EDF). Results indicate that BIRD effectively maintains deadlines for preempted tasks, reduces delays caused by task preemption, and optimizes resource utilization through balanced task distribution. Experimental evaluation also demonstrates that BIRD achieves faster execution times compared to the benchmark policies. However, the study focuses exclusively on rescheduling preempted tasks in fog computing, which may limit its applicability to other areas of computing or technology. The author [59], The study employs a Deep Reinforcement Learning (DRL) based model for fog task scheduling in mobile crowd sensing applications. Specifically, the authors propose an efficient task scheduling algorithm using a Double Deep Q-Learning (DDQL) framework, aimed at reducing service latency and measurement costs while respecting resource and time constraints. Results indicate that the proposed model outperforms simpler scheduling techniques in terms of task completion time and delay. Additionally, the algorithm addresses challenges such as single-point-of-failure issues and fog load balancing. However, it exhibits slow convergence and uncertainty when handling high-dimensional state-action spaces. Moreover, the study does not provide detailed insights into the algorithm's performance regarding network bandwidth and energy consumption

The author [60], present a learning automaton–based algorithm for dynamic, fault-tolerant task scheduling aimed at optimizing response time while ensuring reliable task execution. The approach employs a state transition model to adaptively adjust task scheduling according to the current

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

system state and resource availability. It also integrates fault-tolerance mechanisms to manage failures in the network or devices involved in task execution. This method enhances reliability and system robustness, while the dynamic adjustment of task allocation can reduce delays and improve overall performance. However, maintaining the state transition model and performing continuous dynamic adjustments may require substantial computational resources, and the complexity of the approach could pose challenges for practical implementation and maintenance

Table 4: Machine Learning Scheduling Approaches

Author/ Year	Algorithms	Objectives	Tools	Strength	Limitation
[55]	DRLIS	response time and load balancing	MATALAB	Reduces the execution cost of IoT applications up to 55%, 37%, and 50% in terms of load balancing,	fails to achieve convergence,.
[57]	DDQL	Minimize the service delay, cost and energy	Python	Achieved efficient energy consumption and balance the task b/w VM	waiting time need to be consider
[58]	Deep Learning	Minimize the cost	Python	Reduce the missed rate and cost in fog and cloud considering dependent task	It may require substantial computational resources
[53]	Re-enforcement Learning	CPU Utilization	Python	The approach uses load balancing mechanism to to balance the load b/w fog and cloud	Time complexities
[59]	Reinforcement Learning	Completion Time	Python	The results indicate that the approach outperforms current scheduling methods used in mobile crowd sensing systems	Need to consider network bandwidth, and energy
[56]	DRL(CDDQL)	Minimize energy, cost, and service delay	Python	Effectively optimizes time and energy usage, even under strict resource limitations and deadline requirements.	Tradeoff between energy and delay
[60]	DFTLA	Minimize response time and energy consumption	MATLAB R2017a	The approach achieves higher reliability and robustness than the competing methods	Issue of maintaining the state transition model
[61]	ANN/ABC	Minimize response time and energy consumption	MATLAB 2016a	The approach uses overflow handling to reduce the model response time while using less energy	Model complexity and integrating energy efficiency
[54]	FEDRATED LEARNIG	minimize the completion time	Python	Results demonstrate that the algorithm converges effectively under multiple task-size conditions.	the privacy and security issues need to be consider

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

[62]	Game theory	Minimize execution time and makespan	Python	The method offers enhanced scalability while maintaining low execution time,	VM utilization

4.1.2 Artificial Neural Network (ANN) Based Methods for Task Scheduling

Artificial Neural Networks (ANNs) are computational models inspired by the structure and functioning of the human brain. They consist of interconnected nodes (neurons) arranged in layers (input, hidden, and output layers), which process information by adjusting the weights of connections based on input-output data. In fog–cloud computing, ANNs are increasingly used for task scheduling because they can learn complex patterns and make intelligent predictions or decisions in dynamic and heterogeneous environments. For instance[61], develops an optimized Task Group Aggregation (TGA) overflow handling system for fog computing environments using neural computing. The authors propose an ANN-based algorithm to detect overloaded servers and transfer task data to virtual machines (VMs) in fog nodes. Key resource factors such as CPU, memory, and bandwidth are considered to balance the workload across VMs. Additionally, the Artificial Bee Colony (ABC) algorithm is employed to separate services and users based on their specific quality requirements. Experimental results demonstrate that the ANN-based overflow handling algorithm improves response time and success rate compared to existing approaches. However, the study does not consider energy consumption. Furthermore, given the computational complexity of ANN algorithms, future work could focus on optimizing model complexity and integrating energy efficiency into the scheduling process

4.1.3 Federated Learning Based Methods for Task

Federated Learning (FL) is a distributed machine learning approach that enables multiple parties to collaboratively train a model without sharing their raw data. By leveraging decentralized data sources, such as edge or IoT devices, FL allows model training while preserving data privacy and security. The main advantage of this approach is its ability to utilize distributed computing resources efficiently while protecting sensitive information. However, federated learning requires careful design of communication protocols and aggregation methods to ensure model convergence, accuracy, and scalability.

The author [61] The study proposes a federated learning–based multi-task scheduling mechanism for edge computing that incorporates trusted computing sandbox technology to enhance both efficiency and security. The system consists of a task scheduling module, which assigns tasks to edge nodes based on their workload and capabilities; a federated learning module, enabling collaborative training of machine learning models on distributed data while maintaining privacy; and a trusted computing sandbox module, which provides a secure execution environment for tasks. This integrated approach improves scheduling efficiency, reduces communication overhead, and strengthens system security. However, its implementation requires specialized hardware and software support for trusted computing, which may limit widespread adoption. Overall, the proposed mechanism demonstrates strong potential for improving task scheduling performance and security in edge computing environments.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

4.2 An Overview of Heuristic Based Approaches

Heuristic-based mechanisms for fog task scheduling rely on predefined rules or strategies to assign tasks across fog devices. Unlike machine learning approaches, these methods depend on experience-driven decision rules rather than adaptive learning. While they are relatively simple and straightforward to implement, they may not always provide optimal scheduling outcomes[63]. These methods also lack adaptability, as they cannot adjust to dynamic environments or learn from historical behavior in the way machine learning-based approaches can. In the following section, several heuristic-based scheduling strategies for fog environments are analyzed, focusing on the parameters they employ to achieve more effective and efficient scheduling decisions.

execution duration The author [64] presented a Deadline-aware and energy-efficient IoT task scheduling in fog computing systems using a priority-aware semi-greedy approach to Minimize total energy consumption, deadline violation time and makespan. The proposed method minimizes overall system energy consumption without violating task deadline constraints. However there is the need to consider the dependent IoT tasks while scheduling tasks to fog or cloud layers. The author [65], proposes resource allocation and task offloading for heterogeneous real-time tasks with uncertain duration time in a Fog Queueing System, to address task heterogeneity, a parallel virtual queue model is utilized, where each task type is stored in a separate queue. A framework consisting of three parallel algorithms offloading, buffering, and resource allocation is then applied to enhance load balancing, throughput, and task completion rates. Task offloading decisions are based on urgency using laxity time, which considers the deadline, estimated, and cloud transmission delay, proposed approach enhances throughput and task completion efficiency, while simultaneously preventing task starvation and long latency. However, VM underutilization need to be considers to avoid allocating task to inappropriate virtual machines.

The author [66], presented an energy-efficient and delay-guaranteed workload allocation in iot-edge-cloud computing systems. The proposed approach defines a delay-oriented workload allocation problem aimed at identifying the optimal distribution of tasks across the local edge server, neighboring edge servers, and the cloud, while ensuring minimized energy consumption and meeting delay constraints. To solve this problem, a Delay-Based Workload Allocation (DBWA) algorithm grounded in the Lyapunov drift-plus-penalty framework is employed. The results demonstrate that the proposed system ensures energy efficiency and reliable delay performance in IoT–edge–cloud environments. However Virtual machine placement Need to consider for optimal performance. The author [2], proposed an Energy and Cost-Aware Real-Time Task Scheduling approach with deadline constraints in fog computing, aiming to minimize both energy usage and monetary cost through the ECaTSD method. The experimental results demonstrate that the ECaTSD algorithm achieves strong performance in deadline compliance, with a completion rate of 99.58%, while also offering better energy and cost efficiency compared to other scheduling strategies. However, a limitation of this method is that, although it improves energy efficiency, it results in slower learning and increased convergence time.

The author [67], presented Privacy-preserving multi-objective task scheduling for IoT systems in cloud-fog environments using a goal-programming approach. In this approach a privacy-focused architecture is introduced for IoT task scheduling, and based on this design, a multi-objective algorithm is developed to reduce both service time and cost. The proposed method is evaluated through simulations across four levels of complexity: easy, medium, semi-hard, and hard, and its performance is compared with other multi-objective algorithms. Because the algorithm addresses multiple objectives, the Goal Programming Approach (GPA) is applied to select an optimal

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

solution. The approach significantly lowers service delay and execution cost while meeting deadline constraints and safeguarding IoT device privacy. However Load instability not consider in this research. According to [68], presented An adaptive scheduling approach for fog computing tasks with varying priority levels in intelligent production environments. The proposed method introduces a joint evaluation model that considers both time latency and power consumption, guided by task priority. Based on this model, a Fog Computing Adaptive Scheduling (FCAS) algorithm using dynamic programming is designed to determine the optimal processing path for tasks with varying priorities across fog nodes. Experimental results show that the proposed model and scheduling algorithm achieve reduced power consumption, improved efficiency, and enhanced reliability compared to existing approaches. The experimental findings indicate that using fog-assisted processing reduces latency by 2.7% to 27.9% and enhances power efficiency by 22.2% to 67.2%. However instability of data transmission was not captured.

The author [18], presented A Cost- and Energy-Efficient Task Scheduling method for a cloud–fog framework was proposed, aiming to minimize makespan, energy consumption, and overall cost. The results demonstrate significant improvement, with efficiency increased by 89%, energy consumption reduced by 94%, and total cost decreased by 87% compared to existing algorithms. However, a limitation of this approach is its inability to predict future workloads or determine when task offloading is necessary. The author [69] presented An improved List-Based Task Scheduling Algorithm for Fog Computing Environments, This algorithm schedules tasks represented as a Directed Acyclic Graph (DAG). Its primary goal is to assign tasks to appropriate processing nodes within the fog environment, considering the limited computational capacity of fog nodes. Task allocation takes into account both the computation cost of each node and the expected task completion time. The decision between local optima and the global optimal solution is effectively balanced, further minimizing the makespan and reducing the overall computation cost of the processors. However, the approach can be extended for a dynamic network.

