International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V73I11P104

Volume 73 Issue 11, 39-50, November 2025

Review Article

Heuristic-Based Workload Scheduling Approaches in Edge-Cloud Environments: A Review

Hasnae NOUHAS[1*], Abdessamad BELANGOUR[1], Mahmoud NASSAR[2]

[1]*Laboratory of Artificial Intelligence and Systems, LAIS, Hassan II University, Faculty of Sciences Ben M'sik, Casablanca, Morocco.*
[2]*IMS Team, ADMIR Laboratory, ENSIAS, Rabat IT Center, Mohammed V University in Rabat, Rabat, Morocco.*

[*]*Corresponding Author : hasnae_nouhas@um5.ac.ma*

Received: 01 July 2025 Revised: 29 October 2025 Accepted: 03 November 2025 Published: 25 November 2025

Abstract - *Edge-cloud computing architecture has become appropriate and promising to satisfy the performance requirements of resource-intensive and latency-critical applications. Effective scheduling of workload serves the purpose of exploiting the distributed, heterogeneous, and dynamic characteristics of these environments. Among the developed approaches, heuristic-based scheduling methods are important due to the low level of complexity and practical merit they present in real-time and resource-limited environments. Heuristic-based workload scheduling methods are the center of focus of this paper. Present methods are classified into simple heuristics, metaheuristics, and hybrid schemes, and surveyed on prominent examples of HEFT, ACO, PSO, Min-Min/Max-Min, and Greedy Resource-Aware algorithms. Each of these is put through the prism of the scheduling objectives, benefits, and their tradeoff on various executed metrics like latency, energy efficiency, and flexibility. Though these strategies are beneficial, the issues associated with their usability for dynamic applications and multi-objective scenarios are prominent. Important research gaps are listed along with proposed future works, including adaptations, energy-oriented, and lightweight scheduling models. Of even higher value, it is notable to recognize the growing interest in the application of AI-based schemes, which have the potential to enhance heuristic-based scheduling once integrated into hybrid systems. This survey aspires to present the convenient go-to thesis of the researcher tackling the challenge of creating a productive workload scheduling design in the edge-cloud infrastructures.*

Keywords - *Cloud computing, Edge computing, Workload scheduling, Heuristic Algorithms, Metaheuristic Algorithms.*

1. Introduction

Massive-scale deployments of Internet of Things (IoT) devices and emerging applications, which are latency-sensitive and highly data-intensive, have given rise to the formation of edge-cloud computing paradigms. In this approach, computation occurs at the edge nodes, which are closer to data sources and remote cloud servers, resulting in greater responsiveness, reduced bandwidth costs, and improved scalability. Applications, such as real-time video analytics, self-driving cars, and intelligent health care systems, are increasingly including this hybrid infrastructure so as to satisfy excessive Quality of Service (QoS) and Quality of Experience (QoE) constraints [1]. Workload scheduling, which falls among the major issues in these environments, defies decisions on where, how, and when the computing tasks must be executed on the distributed and heterogeneous resources. Edge-cloud infrastructure is more challenging than the traditional cloud-based infrastructure due to the role of the more dynamic network scenarios, constrained edge resources, and the non-stationary end-users [2]. Smart scheduling must make the proper balance between the competing ends like latency, energy, resource, and SLA fulfillment. Poor scheduling in edge–cloud environments can have serious consequences in real-world systems. For example, in autonomous vehicles, inadequate scheduling (e.g., suboptimal offloading, task dropping) can lead to safety risks: a recent study in Vehicular MEC demonstrated that poor scheduling under fixed bandwidth leads to dropped tasks and elevated latency, whereas optimized scheduling avoids task drops and significantly lowers delay, improving system reliability [3]. In vehicular edge computing, schemes that neglect latency or privacy constraints often underperform: one approach that jointly considered these metrics improved QoS by about 55% relative to state-of-the-art baselines [4].

In healthcare systems, delays in task scheduling for wearable sensor data (such as monitoring and diagnostics) can lead to misdiagnoses, slower response times, or worse patient outcomes. The review Edge Computing in Healthcare: Innovations, Opportunities… shows that real-time decision-making, offloading, and privacy are central issues, and that many healthcare IoT applications are currently constrained by

delays in processing or communication [5]. In another experiment on mobile edge computing in patient monitoring, it is illustrated how the combination of MEC + 5G decreases latency in the data flows of healthcare sensors but also reveals how inadequate bandwidth/resource scheduling continues as a bottleneck [6].