Table 5: Heuristics-Based Scheduling Approaches

Author/ Year	Algorithms	Objectives	Tools	Strength	Limitation
[64]	priority-aware semi-greedy (PSG)	Minimize total energy consumption, deadline violation time and makespan	C++ programming language on Dev-Cpp 5.11 IDE.	The proposed method minimizes overall system energy consumption without violating task deadline constraints	Need to consider the dependent of tasks
[65]	Queuing systems	Execution time and transmission delay	CloudSim toolkit	Proposed approach enhances throughput and task completion efficiency,	VM Underutilization
[66]	Lyapunov drift-plus-penalty theory.	Minimize energy consumption and delay	MATLAB	Results demonstrate that the system ensures energy efficiency and reliable delay	Need to consider VM
[2]	ECaTSD	Energy consumption and monetary costs		The method shows high efficiency in meeting deadlines 99.58% completion rate	inefficient learning and longer convergence

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

[67]	(GPA)	Minimize the service time and service cost	MATLAB	the approach significantly lowers service delay and execution cost while meeting deadline constraints and safeguarding	Load instability does not consider
[68]	(DPA)	Minimize energy and latency	iCandyBox	results show that latency decreases by **2.7%–27.9%**, and power consumption	instability of data transmission
[18]	EEOA	Makespan time, energy consumption and cost		The approach improves efficiency by 89%, energy consumption by 94	cannot predict upcoming workloads .
[69]	(LBSA)	Minimize makespan and cost	iFogSim	The decision between local optima and the global optimal solution is effectively balanced,	To extended for a dynamic network.
[70]	(DATS)	Minimize service delay	Cloudsim	algorithm achieved anc effective trade-off between computing resources and communication capabilities	Need to take the energy consumption
[71]	Task (TCVC)	Minimize execution time and waiting time	Cloudsim	The approach improve the performance of health care IoT efficiency by achieving better AWT and AET	Energy c need to take into account.
[72]	Nonlinear Programming	energy consumption with a guaranteed delay	Cloudsim	Achieved ~22% energy reduction and ~12.5% delay reduction vs FCFS	heterogeneous fog nodes not deeply explored).
[73]	Priority-based task scheduling algorithms	Minimize transmission and overall cost	CloudSim	The proposed prioritized scheduling approach significantly reduces response time and cost while improving overall system efficiency	may focus on enhancing the scheduling algorithm by introducing dynamic priority
[74]	(RMS)	Minimize migration time delay and execution time	Mobfogsim	outperforms the existing DMS technique in migration time,.	High complexity
[27]	ODSP	processing delay		Simulation results show that over 15% per- formance gain, in the system adopted optimal data scheduling policy	Not scalable for larger network
[75]	(MaSAP)	Minimize Makespan and Energy consumption	iFogsim	*The results indicate that the Makespan remained below the required MaxResponse threshold for all generated tasks.*	*To explore a distributed scheduling version.*
[76]	Goal Programming	Minimize service delay and cost	MATLAB	*The solution provides an optimal compromise between time and cost, with the lowest*	Not scalable for larger network

https://doi.org/10.5281/zenodo.18407391

	Approach (GPA)			*rate of violations related to access control and task deadlines*	
[63]	PQFAHP	The completion time, energy consumption	Cloudsim	Around 11% being offloaded to cloud resources.	needed to allow several equal VMs to run simultaneously.
[77]	GA	To minimize the deadline misses of	Python	Results reveal a 20%–55% improvement in deadline adherence over other methods	Computational complexity

According to [70], presented a Dispersive Stable Task Scheduling in Heterogeneous Fog Network, The goal is to efficiently allocate complex computational tasks across multiple neighboring fog nodes to minimize overall service delay using dispersive stable task scheduling (DATS). The proposed DATS algorithm achieved an effective trade-off between computing resources and communication capabilities. However, the approach should incorporate energy efficiency considerations. The author [71] proposed Fog Computing-Based Scheduling of IoT Healthcare Tasks According to Task Importance, to improve IoT applications in healthcare, including mobile health and remote patient monitoring, for various conditions such as heart disease, hypertension, diabetes, and other chronic illnesses. This is achieved using a novel approach called Task Classification and Virtual Machine Categorization (TCVC). This makes the proposed scheduling method well-suited for real-time remote healthcare monitoring. Simulation results indicate that it achieves an optimal balance among Average Waiting Time (AWT), Average Execution Time (AET), and Average Finish Time (AFT) compared to other scheduling algorithms. However energy consumptions need to take into account.

According to [72] an Energy Consumption Optimization approach with a delay threshold in cloud–fog cooperative computing was proposed to minimize energy usage while ensuring delay constraints, using nonlinear programming. The method achieved approximately 22% reduction in energy consumption and around 12.5% reduction in delay compared to FCFS. However, the approach does not thoroughly investigate the use of average values for energy thresholds or the heterogeneity of fog nodes. The author [73] presented a Prioritized Task Scheduling in Fog Computing to accommodate the growing number of IoT and smart devices while enhancing performance and reducing costs, this study proposes a priority-based task scheduling algorithm at the fog layer. The paper provides a detailed description of the proposed architecture, including the queuing and priority models, the priority assignment module, and the priority-based task scheduling algorithms. The evaluation results demonstrate that the proposed algorithm outperforms existing scheduling methods by decreasing both response time and total cost. However Future work may focus on enhancing the scheduling algorithm by introducing dynamic priority adjustments based on real-time request traffic load.

4. 3 Metaheuristic-Based Scheduling Approaches

Meta-heuristic algorithms are widely recognized for their high computational efficiency**,** powerful search capability**,** and effective optimization performance, making them valuable tools for addressing a broad range of optimization problems. In recent years, these algorithms have been successfully applied to solve complex, real-world challenges across diverse fields, demonstrating their robustness and adaptability in handling nonlinear, multidimensional, and dynamic

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

optimization tasks[78]. Metaheuristic algorithms have become indispensable for optimizing task scheduling in edge and fog–cloud computing environments. Their ability to efficiently explore vast and complex search spaces allows them to identify near-optimal solutions even under dynamic and uncertain conditions. These algorithms are particularly effective in addressing resource management challenges, such as balancing computational load, minimizing latency, reducing energy consumption, and improving overall system throughput[14]. By mimicking natural, biological, or physical processes such as evolution, swarm intelligence, and adaptive learning metaheuristic approaches provide flexibility and robustness in solving multi-objective optimization problems where traditional deterministic or heuristic methods often fail. Consequently, they have become a core component of modern task scheduling frameworks designed for intelligent, adaptive, and efficient fog computing systems[79][80]. For example, the Hybrid Meta-Heuristic Optimization Algorithm**,** which integrates the strengths of Genetic Algorithms (GA) and Particle Swarm Optimization (PSO)**,** has shown remarkable improvements in within fog–cloud environments[81]. By combining GA's strong global exploration capability with PSO's rapid convergence and local search efficiency, this hybrid task scheduling efficiency approach achieves superior energy efficiency**,** optimized resource utilization**,** and reduced response times[82]**.** The algorithm dynamically allocates user task requests between fog nodes and cloud servers based on factors such as computational capacity, network latency, and energy constraints. As a result, it enhances load balancing**,** minimizes makespan**,** and ensures high-quality service delivery[83]**.** This hybridization demonstrates how integrating multiple metaheuristic strategies can effectively overcome the limitations of individual algorithms, leading to more robust and adaptive task scheduling solutions in distributed computing environments.

Similarly, nature-inspired algorithms such as Genetic Algorithm (GA)**,** Particle Swarm Optimization (PSO)**,** and Ant Colony Optimization (ACO) have been extensively evaluated in edge computing environments for their effectiveness in optimizing task scheduling and resource allocation. Among these, GA has shown superior performance in minimizing energy consumption through efficient task distribution and adaptive selection mechanisms, while PSO excels in reducing average flow time due to its fast convergence and strong local search capabilities[84]. On the other hand, ACO demonstrates competitive performance in load balancing and path optimization**,** particularly in network-aware scheduling scenarios. The comparative results highlight that each algorithm offers unique advantages depending on the optimization objective GA for energy efficiency, PSO for response time minimization, and ACO for communication-aware optimization making them valuable tools for developing intelligent and adaptive scheduling frameworks in fog–cloud computing systems[84] Recent innovations have introduced advanced meta-heuristic algorithms that further enhance the efficiency of task scheduling in edge computing environments. One such notable development is the Quantum-Behaved Particle Swarm Optimization (QPSO**)** algorithm, which integrates the principles of quantum mechanics with traditional PSO to improve its global search capability and convergence behavior[85]. QPSO has demonstrated remarkable performance in reducing energy consumption, execution cost, and latency by effectively balancing exploration and exploitation during task allocation. By dynamically adjusting the positions of particles within a quantum space, QPSO ensures a broader search of the solution domain, thereby avoiding premature convergence a common limitation in conventional PSO[85]. Through this enhanced adaptability and precision, QPSO achieves efficient mapping of processing elements to computational tasks, resulting in optimized resource utilization and improved overall system performance in fog–cloud and edge computing infrastructures. These

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

findings underscore the adaptability and effectiveness of meta-heuristic algorithms in optimizing task scheduling within fog–cloud computing environments**.** By intelligently balancing exploration and exploitation, these algorithms significantly enhance system performance, energy efficiency, and resource utilization**.** Their dynamic and flexible nature allows them to adapt to the heterogeneity and unpredictability of fog–cloud systems, ensuring improved load balancing, reduced latency, and optimized resource management across distributed networks.

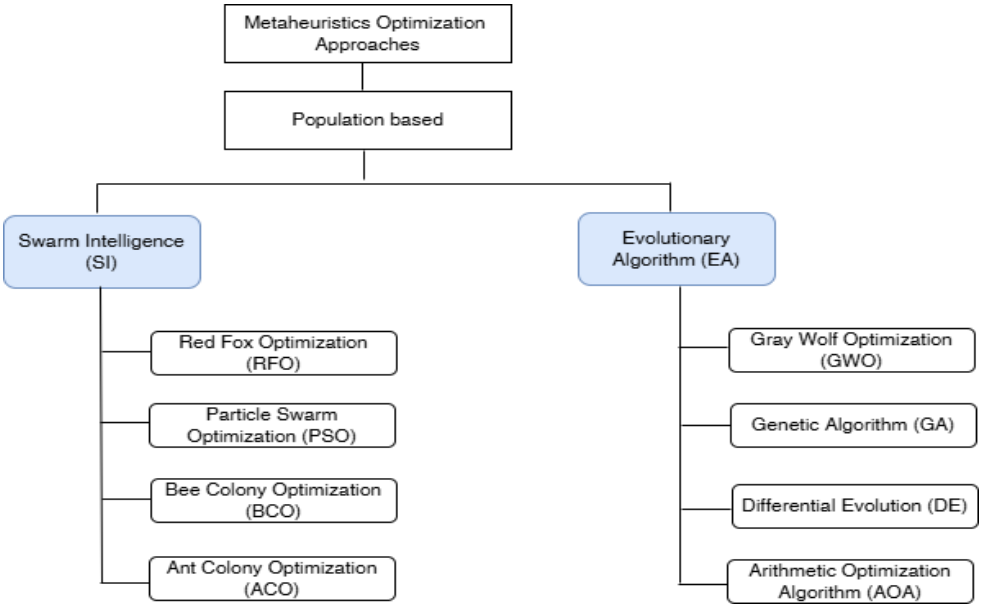

Figure 2: Summary of Metaheuristics Approaches

4.3.1 Population-Based

Population-based algorithms have emerged as a powerful solution for addressing the challenges associated with optimizing large and complex search spaces. These algorithms operate using a group (population) of candidate solutions that evolve iteratively to find near-optimal results. Commonly referred to as nature-inspired algorithms, they draw their inspiration from various natural phenomena and biological processes, such as evolution, swarm behavior, and ecological systems. By mimicking these natural mechanisms, population-based algorithms effectively balance exploration and exploitation, making them well-suited for solving nonlinear, multidimensional, and dynamic optimization problems[86] These algorithms function by exploring the search space through a diverse population of potential solutions**,** which interact, compete, and evolve toward optimal outcomes. Each algorithm employs unique strategies inspired by natural behaviors to balance exploration (searching new areas of the solution space) and exploitation (refining promising solutions). Among the most well-known population-based algorithms are: Genetic Algorithm (GA): inspired by the process of natural selection and genetics, Firefly Algorithm (FFA): based on the flashing behavior of fireflies used for attraction and communication, Gray Wolf Optimizer (GWO): modeled on the leadership hierarchy and hunting tactics of gray wolves, Ant Colony Optimization (ACO): simulates the foraging behavior of ants

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

using pheromone trails to find optimal paths, Particle Swarm Optimization (PSO): inspired by the social behavior of bird flocks and fish schools. These algorithms have been widely adopted due to their robustness, flexibility, and efficiency in handling complex optimization problems across diverse domains such as cloud computing, engineering design, and data analytics[87]. Population-based algorithms are generally categorized into two major classes based on their inspiration and operational mechanisms namely Evolutionary Algorithm (EA) and Swarm Intelligence (SI)[88].

4.3.2 Swarm Intelligence (SI)

Swarm Intelligence (SI) is inspired by the collective and self-organized behavior of natural systems, particularly those exhibited by social species such as birds, ants, bees, and fish. These algorithms emulate the socio-cooperative mechanisms that enable groups of simple agents to collectively solve complex problems without centralized control[89]. Typically, SI-based algorithms draw inspiration from the foraging, hunting, and communication behaviors of these species to design decentralized and adaptive optimization strategies. Through cooperation and information exchange among individuals (or agents), SI algorithms can efficiently explore and exploit large and complex search spaces. Common examples of Swarm Intelligence algorithms include: Red Fox Optimization (RFO): Inspired by the intelligent hunting strategies and adaptive movement of red foxes, Particle Swarm Optimization (PSO): Modeled after the coordinated motion and information sharing in bird flocks or fish schools, Ant Colony Optimization (ACO): Based on the pheromone-trail-following behavior of ants during food foraging, Artificial Bee Colony (ABC): Derived from the foraging and communication patterns of honeybees searching for nectar sources[89]. These algorithms have been widely applied across diverse optimization domains due to their robustness, scalability, and adaptability in handling nonlinear, multidimensional, and dynamic problems

4.3.3 Evolutionary Algorithms (EA)

An Evolutionary Algorithm (EA) is a class of population-based stochastic optimization techniques inspired by the processes of natural evolution and biological adaptation [85]. These algorithms mimic the way species evolve over generations, guided by mechanisms such as selection, mutation, crossover (recombination), and survival of the fittest. The primary goal of EAs is to iteratively improve a population of candidate solutions through these evolutionary operators, enabling the algorithm to discover near-optimal or optimal solutions to complex optimization problems without requiring gradient information. Common examples of Evolutionary Algorithms include: Arithmetic Optimization Algorithm (AOA): Inspired by arithmetic operators, it uses mathematical principles to balance exploration and exploitation, Genetic Algorithm (GA): Based on the process of natural selection and genetic inheritance, Differential Evolution (DE): Relies on mutation and recombination operations to explore and refine solutions, Gray Wolf Optimizer (GWO): Modeled after the leadership hierarchy and cooperative hunting behavior of gray wolves, Firefly Algorithm (FA): Inspired by the flashing behavior of fireflies, where brightness guides the search process. These algorithms are highly effective for solving complex, nonlinear, and multi-objective optimization problems, especially in environments like fog–cloud computing, where adaptability and efficiency are essential. Figure **5** provides a summarized overview of these meta-heuristic approaches

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

Table 6: Metaheuristics Scheduling Approaches

Author/ Year	Algorithms	Objectives	Tools	Strength	Limitation
[90]	RFOAOA	energy and makespan time	MATLAB	addressed the issues of load instability, slow convergence rate and	Scalability issue
[91]	MOGWO	Delay and energy consumption	MATLAB	The simulation result verifies the effectiveness of the proposed algorithm	High-dimensional multi-objective problems.
[92]	PSO	Response and processing time.		The result shows that PSO outperform round robin, Active Vm,.	Computation complexity
[39]	MPSO/MCSO	Minimize energy, execution time	iFogSim	*The results indicate that NBIHA performs effectively with reduced energy*	Computation complexity
[41]	(IWO-CA)	Minimize energy consumption	Cloudsim	proposed algorithm outperforms existing methods	Scalability issue
[93]	GA	Makespan time, energy and	Python	reduced the reduced the26%, decreasing the energy by 32.4%, decreasing total.	Computation complexity
[94]	GA	completion time, energy consumption, and cost.	Python	The result show that LCO is optimal for latency-sensitive applications	not suitable when dealing with competitive environment
[95]	MS-PSO	Makespan, cost, energy		MS-PSO outperforms the canonical PSO	Lacks adaptability
[96]	IGA-POP	Ttrade-off between various performance		The proposed algorithm can achieve a better balance between the makespan and cost art algorithms.	failure rate in some scenarios,
[97]	CHMPAD	Makespan time and throughput		The approach shows makespan time improvements of 1.12–43.20% (for synthetic workloads), .	consider task Defendancy
[98]	GA	Makespan time, energy consumption and degree of imbalance		The approach reduced the makespan by 26%, decreasing the energy consumption by 32.4%,.	The approach can converge prematurely
[99]	GA-PSO	**Makespan** and **energy consumption**		Outperforms GA and PSO individually in simulations -.	Does not consider **dynamic task arrivals**
[100]	GA	minimize overall service request latency		The results show that the overall latency for the proposed approach is 21.9% to 46.6% better than the other algorithms.	Entrapment at local search

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

[101]	CMS/ACO	Minimize energy consumption	Python	*Addressed multi-task scheduling*	*should extend the scheduling algorithm to support online task scheduling.*
[102]	(PMTSF)	Minimize delay and energy consumption	Cloudsim	*This approach enables optimal task offloading from fog devices to cloud infrastructure*	Can converge pre manually to local optima
[103]	(GA/SA)	Minimize makespan time and speedup	Python	*Demonstrates superior performance, achieving 10.17%, 9.31%, 7.76%, and 8.21%.*	Computation complexity
[104]	(IEGA)	Minimize makespane and energy	Java	*IEGA demonstrates superior performance compared to all benchmarks.*	Scalability issue
[12]	(OppoCWOA)	Minimize time and energy consumption	Cloudsim	algorithm achieves notable improvements in time-energy	Computation complexity
[7]	EGWO	Minimize makespan and energy consumption	Python 3.12	The results revealed that the EGWO consistently achieves a lower makespan	Need to consider VMs migration
[105]	Evolutionary Algorithm (EA)	Minimize Execution time and cost	Python	proposed algorithm outperforms the Bee Life Algorithm (BLA) by 38.6% in execution time–cost	*Budget and deadline constraints.*
[106]	(HHOLS)	Minimize energy consumption,	iFogsim	The method outperforms benchmark algorithms in terms of energy	complex configuration
[107]	(MFO)	Minimize total execution time	iFogSim	The proposed algorithm effectively determines an optimal scheduling strategy,	Virtual Machine placement

This section offers a detailed review of state-of-the-art metaheuristic scheduling techniques within fog–cloud computing environments. The discussion focuses on emerging strategies developed to address the inherent complexity associated with the dynamic characteristics and constrained resources of fog infrastructure. Key issues examined include fluctuating task workloads, real-time variability in resource availability, and the need to achieve efficient task scheduling while maintaining minimal latency and energy consumption. Through the evaluation of recent contributions, this section outlines the underlying methodologies, advantages, and limitations of dynamic metaheuristic scheduling approaches, and demonstrates how these advancements contribute to improved resource utilization and enhanced overall performance of fog–cloud systems.

The study in [90] introduced a Hybrid Enhanced Optimization-Based Intelligent Task Scheduling framework for sustainable edge computing, aimed at mitigating load imbalance, slow convergence, and virtual machine underutilization, while accounting for both energy consumption and makespan. The proposed approach, termed RFOAOA, integrates Red Fox Optimization (RFO) with the Arithmetic Optimization Algorithm (AOA) to effectively address these challenges.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

Experimental results show that the method outperforms state-of-the-art techniques in reducing energy usage and makespan. However, a notable limitation of this work is the lack of consideration for task dependency and scalability, which are essential for efficient large-scale task scheduling. The author [91], proposed a Multi-Objective Grey Wolf Optimizer (MOGWO) for task scheduling in cloud–fog computing environments. The primary challenge addressed is the allocation of tasks to suitable resources based on task requirements. The study highlights an inherent trade-off: assigning tasks to fog nodes reduces latency but increases user-side energy consumption, whereas offloading tasks to the cloud lowers energy usage but may increase delay. Using the MOGWO approach, simulation results demonstrated improved performance compared to existing methods, particularly in reducing delay and energy consumption while effectively balancing the trade-off between the two objectives. However, a limitation of the method is its relatively slow convergence to the optimal Pareto front, especially when dealing with complex or high-dimensional multi-objective optimization problems.