In the same way, for large-scale edge case deployments, studies of balancing latency and energy constraints for heterogeneous networks indicate that suboptimal scheduling could cause unacceptable latency or high waste of energy. On the positive side, optimized scheduling can decrease latency by tens of milliseconds as well as enhance energy efficiency to a significant extent [7]. Such cases highlight the urgent necessity for efficient as well as adaptive scheduling approaches in edge–cloud environments.

There are numerous approaches to scheduling issues, but heuristic methods are very fashionable due to their simplicity of application, as well as processing, along with adaptability. Rather than exact optimization algorithms that can be costly in terms of time and processing, heuristics offer near-optimal answers with less time invested, which is suitable for many real-time applications where resources are scarce. For example, the Heterogeneous Earliest Finish Time (HEFT) algorithm has been extensively adopted for task scheduling in heterogeneous systems. In more recent publications, there are known efforts that use hybrid methods that involve combining neural network-based approaches and heuristic policies to enhance scheduling performance in edge-cloud environments [8].

However, despite the growing body of research, existing surveys and reviews often consider heuristics as part of a broader taxonomy that includes optimization and AI-driven approaches. As a result, there is no dedicated and systematic survey that thoroughly examines workload scheduling, which is based on heuristics in edge cloud systems. This gap renders it hard to compare algorithms reliably, detect their tradeoffs, and trace their adaptability to edge-specific challenges like workload dynamics, energy constraints, and heterogeneous resources. This survey intends to present a systematic and focused survey of heuristic-based workload scheduling strategies in edge-cloud infrastructure. The goal of this paper is to:

- Categorize heuristic algorithms into three types: simple heuristics, metaheuristics, and hybrid heuristic approaches.
- Analyse and compare representative schemes on various performance metrics, i.e., latency, energy efficiency, scalability, adaptability, computation overhead, and ease of implementation.
- Determine the strengths, limitations, and tradeoffs of these strategies relative to the requirements of the edge cloud.
- Highlight research gaps and propose directions for developing adaptive, energy-aware, and lightweight heuristic scheduling models.
- Elucidate research voids and put forward future directions toward the development of adaptive, energy-aware, and lightweight heuristic scheduling models.

These are the subsequent research questions that have been raised towards the fulfillment of the stated purpose:

- RQ1: What are the key heuristic algorithms used for workload scheduling in edge cloud environments, and how can they be systematically categorized?
- RQ2: How do these heuristic methods perform with respect to critical performance metrics, including energy efficiency, latency, adaptability, scalability, computational cost, and implementation ease?
- RQ3: What are the tradeoffs and limitations of existing heuristic scheduling methods in addressing the dynamic, heterogeneous, and limited-resource nature of edge-cloud environments?
- RQ4: What are the open research challenges, and how can future heuristic strategies (simple, metaheuristic, or hybrid) be designed to improve workload scheduling in edge–cloud environments?

This paper fills the noted research gap by the following:

- Classification of heuristic scheduling approaches into three classes: simple heuristics, hybrid heuristic strategies, and metaheuristics.
- Detailed examination of representative algorithms, such as HEFT, Min-Min/Max-Min, ACO, PSO, and Greedy Resource-Aware heuristics.
- Comparison of the chosen methods according to latency, energy efficiency, scalability, adaptability, computational cost, and implementation ease.
- Research gap and future trend identification, focusing on adaptive, lightweight, and energy-aware heuristic models for future-edge cloud systems.

In contrast to earlier surveys, where heuristics are covered as a subsidiary section within expansive literature reviews of cloud, fog, or edge scheduling [8-13], this paper gives the primary committed and systematic survey solely dedicated to the topic of workload scheduling based on heuristics in the field of edge-cloud systems. This paper has originality in three folds:

- Scope specificity: While current reviews contemplate numerous scheduling approaches' families (optimization, AI, heuristics, etc.), this survey focuses only on heuristic algorithms (simple, metaheuristic, and hybrid heuristic strategies) and their application to the edge–cloud scenarios.
- Comparative depth: In contrast to earlier studies that

provide descriptive overviews, this paper delivers a structured comparative evaluation across six performance criteria: latency awareness, energy efficiency, scalability, adaptability, computational cost, and implementation ease.
- Gap addressing: The paper points to untouched challenges of heuristics in edge–cloud like adaptability for mobility, energy awareness for resource-limited nodes, and lightweight metaheuristic design, and suggests specific directions for future research.