The study presented in [92] proposed an efficient resource utilization framework for a cloud–fog environment integrated with smart grids, aimed at minimizing response time and processing time through the use of Particle Swarm Optimization (PSO). Experimental results indicate that the PSO-based approach outperforms Round Robin, Active VM, ACO, Throttled, and ODDS scheduling techniques with respect to both response and processing times. However, the method is limited by its computational complexity, which may affect scalability and real-time applicability. The author [41] presented a Hybrid Metaheuristic-Based Energy-Aware Task Scheduling in Fog Computing, In this work, an energy-aware scheduling approach is proposed that leverages Dynamic Voltage and Frequency Scaling (DVFS) to reduce energy consumption. To generate effective task execution sequences, a hybrid evolutionary algorithm combining Invasive Weed Optimization and Cultural Algorithm (IWO-CA) is employed. Experimental results demonstrate that the proposed method outperforms several existing algorithms, particularly with respect to energy efficiency. The author[93] introduced a task scheduling optimization approach for Cloud–Fog MCS environments using a Genetic Algorithm (GA) combined with game-theoretic principles, with the objective of minimizing makespan, energy consumption, and system imbalance. Experimental evaluation demonstrated notable improvements, including a 26% reduction in makespan, a 32.4% decrease in energy consumption, a 28% reduction in total system cost, and a 21.53% reduction in imbalance when compared to existing scheduling techniques. However, a limitation of the GA-based approach is its tendency toward premature convergence, which may result in suboptimal performance by becoming trapped in local optima rather than achieving the global best solution

According to [94] an efficient offloading and task scheduling mechanism was developed for Internet of Things (IoT) cloud–fog environments to enhance Quality of Service (QoS) by minimizing completion time, energy consumption, and operational cost. The approach employed a Genetic Algorithm (GA) combined with a weighted-sum objective function to balance multiple performance metrics. The results demonstrated that the Latency-Constrained Optimization (LCO) strategy is most suitable for latency-sensitive applications, the Energy-Based Optimization (EBO) approach performs best in scenarios where energy efficiency is a priority, while the Equilibrium Optimization (EO) method offers a balanced trade-off across all evaluated criteria. However, the study is limited by its evaluation scope, considering a maximum of only 100 tasks. This restriction reduces scalability and makes the approach less suitable for highly competitive or large-scale environments, where solution diversity and broader task complexity are required. In [95], a Multi-Swarm Particle Swarm Optimization (MS-PSO) algorithm was proposed for static workflow

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

scheduling in cloud–fog environments with the objective of minimizing makespan, cost, energy consumption, and improving load balancing across the cloud and fog tiers. Experimental results demonstrated that the proposed MS-PSO approach consistently outperformed the canonical PSO algorithm across a range of scientific workflows and performance metrics. However, despite these improvements, the method remains susceptible to premature convergence, where the search process may stagnate in local optima rather than progressing toward the global optimal solution.

The study in [96], proposed a Real-Time Task Scheduling Algorithm for IoT-based applications in cloud–fog environments, aiming to achieve an optimal balance between performance metrics such as makespan and total execution cost using the IGA-POP approach. While the algorithm demonstrates a lower failure rate in certain scenarios, it still exhibits a notable failure rate for instance, 45.3 ± 0.8% when executing 400 tasks under scenario 1 highlighting a limitation in its reliability under heavier workloads. The study in [97], proposed an Intelligent Chimp Optimizer (CHMPAD) for scheduling IoT application tasks in fog computing, with the objective of minimizing makespan and improving throughput. Experimental results indicate that CHMPAD achieves average makespan improvements of 1.12–43.20% for synthetic workloads, 1.00–43.43% for NASA iPSC workloads, and 2.75–42.53% for HPC2N workloads compared to other scheduling algorithms. However, a limitation of this approach is that task dependencies were not considered across all workload traces, which may affect scheduling efficiency in dependent-task scenarios. The study in [98], proposed a task scheduling approach for cloud–fog computing systems aimed at minimizing makespan, energy consumption, and system imbalance. The results demonstrate significant improvements, including a 26% reduction in makespan, a 32.4% decrease in energy consumption, a 28% reduction in total system cost, and a 21.53% decrease in imbalance compared to existing scheduling methods. However, a limitation of the approach is its tendency to converge prematurely to suboptimal solutions, potentially becoming trapped in local optima instead of achieving the global optimum. The study in [99], proposed a service allocation approach in fog computing environments using a hybrid bio-inspired algorithm to minimize makespan and energy consumption. The method integrates the exploration capabilities of Genetic Algorithm (GA) with the exploitation strengths of Particle Swarm Optimization (PSO), achieving better convergence than either GA or PSO alone. Simulation results demonstrate improved performance under transmission time and resource constraints. However, a limitation of this work is that it does not account for dynamic task arrivals or real-time execution scenarios. The study in [100], proposed a scheduling approach for IoT requests in hybrid fog–cloud computing, aimed at minimizing overall service request latency using a Genetic Algorithm. Results indicate that the proposed method achieves a 21.9% to 46.6% reduction in latency compared to other algorithms and improves the rate of meeting request deadlines by up to 31%. However, a limitation of this approach is its tendency to become trapped in local optima during the search process.

Table 7 presents a comprehensive comparative analysis of recent task scheduling approaches within fog–cloud computing environments. The comparison highlights whether each study addressed key performance evaluation metrics, including energy consumption, completion time, execution time, makespan, delay, cost, processing time, and throughput. This structured assessment enables a clear understanding of the optimization goals prioritized by existing research and reveals gaps that may inform future advancements.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

Table 7: Comparison of Quality-of-service Parameters for Task Scheduling Techniques

Ref.	Energy Consumption	Execution Time	Makespan Time	Delay	Cost	Throughput	Completion Time	Response Time	Environment
[54]	X	X	X	X	X	X	✓	X	Fog
[58]	X	X	X	X	✓	X	X	X	Fog – cloud
[56]	X	X	X	✓	✓	X	X	X	Fog
[62]	X	✓	✓	X	X	X	X	X	IoT – Fog
[57]	✓	X	X	✓	✓	X	X	X	Fog
[55]	X	X	X	✓	X	X	X	X	Fog –edge
[60]	✓	X	X	X	X	X	X	✓	Fog
[61]	✓	X	X	X	X	X	X	✓	Fog
[59]	X	X	X	X	X	X	✓	X	Fog
[53]	✓	X	X	X	X	X	X	X	Fog
[64]	✓	X	✓	X	X	X	X	X	Fog
[65]	X	✓	X	✓	X	X	X	X	Fog – cloud
[66]	✓	X	X	✓	X	✓	X	X	Edge – cloud
[2]	✓	X	X	X	✓	X	X	X	Fog
[67]	X	X	X	✓	X	X	X	✓	Fog – cloud
[68]	✓	X	✓	X	X	X	X	X	Fog – cloud
[18]	✓	X	✓	X	X	X	X	X	Fog- cloud
[69]	X	X	✓	X	✓	X	X	X	Fog
[70]	X	X	X	✓	X	X	X	X	Fog
[71]	X	✓	X	X	X	X	X	✓	Fog
[72]	✓	X	✓	✓	X	X	X	X	Fog- cloud
[73]	✓	X	X	X	✓	X	X	X	Fog
[74]	X	X	X	✓	X	X	X	✓	Fog
[27]	X	X	X	✓	X	X	✓	X	Fog- cloud
[75]	✓	X	✓	X	X	X	X	X	Fog – cloud
[76]	X	X	X	✓	✓	X	X	X	Fog – cloud
[63]	✓	X	X	X	X	✓	✓	X	Mobile - fog
[39]	✓	✓	X	X	X	X	X	✓	Fog – cloud
[101]	✓	X	X	X	X	X	X	X	Fog – cloud
[102]	✓	X	X	✓	X	X	X	X	Fog – cloud
[103]	✓	X	✓	X	X	X	X	X	Fog – cloud
[104]	✓	X	✓	X	X	X	X	X	Fog
[108]	✓	X	✓	X	X	X	X	X	Fog – cloud
[7]	✓	X	✓	X	X	X	X	X	Edge
[41]	✓	X	X	X	X	X	X	X	Fog
[105]	X	✓	X	X	✓	✓	X	X	Fog – cloud

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

[106]	✓	X	✓	X	X	X	X	X	Fog – cloud
[107]	X	✓	X	X	X	X	X	X	Fog – cloud
[90]	✓	X	✓	X	X	X	X	✓	Fog – cloud
[93]	✓	X	✓	X	X	X	X	X	Fog – cloud
[91]	✓	X	X	✓	X	X	X	X	Fog – Cloud
[94]	✓	X	X	X	✓	X	✓	X	Fog- cloud
[95]	✓	X	✓	X	✓	X	X	X	Fog- cloud
[96]	X	X	✓	X	✓	X	X	X	Fog- cloud
[97]	X	X	✓	X	X	✓	X	X	Fog
[98]	✓	X	✓	X	X	X	X	X	Fog- cloud
[99]	✓	X	✓	X	X	X	X	X	Fog
[100]	✓	X	X	X	X	X	✓	X	Fog- cloud
Total	30	**6**	**18**	**13**	**11**	**4**	6	**7**	

Table 8 provides a comparative overview of task scheduling techniques employed in fog–cloud computing environments, with a specific focus on the simulation platforms used for performance evaluation. The analysis highlights commonly utilized tools such as CloudSim, iFogSim, Python-based simulation environments, MATLAB, and other customized or hybrid frameworks. This comparison offers insights into the suitability, modeling capabilities, and adaptability of each simulation tool, enabling researchers to identify appropriate platforms for replicating or extending existing studies.

Table 8: Comparison based on Simulation Tools used in Task Scheduling

Ref.	Cloudsim	iFogsim	Python	Matlab	Others	Environment
[54]			✓			Fog
[58]			✓			Fog – cloud
[56]			✓			Fog
[62]			✓			IoT – Fog
[57]			✓			Fog
[55]			✓			Fog –edge
[60]				✓		Fog
[61]				✓		Fog
[59]			✓			Fog
[53]			✓			Fog
[64]					✓	Fog
[65]	✓					Fog – cloud
[66]				✓		Edge – cloud
[2]			✓			Fog
[67]				✓		Fog – cloud
[68]				✓		Fog – cloud
[18]			✓			Fog- cloud
[69]		✓				Fog
[70]		✓				Fog

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

[71]	✓					Fog
[72]					✓	Fog- cloud
[73]	✓					Fog
[74]		✓				Fog
[27]				✓		Fog- cloud
[75]		✓				Fog – cloud
[76]				✓		Fog – cloud
[63]	✓					Mobile – fog
[39]		✓				Fog – cloud
[101]			✓			Fog – cloud
[102]	✓					Fog – cloud
[103]			✓			Fog – cloud
[104]					✓	Fog
[108]			✓			Fog – cloud
[7]			✓			Edge
[41]	✓					Fog
[105]			✓			Fog – cloud
[106]		✓				Fog – cloud
[107]		✓				Fog – cloud
[90]				✓		Fog – cloud
[93]			✓			Fog – cloud
[91]				✓		Fog – Cloud
[94]		✓				Fog- cloud
[95]		✓				Fog- cloud
[96]				✓		Fog- cloud
[97]				✓		Fog
[98]					✓	Fog- cloud
[99]					✓	Fog
[100]					✓	Fog- cloud
Total	**6**	**9**	**16**	**11**	6	

5. DISCUSSION AND COMPARISON

This section presents a systematic review and provides an analytical discussion of the current task scheduling techniques in fog–cloud computing, focusing on approaches based on machine learning, heuristics, and metaheuristics. Task scheduling techniques in fog–cloud computing environments have been examined for the period spanning 2018 to 2025. Based on the reviewed literature, these approaches can be categorized into three major groups: machine learning-based methods, heuristic-based strategies, and metaheuristic techniques. Figure 6 illustrates the performance of various techniques used in fog - cloud task scheduling environment. The results indicate that metaheuristic approaches are the most widely adopted with (44%), followed by heuristic methods with (36%) , while machine learning-based techniques appear to be the least utilized with (20%).

The dominance of metaheuristic approaches in fog–cloud task scheduling research with (44%), compared to heuristic and machine learning techniques, is due to their high flexibility and adaptability, which make them particularly well-suited for dynamic fog–cloud environments.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

These methods can respond to real-time changes and rapidly adjust scheduling decisions as system conditions evolve, enabling effective handling of complex optimization problems and large, dynamic solution spaces. Their suitability for NP-hard problems and real-time applications further explains their prevalence in the literature. However, despite these advantages, metaheuristic methods can face limitations, including high computational overhead, slow convergence in large-scale systems, and the risk of being trapped in local optima if not carefully tuned. The reason why heuristic approaches with (36%) are the second most used in the reviewed articles is that they provide a good balance between simplicity and performance. They are computationally efficient, easy to implement, and capable of delivering reasonably effective scheduling decisions without requiring extensive resources or complex model training. This makes them suitable for fog–cloud environments, especially where lightweight and rapid task allocation is required. However, their main limitation is that they may not guarantee optimal results and can struggle to adapt to highly dynamic or large-scale environments compared to more advanced metaheuristic approaches
Finally The reason why machine learning (ML) performs the least in task scheduling for fog–cloud computing with (20%) is that, despite its ability to learn from historical system data, adapt to changing workloads, predict task execution times, and support proactive and scalable real-time decision-making, ML methods face significant limitations in this context. They require substantial computational resources and large datasets for training, may struggle to generalize to scenarios that differ from their training data, can be complex and time-consuming to design and tune, often lack interpretability, and may introduce latency if predictions or model updates are not efficiently managed. These challenges reduce their overall effectiveness compared to heuristic and metaheuristic approaches in dynamic fog–cloud environments.

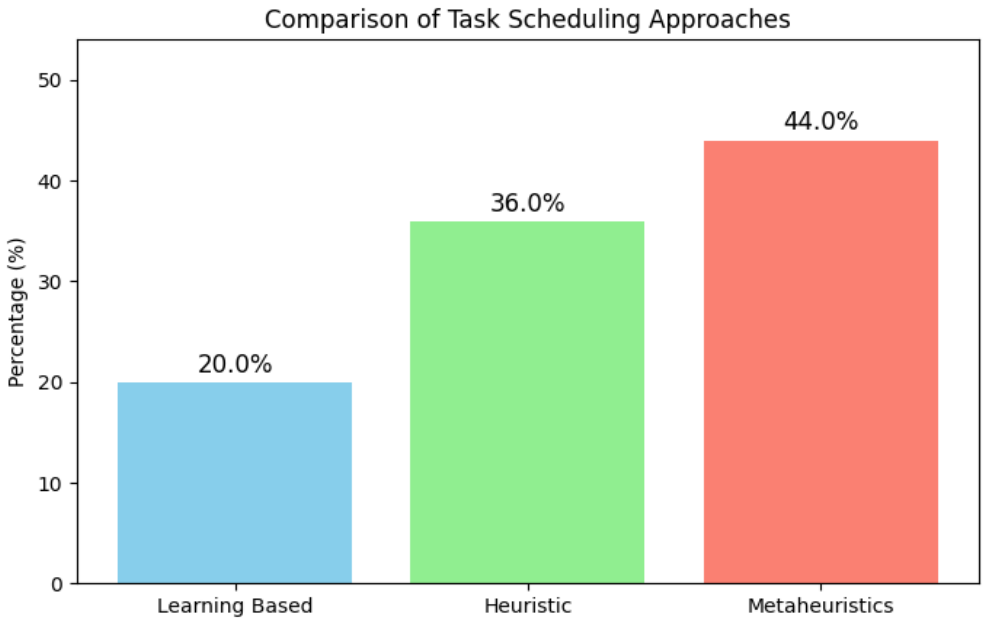

Figure 3: Task scheduling Approaches

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

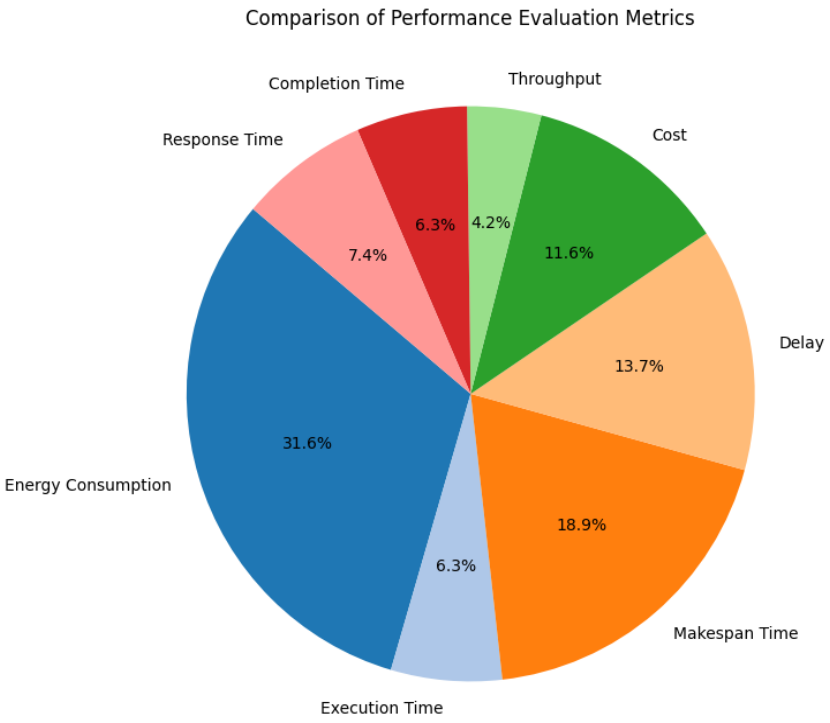

Figure 4: Quality of Service Performance Evaluation Metrics

According to the data in Figure 7, energy consumption is the most frequently used evaluation metric in the reviewed literature on task scheduling techniques, with 32.6% of the recommended strategies focusing on minimizing energy usage. Makespan time is the second most common metric, representing 18.9% of usage, followed by delay at 13.7%, cost at 11.6%, and response time at 7.4%. Completion and execution time metrics together account for 6.3%, while throughput is the least considered, with only 4.2% of the studies using it as a criterion for evaluating task scheduling performance.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

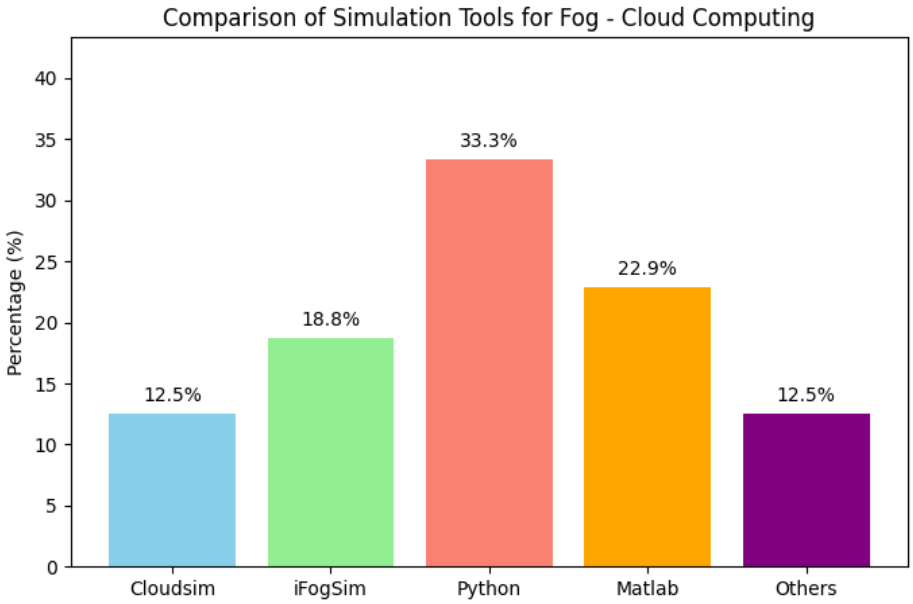

Figure 5: Simulation Tools

Figure 8 presents the distribution of simulation tools used for task scheduling in fog–cloud environments. Python is the predominant tool, utilized in 33.3% of the studies, followed by Matlab at 22.9%. iFogSim accounts for 18.8%, while CloudSim and other tools are each employed in 12.5% of the reviewed studies.

5.1 Comparative Performance of Scheduling Approaches under Different System Conditions

While numerous task scheduling techniques have been proposed for fog–cloud computing environments, their effectiveness varies significantly depending on system objectives and operating conditions. A comparative analysis of the reviewed studies reveals clear performance trends across latency-sensitive and energy-constrained scenarios.

For latency-sensitive environments, such as real-time IoT applications, healthcare monitoring systems, and intelligent transportation systems, machine learning–based approaches, particularly deep reinforcement learning (DRL) and hybrid metaheuristic methods, consistently demonstrate superior performance. These approaches dynamically adapt to changing workloads and network conditions, enabling faster decision-making and reduced task response times. Studies employing DRL-based schedulers report notable reductions in response time and service delay due to their ability to learn optimal scheduling policies from historical and real-time system states. Similarly, hybrid metaheuristic algorithms that combine fast-converging techniques (e.g., PSO) with strong global search capabilities (e.g., GA or GWO) achieve lower makespan and improved responsiveness, making them well-suited for delay-critical applications.

In contrast, for energy-constrained systems, such as battery-powered fog nodes, mobile edge devices, and large-scale IoT deployments, metaheuristic-based approaches consistently outperform other techniques. Algorithms such as Genetic Algorithm (GA), Grey Wolf Optimizer (GWO), Quantum-behaved Particle Swarm Optimization (QPSO), and their hybrid variants demonstrate strong capabilities in minimizing energy consumption while maintaining acceptable execution times. These methods effectively balance exploration and exploitation in large search spaces, enabling efficient resource utilization and reduced power consumption. Empirical evidence from multiple studies indicates that energy-aware metaheuristic

https://doi.org/10.5281/zenodo.18407391

schedulers achieve substantial reductions in total energy usage compared to heuristic and rule-based methods, particularly under high workload conditions.