The rest of this paper is organized as follows. Section II discusses the related work on workload scheduling of the edge cloud. Section III gives the required background and gives the formal definition of the scheduling issue. Section IV gives an overview of heuristic scheduling algorithms, and Section V outlines some representative algorithms and their real-world applications. Section VI gives a comparison of these algorithms. Section VII outlines the major challenge and indicates the future promising works, and finally, Section VIII summarizes the paper with the major insights.

2. Related Work

Extensive reviews have appeared on the topic of task scheduling under the paradigm of edge-cloud, and hundreds of survey and review papers have appeared evaluating the vast range of methods, including heuristic, metaheuristic, optimization-based, and machine learning-based approaches. However, these reviews are of varying scope and depth, and they seldom present an in-depth review of the heuristic-based approaches only. Paper [8] presented one of the earliest surveys of heuristic algorithms, but they only considered cloud computing infrastructures. Though it can be useful towards classical algorithms like Min–Min/Max–Min and HEFT, it fails to consider edge-cloud-related issues like heterogeneity, dynamic workloads, and latency-conscious applications. However, the latest studies have given increased focus on hybrid and AI-based scheduling. Paper [9] put forward the idea of a deep learning-based heuristic scheduler by hybridizing PSO and the Firefly Algorithm, improving the run of resource-intensive IoT tasks for cloud systems. Paper [10] examined IoT task scheduling under the scenarios of edge and fog computing and observed that classical metaheuristics like NSGA-II and NSGA-III usually suffer from the restriction of scalability and adaptability.

To overcome these shortcomings, they presented DRLIS, a scheduler based on Deep Reinforcement Learning, which showed better convergence and optimization ability than classical methods. Several surveys have considered broader viewpoints on scheduling. Paper [11] classified scheduling of the edge–cloud setting into heuristic, metaheuristic, and optimization-based families, and additionally considered Quality-of-Service (QoS) demands and fault tolerance. Paper [12] presented a comprehensive survey of resource scheduling in the edge computing setting and covered key topics of task offloading, resource scheduling, and resource provisioning approaches. Paper [13] covered heuristic scheduling of the fog computing setting, and presented greedy, priority-based, and nature-inspired heuristics, and hybrid heuristic approaches, and focused on energy and resource constraints.

Although these works are contributory, they generally consider the heuristics as only one of the numerous categories, but not the core of the study. None of these gives a systematic survey containing the followings: (i) classifying the heuristic strategies as simple, metaheuristics, and hybrid strategies; (ii) providing balanced comparison of the most significant performance aspects like latency, energy efficiency, scalability, adaptability, computational cost, and implementation complexity; and (iii) identifying outstanding issues inherent to heuristic scheduling for hybrid edge-cloud environments. This work fills this gap by narrowly focusing on heuristic-based workload scheduling. This paper offers a structured coverage and analysis of representative algorithms for key criteria, with a predominantly illustrative perspective on research directions, underpinning interest in adaptive, lightweight, or energy-aware heuristic models for edge cloud environments.

However, a big gap still prevails in the literature. Either they are:

- Concentrates on cloud or fog computing but fails to consider specific edge cloud integration constraints,
- Focus on AI-based approaches, but ignore classical heuristic approaches, or
- Give a descriptive summary rather than a systematic, comparative assessment for a variety of important performance criteria, such as latency awareness, scalability, energy efficiency, and implementation complexity.

This paper will fill this gap by being the first systematic literature review on workload scheduling with a basis on heuristic approaches conducted for edge clouds. Unlike other existing literature reviews, this paper will classify heuristic approaches into three categories, which include simple heuristics, metaheuristics, and hybrid heuristic strategies, in order to perform a fair analysis that will compare different approaches, with comprehensive quantitative insights taken from existing literature that will validate any qualitative assessments made. This specialized view will not only shed insight on where heuristics belong in hybrid architectures but will also point out some possible uses for heuristics with AI.

3. Background

3.1. Edge-Cloud Architecture Overview

Edge cloud computing forms a hybrid model that amalgamates two stages of computing infrastructure: edge nodes that are geographically close to data-generating resources, such as IoT sensors, as well as cloud servers that

have high compute capabilities. It is the type of data center hierarchy where data processing could be assigned for different sites, dependent on the applications' latency as well as resource requirements. It runs on the edge layer for latency and bandwidth-sensitive tasks, as well as in the cloud for computation-intensive processes, along with long-term data storing. It enhances the responsiveness, reduces data transmission overheads, and enhances scalability and fault tolerance in distributed systems [14].