Heuristic-based approaches, although computationally lightweight and easy to implement, generally perform well only in small-scale or moderately dynamic environments. Their reliance on predefined rules limits their adaptability, resulting in suboptimal performance when system conditions fluctuate significantly. Consequently, heuristic methods are more suitable for scenarios with predictable workloads and strict execution deadlines but are less effective in highly dynamic or heterogeneous fog–cloud systems.

Overall, the comparative analysis suggests that no single scheduling approach is universally optimal. Machine learning and hybrid metaheuristic methods are better suited for latency-sensitive and highly dynamic environments, while metaheuristic algorithms excel in energy-constrained systems. This highlights the importance of context-aware and hybrid scheduling frameworks that adaptively select or combine multiple techniques based on application requirements, resource availability, and QoS objectives.

In certain scenarios, simple heuristic-based scheduling approaches can outperform complex AI models in fog–cloud environments. Specifically, in small-scale systems with predictable workloads, limited task diversity, and stable network conditions, heuristics provide faster scheduling decisions with negligible computational overhead. In real-time and delay-critical applications, such as emergency response or industrial control systems, the low execution latency of heuristics enables timely task allocation, whereas AI-based models may introduce delays due to training, inference, or model updating. Additionally, in resource-constrained fog nodes with limited processing power and memory, heuristics are more practical and reliable than machine learning models, which require substantial computational and storage resources. These findings indicate that lightweight heuristic methods remain effective and sometimes preferable when system simplicity, responsiveness, and low overhead are primary requirements.

5.2 Threats to Validity

Despite efforts to conduct a comprehensive and unbiased review, this study may be subject to publication and simulation bias. Publication bias arises because studies reporting positive or improved performance outcomes are more likely to be published than those with negative or neutral results. Additionally, simulation bias is evident as most reviewed studies rely on simulated environments (e.g., CloudSim, iFogSim, MATLAB, Python-based simulators), which may not fully capture real-world network dynamics, hardware constraints, or workload variability. These factors can lead to optimistic performance evaluations and limit the generalizability of the findings. Future research should prioritize real-world deployments, standardized benchmarks, and reproducible experimental setups to mitigate these biases.

5.3 Limitations

This research focuses on employing machine learning, heuristic, and metaheuristic algorithms as primary strategies to address the NP-hard problem of efficient resource scheduling in fog–cloud computing environments. However, the study does not comprehensively cover all constraints and Quality of Service (QoS) parameters, such as load balancing, storage capacity, SLA violations, and bandwidth limitations, and the investigation of practical, real-world applications remains limited. Moreover, only conference papers and peer-reviewed journal articles were considered, which may have led to the exclusion of relevant works published in other formats, while some potentially important studies were not included due to restricted access or purchase requirements.

5.4 Future Research

Fog–cloud computing environments have made significant strides in developing robust computing infrastructures. While metaheuristic algorithms have been successfully applied in various contexts, considerable scope remains for further exploration of contemporary challenges. Future research

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

should focus on integrating metaheuristics with artificial intelligence (AI) to enhance scheduling efficiency, aiming to minimize resource usage while ensuring efficient task execution and maintaining desired levels of Quality of Service (QoS). As demonstrated by state-of-the-art approaches, no single algorithm can address all scheduling objectives simultaneously, highlighting the potential of hybrid methods that combine metaheuristics and AI to optimize multiple performance metrics effectively.

5.5 Practical Implementation Challenges of Metaheuristic-based Scheduling in Fog - Cloud Computing

Although metaheuristic algorithms have shown strong potential for optimizing task scheduling in fog - cloud computing environments, their deployment in real-world systems introduces several practical challenges that require careful consideration.

i. *Dynamic and Unpredictable Workloads* IoT-generated tasks arrive dynamically and often unpredictably. Metaheuristic algorithms may struggle to adapt quickly to sudden workload fluctuations, leading to suboptimal scheduling decisions in real-time environments.
ii. *Task Dependency and Workflow Management* Many IoT applications involve dependent tasks or workflows (e.g., DAG-based tasks). Handling task dependencies efficiently within metaheuristic frameworks increases complexity and may prolong scheduling and execution time.
iii. ***Algorithm Complexity and Convergence Time*** Although some hybrid metaheuristic algorithms are effective at reaching global optimal solutions, they often exhibit high computational complexity and slow convergence rates. These characteristics limit their scalability and reduce responsiveness, especially in scenarios involving large-scale task scheduling.
iv. ***Scalability and Robustness*** in highly distributed and dynamic edge environments, task scheduling algorithms are required to scale efficiently across numerous nodes while remaining resilient to network variations, node failures, and fluctuating workloads. However, many existing metaheuristic approaches do not adequately account for these real-world deployment conditions, creating a gap between their theoretical effectiveness and practical applicability.
v. *VM Availability and Resource Heterogeneity* Fog–cloud systems consist of heterogeneous resources with varying capabilities and availability. Metaheuristic schedulers may face difficulty in handling frequent VM state changes, mobility, or failures.

6. CONCLUSION

This paper investigates the state-of-the-art task scheduling approaches in fog–cloud computing. Based on the reviewed articles from 2018 to 2025, the number of publications in the field of fog–cloud task scheduling peaked in 2021. A taxonomy of fog – cloud task scheduling techniques is presented, encompassing machine learning, heuristic-based, and metaheuristic-based approaches, with the selected articles analyzed within these three categories. The benefits and limitations of each technique are reviewed according to the quality criteria defined in the study. The analysis reveals that energy consumption and makespan time are the most critical evaluation criteria for implementing effective scheduling, while Python is the most widely used simulation tool in fog–cloud environments. Overall, the survey aims to provide a comprehensive understanding of task

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391

scheduling strategies to support energy reduction, performance improvement, latency minimization, security enhancement, and fault-tolerance management in fog–cloud computing.

REFERENCES

[1] M. Arif, F. Ajesh, S. Shamsudheen, and M. Shahzad, "Retracted: Secure and Energy-Efficient Computational Offloading Using LSTM in Mobile Edge Computing," vol. 2022, 2024.

[2] M. Trabelsi and S. Ben Ahmed, "Energy and Cost-Aware Real-Time Task Scheduling with Deadline-Constraints in Fog Computing Environments," no. Enase, pp. 434–441, 2024, doi: 10.5220/0012637600003687.

[3] P. Choppara and B. Lokesh, "Efficient Task Scheduling and Load Balancing in Fog Computing for Crucial Healthcare Through Deep Reinforcement Learning," *IEEE Access*, vol. 13, no. February, pp. 26542–26563, 2025, doi: 10.1109/ACCESS.2025.3539336.

[4] J. Aminu, R. Latip, Z. M. Hanafi, S. Kamarudin, and D. Gabi, *Systematic review of metaheuristic-based task scheduling strategies in edge computing environments*. 2025.

[5] P. Choppara and S. Mangalampalli, "An efficient deep reinforcement learning based task scheduler in cloud-fog environment," *Cluster Comput.*, vol. 28, no. 1, pp. 1–26, 2025, doi: 10.1007/s10586-024-04712-z.

[6] X. Yu, M. Zhu, M. Zhu, X. Zhou, and L. Long, "Location-aware job scheduling for IoT systems using cloud and fog," *Alexandria Eng. J.*, vol. 110, no. September 2024, pp. 346–362, 2025, doi: 10.1016/j.aej.2024.09.055.

[7] J. Aminu, R. Latip, Z. M. Hanafi, S. Kamarudin, B. U. Kangiwa, and A. Liman, "An Enhanced Grey Wolf Optimization Algorithm for Efficient Task Scheduling in Mobile Edge Computing," *Int. J. Comput. Appl.*, vol. 186, no. 56, pp. 39–44, 2024, doi: 10.5120/ijca2024924287.

[8] P. Choppara and S. S. Mangalampalli, "Resource Adaptive Automated Task Scheduling Using Deep Deterministic Policy Gradient in Fog Computing," *IEEE Access*, vol. 13, no. February, pp. 25969–25994, 2025, doi: 10.1109/ACCESS.2025.3539606.

[9] J. Aminu, R. Latip, Z. M. Hanafi, S. Kamarudin, and D. Gabi, "Efficient Task Allocation for Energy and Execution Time Trade-Off in Edge Computing Using Multi-Objective IPSO," 2025, doi: 10.32604/cmc.2025.062451.

[10] A. Khan, F. Ullah, D. Shah, M. H. Khan, S. Ali, and M. Tahir, "EcoTaskSched : a hybrid machine learning approach for energy- efficient task scheduling in IoT- based fog-cloud environments," pp. 1–27, 2025.

[11] R. Latip, J. Aminu, Z. Mohd, H. Shafinah, and K. Danlami, "Metaheuristic task offloading approaches for minimization of energy consumption on edge computing : a systematic review," *Discov. Internet Things*, 2024, doi: 10.1007/s43926-024-00089-y.

[12] Z. Movahedi, *An efficient population-based multi-objective task scheduling approach in fog computing systems*. 2021.

[13] J. Aminu *et al.*, "Systematic Review of Metaheuristic Task Scheduling Algorithms on Edge Computing," 2024.

[14] C. Computing, "Performance Optimization of Task Scheduling in Fog and Edge Computing using meta-heuristic algorithms for IoT Networks : A Comparative Study Pratyush Nigel Baxla National College of Ireland Supervisor :".

[15] S. Reports, "Scientific Reports Article in Press Improved multi-strategy secretary bird

https://doi.org/10.5281/zenodo.18407391

optimization for efficient IoT task scheduling in fog cloud computing IN IN," 2025.
[16] V. Sharma and M. Bala, "An Improved Task Allocation Strategy in Cloud using Modified K-means Clustering Technique," *Egypt. Informatics J.*, vol. 21, no. 4, pp. 201–208, 2020, doi: 10.1016/j.eij.2020.02.001.
[17] H. Zavieh and A. Javadpour, "Enhanced Efficiency in Fog Computing : A Fuzzy Data-Driven Machine Selection Strategy," *Int. J. Fuzzy Syst.*, vol. 26, no. 1, pp. 368–389, 2024, doi: 10.1007/s40815-023-01605-y.
[18] C. Framework, "EEOA : Cost and Energy Efficient Task Scheduling in a Cloud-Fog Framework," 2023.
[19] P. Choppara and S. Mangalampalli, "EAI Endorsed Transactions An Effective analysis on various task scheduling algorithms in Fog computing," vol. 10, pp. 1–7, 2024, doi: 10.4108/eetiot.4589.
[20] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog Computing May Help to Save Energy in Cloud Computing," vol. 8716, no. c, pp. 1–12, 2016, doi: 10.1109/JSAC.2016.2545559.
[21] R. A. Al-arasi and A. Saif, "on Cloud Systems EAI Endorsed Transactions Task scheduling in cloud computing based on meta- heuristic techniques : A review paper," vol. 6, no. 17, 2020.
[22] A. Singhrova, "PRIORITIZED GA-PSO ALGORITHM FOR EFFICIENT RESOURCE," vol. 11, no. 6, pp. 907–916, 2020.
[23] F. Chad, "Adaptive Resource Allocation for Energy- Efficient Task Offloading in Mobile Cloud Computing Author : Felix Chad Date : 24 th Feb 2025 Abstract," no. February, 2025.
[24] H. Suleiman, "A Cost-Aware Framework for QoS-Based and Energy-Efficient Scheduling in Cloud – Fog Computing," 2022.
[25] Z. Sun, C. Li, L. Wei, Z. Li, Z. Min, and G. Zhao, "Intelligent sensor-cloud in fog computer: A novel hierarchical data job scheduling strategy," *Sensors (Switzerland)*, vol. 19, no. 23, 2019, doi: 10.3390/s19235083.
[26] M. Adhikari and H. Gianey, "Energy efficient offloading strategy in fog-cloud environment for IoT applications," *Internet of Things (Netherlands)*, vol. 6, p. 100053, 2019, doi: 10.1016/j.iot.2019.100053.
[27] W. Wang, G. Wu, Z. Guo, L. Qian, L. Ding, and F. Yang, *Data scheduling and resource optimization for fog computing architecture in industrial IoT*, vol. 11319 LNCS. Springer International Publishing, 2019. doi: 10.1007/978-3-030-05366-6_11.
[28] I. Engineering, "Energy Consumption Tradeoff for," *ICC 2019 - 2019 IEEE Int. Conf. Commun.*, pp. 1–6, 2019.
[29] J. Wang and D. Li, "Task scheduling based on a hybrid heuristic algorithm for smart production line with fog computing," *Sensors (Switzerland)*, vol. 19, no. 5, 2019, doi: 10.3390/s19051023.
[30] M. Barzegaran, A. Cervin, and P. Pop, "Towards quality-of-control-aware scheduling of industrial applications on fog computing platforms," *IoT-Fog 2019 - Proc. 2019 Work. Fog Comput. IoT*, pp. 1–5, 2019, doi: 10.1145/3313150.3313217.
[31] L. Abualigah and A. Diabat, "A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments," *Cluster Comput.*, vol. 24, no. 1, pp. 205–223, 2021, doi: 10.1007/s10586-020-03075-5.
[32] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for IoT systems:

https://doi.org/10.5281/zenodo.18407391

A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, 2018, doi: 10.1109/JIOT.2018.2838022.

[33] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog Computing: A taxonomy, survey and future directions," *Internet of Things*, vol. 0, no. 9789811058608, pp. 103–130, 2018, doi: 10.1007/978-981-10-5861-5_5.

[34] R. K. Naha *et al.*, "Fog computing: Survey of trends, architectures, requirements, and research directions," *IEEE Access*, vol. 6, no. c, pp. 47980–48009, 2018, doi: 10.1109/ACCESS.2018.2866491.

[35] Z. Hao, E. Novak, S. Yi, and Q. Li, "Challenges and Software Architecture for Fog Computing," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 44–53, 2017, doi: 10.1109/MIC.2017.26.

[36] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl.*, vol. 98, pp. 27–42, 2017, doi: 10.1016/j.jnca.2017.09.002.

[37] Y. Liu, A. Yang, Q. Zeng, Y. Sun, J. Gao, and Z. Lv, "Task Scheduling of Real-Time Traffic Information Processing Based on Digital Twins," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 13171–13179, 2023, doi: 10.1109/TITS.2022.3196166.

[38] Q. Qi *et al.*, "Scalable Parallel Task Scheduling for Autonomous Driving Using Multi-Task Deep Reinforcement Learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13861–13874, 2020, doi: 10.1109/TVT.2020.3029864.

[39] H. Rafique, M. A. Shah, S. U. Islam, T. Maqsood, S. Khan, and C. Maple, "A Novel Bio-Inspired Hybrid Algorithm (NBIHA) for Efficient Resource Management in Fog Computing," *IEEE Access*, vol. 7, pp. 115760–115773, 2019, doi: 10.1109/ACCESS.2019.2924958.

[40] X. Yang and N. Rahmani, "Task scheduling mechanisms in fog computing: review, trends, and perspectives," *Kybernetes*, vol. 50, no. 1, pp. 22–38, 2021, doi: 10.1108/K-10-2019-0666.

[41] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *J. Parallel Distrib. Comput.*, vol. 143, pp. 88–96, 2020, doi: 10.1016/j.jpdc.2020.04.008.

[42] A. Asensio *et al.*, "Designing an efficient clustering strategy for combined Fog-to-Cloud scenarios," *Futur. Gener. Comput. Syst.*, vol. 109, pp. 392–406, 2020, doi: 10.1016/j.future.2020.03.056.

[43] A. Shahidinejad, M. Ghobaei-Arani, and M. Masdari, "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach," *Cluster Comput.*, vol. 24, no. 1, pp. 319–342, 2021, doi: 10.1007/s10586-020-03107-0.

[44] I. Ullah and H. Y. Youn, "Task Classification and Scheduling Based on K-Means Clustering for Edge Computing," *Wirel. Pers. Commun.*, vol. 113, no. 4, pp. 2611–2624, 2020, doi: 10.1007/s11277-020-07343-w.

[45] P. Vinothiyalakshmi and R. Anitha, "Workload Mining in Cloud Computing using Extended Cloud Dempster–Shafer Theory (ECDST)," *Wirel. Pers. Commun.*, vol. 114, no. 1, pp. 389–407, 2020, doi: 10.1007/s11277-020-07368-1.

[46] M. Hosseinzadeh *et al.*, "Task Scheduling Mechanisms for Fog Computing : A Systematic Survey," *IEEE Access*, vol. 11, no. May, pp. 50994–51017, 2023, doi: 10.1109/ACCESS.2023.3277826.

[47] V. Sharma and A. K. Tripathi, "A systematic review of meta-heuristic algorithms in IoT based application," *Array*, vol. 14, no. April, p. 100164, 2022, doi: 10.1016/j.array.2022.100164.
[48] Z. A. Khan, I. A. Aziz, N. A. B. Osman, and I. Ullah, "A Review on Task Scheduling Techniques in Cloud and Fog Computing: Taxonomy, Tools, Open Issues, Challenges, and Future Directions," *IEEE Access*, vol. 11, no. November, pp. 143417–143445, 2023, doi: 10.1109/ACCESS.2023.3343877.
[49] S. Bansal, "S U RV E Y A RT I C L E A systematic review of task scheduling approaches in fog computing," no. January, 2022, doi: 10.1002/ett.4523.
[50] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "aTask scheduling approaches in fog computing: A survey," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 3, pp. 1–11, 2022, doi: 10.1002/ett.3792.
[51] K. Matrouk and K. Alatoun, "Scheduling Algorithms in Fog Computing : A Survey," vol. 9, pp. 59–74, 2021.
[52] M. R. Alizadeh and A. M. Rahmani, "Task scheduling approaches in fog computing : A systematic review," no. July, pp. 1–36, 2020, doi: 10.1002/dac.4583.
[53] S. M. S. Bhanu, "A reinforcement learning algorithm for rescheduling preempted tasks," *J. Sched.*, vol. 25, no. 5, pp. 547–565, 2022, doi: 10.1007/s10951-022-00725-x.
[54] H. Liu *et al.*, "A Federated Learning Multi-Task Scheduling Mechanism Based on Trusted Computing Sandbox," pp. 1–19, 2023.
[55] Z. Wang, M. Goudarzi, M. Gong, and R. Buyya, "Deep Reinforcement Learning-based scheduling for optimizing system load and response time in edge and fog computing environments," *Futur. Gener. Comput. Syst.*, vol. 152, no. August 2023, pp. 55–69, 2024, doi: 10.1016/j.future.2023.10.012.
[56] S. Swarup, E. M. Shakshuki, and A. Yasar, "ScienceDirect ScienceDirect Energy Efficient Task Scheduling in Fog Environment using Deep Energy Efficient Task Scheduling in Fog Environment using Deep Reinforcement Learning Approach Reinforcement Learning Approach," *Procedia Comput. Sci.*, vol. 191, pp. 65–75, 2021, doi: 10.1016/j.procs.2021.07.012.
[57] P. Gazori, D. Rahbari, and M. Nickray, "Saving time and cost on the scheduling of fog-based IoT applications using deep reinforcement learning approach," *Futur. Gener. Comput. Syst.*, vol. 110, pp. 1098–1115, 2020, doi: 10.1016/j.future.2019.09.060.
[58] S. Shadroo, A. M. Rahmani, and A. Rezaee, "The two-phase scheduling based on deep learning in the Internet of Things," *Comput. Networks*, p. 107684, 2020, doi: 10.1016/j.comnet.2020.107684.
[59] H. E. Li, K. Ota, and M. Dong, "Deep Reinforcement Scheduling for Mobile Crowdsensing," vol. 19, no. 2, 2019.
[60] S. Ghanavati, J. Abawajy, D. Izadi, and I. Ntroduction, "Automata-based Dynamic Fault Tolerant Task Scheduling Approach in Fog Computing," vol. 6750, no. c, 2020, doi: 10.1109/TETC.2020.3033672.
[61] H. S. Arri, R. Singh, D. Prashar, and G. P. Joshi, "Optimized Task Group Aggregation-Based Overflow Handling on Fog Computing Environment Using Neural Computing," pp. 1–15, 2021.
[62] S. Arisdakessian *et al.*, "FoGMatch : An Intelligent Multi-Criteria IoT-Fog Scheduling Approach Using Game Theory," pp. 1–11, 2020.