3.2. Workload Characteristics

Workloads in edge-cloud environments are extremely heterogeneous, derived from various sources, including sensor types, mobile devices, cameras, and industrial robots. Workload differs in terms of computation intensity, latency sensitivity, data volume, and frequency. Also, the arrival of the workload is dynamic and unpredictable, making the scheduling decisions at run time critical. Schedulers need to consider different factors such as available resources, network states, as well as application priority, so as to guarantee service level objectives [15].

3.3. Scheduling Objectives

Scheduling workloads in Edge-Cloud requires considering as well as balancing several competing objectives:

- Latency Minimization: One of the key factors that makes edge computing more attractive for implementation is that it promotes latency minimization. Adjusting scheduling algorithms must aim at delegating more tasks near the data origin to minimize the round-trip time [16].
- Enhancing Energy Efficiency: Typically, edge devices are employed in a power-restricted environment, which may cause edge devices to run out of power. Scheduling algorithms must avoid excessive data transfer and avoid overloading power-restricted nodes [11].
- Load Balancing: In order to avoid resource loading on certain nodes, whereas others are underloaded, it is recommended that schedulers use equal task distribution. This helps in maintaining steady performance, improving system throughput, and reducing execution time [15].
- Quality of Service (QoS)/Quality of Experience (QoE): Almost every application requires certain guarantees, for example, on-time execution, guaranteed availability, and accurate data. For this purpose, scheduling needs to respect Service Level Agreements (SLAs), with a good QoE at the same time [15].

3.4. Limitations of Traditional Scheduling Approaches

Conventionally, schedulers employed in cloud computing, designed based upon static policies or deterministic approaches, are not suited for edge cloud since they react rigidly to changes occurring in real time, consuming high computation resources, having high overhead in resources with a large number of edge nodes as well as mobile devices, and being centralized, with high communication overhead and lack of fault tolerance. It indicates a need for a lightweight, adaptive, and decentralized scheduler, for which these approaches based upon heuristics had immense promise [11, 17].

4. Heuristic-Based Scheduling Approaches Classification

Heuristic scheduling algorithms are crucial for managing the dynamic nature of workloads that edge clouds exhibit. Figure 1 below shows that there are three categories under which heuristic scheduling algorithms are classified, including simple heuristics, metaheuristics, and hybrid heuristics.

4.1. Simple Heuristics

Simple heuristics refer to techniques that use rules for decision-making in relation to scheduling, based on a set of predefined rules. Simple heuristics are computationally optimal and perform best in real-time systems [18]. Simple heuristics include:

- First-Come, First-Served (FCFS): processes are allocated based on the order of arrival, regardless of system conditions, independent of individual needs. FCFS is simple. It may suffer from inefficiencies, resource wastage, and delays [19].
- Shortest Job First (SJF): This scheduling technique involves selecting processes that require the shortest execution time to minimize waiting time [19]. One major problem with this technique may lie in estimating task lengths.
- Min-Min and Max-Min: These schedules choose the tasks having minimum or maximum complete time, respectively, for optimization of overall scheduling efficiency [18].
- Round Robin (RR): Assigns time slices for individual use on a cycle basis in an attempt to be fair, but with the benefit of potentially less context-switching overhead [18].

4.2. Metaheuristics

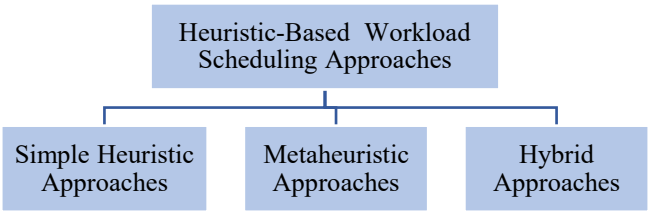

Fig. 1 Classification of heuristic scheduling approaches

Metaheuristics are higher-level procedures for the optimization of exploration of the solution space. They become employable for complex optimization problems, such as task scheduling for edge-cloud systems. Major techniques that fall into the category of metaheuristics are as follows.