https://doi.org/10.5281/zenodo.18407391

[63] E. Hosseini, M. Nickray, and S. Ghanbari, "Optimized task scheduling for cost-latency trade-off in mobile fog computing using fuzzy analytical hierarchy process," *Comput. Networks*, vol. 206, no. December 2021, p. 108752, 2022, doi: 10.1016/j.comnet.2021.108752.
[64] S. Azizi, M. Shojafar, J. Abawajy, and R. Buyya, "Journal of Network and Computer Applications Deadline-aware and energy-efficient IoT task scheduling in fog computing systems : A semi-greedy approach," *J. Netw. Comput. Appl.*, vol. 201, no. January, p. 103333, 2022, doi: 10.1016/j.jnca.2022.103333.
[65] L. E. I. Li, Q. Guan, and S. Member, "Resource Allocation and Task Offloading for Heterogeneous Real-Time Tasks With Uncertain Duration Time in a Fog Queueing System," *IEEE Access*, vol. 7, pp. 9912–9925, 2019, doi: 10.1109/ACCESS.2019.2891130.
[66] M. Guo, L. Li, and Q. Guan, "Workload Allocation in IoT-Edge-Cloud Computing Systems," no. 1, pp. 1–13.
[67] A. Najafizadeh, A. Salajegheh, A. Masoud, and R. Amir, "Privacy - preserving for the internet of things in multi - objective task scheduling in cloud - fog computing using goal programming approach," *Peer-to-Peer Netw. Appl.*, no. 0123456789, 2021, doi: 10.1007/s12083-021-01222-2.
[68] I. Conference *et al.*, "ScienceDirect ScienceDirect Adaptive Adaptive scheduling scheduling strategy strategy of of fog fog computing computing tasks tasks with with different different priority priority for for intelligent intelligent production production lines lines," 2021, doi: 10.1016/j.procs.2021.02.064.
[69] R. M. B. Lydia and E. V Rhymend, *An improved list - based task scheduling algorithm for fog computing environment*, no. 0123456789. Springer Vienna, 2021. doi: 10.1007/s00607-021-00935-9.
[70] Z. Liu, X. Yang, Y. Yang, K. Wang, and G. Mao, "DATS : Dispersive Stable Task Scheduling in Heterogeneous Fog Networks," *IEEE Internet Things J.*, vol. PP, no. c, p. 1, 2018, doi: 10.1109/JIOT.2018.2884720.
[71] T. Aladwani and T. Aladwani, "ScienceDirect ScienceDirect ScienceDirect Scheduling IoT Tasks Fog Computing Based on their 16th Healthcare Scheduling IoT Healthcare Tasks in Fog Computing Based on their Importance Scheduling IoT Healthcare Tasks in Fog Computing Based on their Importance Importance," *Procedia Comput. Sci.*, vol. 163, pp. 560–569, 2019, doi: 10.1016/j.procs.2019.12.138.
[72] G. Li, J. Yan, L. Chen, J. Wu, Q. Lin, and Y. Zhang, "Energy Consumption Optimization with a Delay Threshold in Cloud-Fog Cooperation Computing," *IEEE Access*, vol. 7, pp. 159688–159697, 2019, doi: 10.1109/ACCESS.2019.2950443.
[73] T. Choudhari and M. Moh, "Prioritized Task Scheduling in Fog Computing".
[74] K. Verma, A. Kumar, M. Salim, U. Islam, T. Kanwar, and M. Bhushan, "Informatics in Medicine Unlocked Rank based mobility-aware scheduling in Fog computing," *Informatics Med. Unlocked*, vol. 24, no. April, p. 100619, 2021, doi: 10.1016/j.imu.2021.100619.
[75] R. M. Abdelmoneem, A. Benslimane, and E. Shaaban, "Mobility-aware task scheduling in cloud-Fog IoT-based healthcare architectures," *Comput. Networks*, vol. 179, no. October 2019, p. 107348, 2020, doi: 10.1016/j.comnet.2020.107348.
[76] A. Najafizadeh and A. Salajegheh, "Multi-objective Task Scheduling in cloud-fog computing using goal programming approach," *Cluster Comput.*, vol. 8, 2021, doi: 10.1007/s10586-021-03371-8.

[77] R. O. Aburukba, T. Landolsi, and D. Omer, "Journal of Network and Computer Applications A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices," *J. Netw. Comput. Appl.*, vol. 180, no. January, p. 102994, 2021, doi: 10.1016/j.jnca.2021.102994.
[78] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Comput. Ind. Eng.*, vol. 137, no. September, p. 106040, 2019, doi: 10.1016/j.cie.2019.106040.
[79] D. Aldossary, E. Aldahasi, and T. Balharith, "A Systematic Literature Review on Load-Balancing Techniques in Fog Computing : Architectures , Strategies , and Emerging Trends," pp. 1–42, 2025.
[80] S. K. Bothra, S. Singhal, С. К. Ботра, and С. Сингхал, "Nature-inspired metaheuristic scheduling algorithms in cloud : a systematic review Биоинспирированные метаэвристические алгоритмы построения расписаний в облаке : систематический обзор," vol. 21, no. 4, pp. 463–472, 2021, doi: 10.17586/2226-1494-2021-21-4-463-472.
[81] A. Khan, A. Abbas, H. A. Khattak, F. Rehman, I. U. Din, and S. Ali, "Effective Task Scheduling in Critical Fog Applications," vol. 2022, 2025, doi: 10.1155/2022/9208066.
[82] N. B. M. Isa, T. C. Wei, and A. H. M. Yatim, "Smart grid technology: Communications, power electronics and control system," *Proceeding - 2015 Int. Conf. Sustain. Energy Eng. Appl. Sustain. Energy Gt. Dev. ICSEEA 2015*, pp. 10–14, 2016, doi: 10.1109/ICSEEA.2015.7380737.
[83] S. S. Mangalampalli, P. V. Reddy, G. R. Karri, G. Tippani, and H. Kota, "Priority-Aware Multi-Objective Task Scheduling in Fog Computing Using Simulated Annealing," pp. 1–30, 2025.
[84] M. Hamzei, S. Khandagh, and N. Jafari Navimipour, "A Quality-of-Service-Aware Service Composition Method in the Internet of Things Using a Multi-Objective Fuzzy-Based Hybrid Algorithm," *Sensors*, vol. 23, no. 16, pp. 1–29, 2023, doi: 10.3390/s23167233.
[85] W. Shu and Y. Li, "Joint offloading strategy based on quantum particle swarm optimization for MEC-enabled vehicular networks," *Digit. Commun. Networks*, vol. 9, no. 1, pp. 56–66, 2023, doi: 10.1016/j.dcan.2022.03.009.
[86] A. Kumar, M. Nadeem, and H. Banka, "Nature inspired optimization algorithms: a comprehensive overview," *Evol. Syst.*, vol. 14, no. 1, pp. 141–156, 2023, doi: 10.1007/s12530-022-09432-6.
[87] O. A. Alomari *et al.*, "Hybrid Feature Selection Based on Principal Component Analysis and Grey Wolf Optimizer Algorithm for Arabic News Article Classification," *IEEE Access*, vol. 10, pp. 121816–121830, 2022, doi: 10.1109/ACCESS.2022.3222516.
[88] A. R. Kashani, C. V. Camp, M. Rostamian, K. Azizi, and A. H. Gandomi, *Population-based optimization in structural engineering: a review*, vol. 55, no. 1. Springer Netherlands, 2022. doi: 10.1007/s10462-021-10036-w.
[89] W. Sun, M. Tang, L. Zhang, Z. Huo, and L. Shu, "A survey of using swarm intelligence algorithms in IoT," *Sensors (Switzerland)*, vol. 20, no. 5, 2020, doi: 10.3390/s20051420.
[90] M. A. Elaziz, I. Attiya, L. Abualigah, M. Iqbal, and A. Ali, "Hybrid Enhanced Optimization-Based Intelligent Task Scheduling for Sustainable Edge Computing," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 889–898, 2024, doi: 10.1109/TCE.2023.3321783.
[91] F. A. Saif, R. Latip, Z. M. H. Member, and K. Shafinah, "Multi-objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing," *IEEE Access*, vol.

PP, p. 1, 2023, doi: 10.1109/ACCESS.2023.3241240.
[92] J. Akram and A. Rafi, "Efficient Resource Utilization in Cloud-Fog Environment Integrated with Smart Grids," *2018 Int. Conf. Front. Inf. Technol.*, pp. 188–193, 2018, doi: 10.1109/FIT.2018.00040.
[93] A. R. Kadhim and F. Rabee, "Task Scheduling Optimization in Cloud-Fog- MCS Environment Using Genetic Algorithm and Game Theory," vol. 13, no. 3, 2024, doi: 10.18178/ijeetc.13.3.200-213.
[94] M. Gamal, S. Awad, R. F. Abdel-kader, K. Abd, and E. Salam, "Efficient offloading and task scheduling in internet of things- cloud-fog environment," vol. 14, no. 4, pp. 4445–4455, 2024, doi: 10.11591/ijece.v14i4.pp4445-4455.
[95] D. Subramoney and C. N. Nyirenda, "Multi-Swarm PSO Algorithm for Static Workflow Scheduling in Cloud-Fog Environments," *IEEE Access*, vol. 10, no. October, pp. 117199–117214, 2022, doi: 10.1109/ACCESS.2022.3220239.
[96] A. S. A. Amir, E. Ghamry, and E. Hamouda, *Real - Time Task Scheduling Algorithm for IoT - Based Applications in the Cloud – Fog Environment*. Springer US, 2022. doi: 10.1007/s10922-022-09664-6.
[97] I. Attiya, L. Abualigah, D. Elsadek, S. A. Chelloug, and M. A. Elaziz, "An Intelligent Chimp Optimizer for Scheduling of IoT Application Tasks in Fog Computing," pp. 1–18, 2022.
[98] N. L. S. Fonseca, "Task scheduling in cloud-fog computing systems," pp. 962–977, 2021.
[99] V. Yadav, B. V. Natesha, and R. M. R. Guddeti, "GA-PSO: Service Allocation in Fog Computing Environment Using Hybrid Bio-Inspired Algorithm," *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, vol. 2019-Octob, pp. 1280–1285, 2019, doi: 10.1109/TENCON.2019.8929234.
[100] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing," *Futur. Gener. Comput. Syst.*, vol. 111, pp. 539–551, 2020, doi: 10.1016/j.future.2019.09.039.
[101] T. Wang, X. Wei, C. Tang, and J. Fan, "Efficient multi-tasks scheduling algorithm in mobile cloud computing with time constraints," 2017, doi: 10.1007/s12083-017-0561-9.
[102] M. Jia, J. Zhu, and H. Huang, "Energy and delay-ware massive task scheduling in fog-cloud computing system," 2021.
[103] M. Tanha, M. Hosseini, S. Amir, and M. Rahmani, *A hybrid meta-heuristic task scheduling algorithm based on genetic and thermodynamic simulated annealing algorithms in cloud computing environments*, vol. 9. Springer London, 2021. doi: 10.1007/s00521-021-06289-9.
[104] M. A. Basset and M. J. Ryan, "IEGA : An improved elitism - based genetic algorithm for task scheduling problem in fog computing," no. April, pp. 1–40, 2021, doi: 10.1002/int.22470.
[105] H. Thi, T. Binh, and D. B. Son, "An Evolutionary Algorithm for Solving Task Scheduling Problem in Cloud-Fog Computing Environment," pp. 397–404, 2020.
[106] T. Journal, "No Title," vol. 4662, no. c, 2020, doi: 10.1109/JIOT.2020.3012617.
[107] M. Ghobaei-arani, "An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing," no. December 2018, pp. 1–14, 2019, doi: 10.1002/ett.3770.
[108] Z. Movahedi, *An efficient population-based multi-objective task scheduling approach in fog computing systems*. Journal of Cloud Computing, 2021.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.18407391