- Genetic Algorithm (GA): Genetic selections provide inspiration for Genetic Algorithms (GAs) owing to the application of mechanisms such as selection, crossover, and mutation by genetic algorithms for evolving a solution over several generations. Though genetic algorithms are helpful in addressing much larger problem sizes, they consume a lot of computational power [20].
- Particle Swarm Optimization (PSO): Its Individuals are modeled as particles that move in a solution space, where the change in the position of a particle occurs based on both individual and global past experience. It is clear that both simplicity and fast convergence speed are built into PSO models [20].
- Ant Colony Optimization (ACO): This method rests on the premise of simulating ant foraging behavior. Pheromone trails are employed as a technique for building solutions that address combinatorial problems in artificial ant foraging behavior studies [21].
- Simulated Annealing (SA): Its inspiration comes from the heating and cooling treatment of metals. The model makes suboptimal decisions to avoid reaching a local minimum, which facilitates a more efficient resolution of challenging scheduling problems [20].

Recent studies have utilized various metaheuristic techniques to enhance scheduling efficiency in cloud and edge environments. For example, one study employed ACO to compute a fitness function and balance multiple scheduling objectives, demonstrating its usefulness in addressing complex problems. Another study adapted the firefly algorithm for workflow scheduling across cloud–edge systems, making improvements in both makespan and cost [21, 22].

4.3. Hybrid Heuristics

Hybrid heuristics are a combination of various heuristic or metaheuristic strategies with the goal of benefiting from their respective strengths and mitigating their respective weaknesses. Hybrid strategies have been employed to address the complex task scheduling problem within edge–cloud systems.

As an example, a hybrid metaheuristic algorithm was proposed for IoT service placement with the optimization of such factors as makespan, cost, and energy usage [20]. While metaheuristics and hybrid approaches themselves may not be AI-driven, they belong to the family of heuristic-based approaches. Incorporating machine learning or neural components into these systems, however, is the path toward AI-driven scheduling.

5. Representative Heuristic Approaches

This section reviews heuristic and metaheuristic methods used for workload scheduling in edge-cloud computing. The subsequent subsections describe the selection process and the reasoning behind the chosen algorithms.

5.1. Methodology for Algorithm Selection

The selection of representative algorithms followed a structured process designed to ensure relevance, diversity, and coverage across the main categories of heuristic-based scheduling. The process was guided by a review of peer-reviewed studies published between 2018 and 2025, retrieved from major digital libraries.

Only algorithms that were explicitly applied to edge, fog, or edge–cloud scheduling were taken into account. Additionally, it was ensured that each study evaluated:

- Proposed or analyzed a heuristic or hybrid heuristic scheduling algorithm.
- Addressed at least one key performance objective, including latency, energy efficiency, scalability, or flexibility.
- Provided comparable or empirical results that enable a systematic assessment based on certain criteria.
- Contained enough information about the implementation or concept for analysis purposes.
- Addressed at least one key performance objective, like latency, energy efficiency, scalability, or flexibility.

Studies focusing solely on exact optimization algorithms, deep learning, or cloud scheduling were not considered, as were domain-specific solutions that are not generally applicable.

This ensured that generally applicable heuristics for edge cloud environments are encompassed by the selected algorithms.

5.2. Justification of Selection

On the basis of this, we chose five algorithms that represent:

- HEFT: A classic static heuristic widely employed in heterogeneous and edge platforms.
- Min-Min, Max-Min: Lightweight heuristics that are still benchmark algorithms in heterogeneous scheduling problems.
- ACO: Metaheuristic based on swarm intelligence, suited for multi-objective optimization for edge-cloud systems.
- PSO: A metaheuristic valued for balancing global and local search, adaptable to dynamic conditions.
- GRAH: Modern heuristics explicitly designed for dynamic, resource-constrained edge–cloud environments.

Though other techniques, including Genetic Algorithms (GA), Cuckoo Search (CS), Simulated Annealing (SA), Artificial Bee Colony (ABC), Tabu Search (TS), or Grey Wolf Optimizer (GWO), among others, also satisfy the criteria for inclusion, we restrict this work to this set for clarity. All of them encompass the diversity of heuristic approaches:

- Simple heuristics: HEFT, Min-Min, and Max-Min.
- Metaheuristics: ACO and PSO.
- Hybrid/resource-aware heuristics: GRAH.

This representative set allows for a systematic evaluation across latency, energy efficiency, scalability, adaptability, computational cost, and implementation ease.

5.3. Heterogeneous Earliest Finish Time (HEFT)

The HEFT algorithm, a classic static algorithm, was first developed for task scheduling in heterogeneous computing environments. This algorithm gives more priority to tasks with upward ranks, which assign a task to a processor in such a way that it gets completed as early as possible. Because of its effectiveness in minimizing makespan with high efficiency, HEFT has been largely used in edge cloud environments [18]. But the major drawback of this algorithm was that it was a static algorithm, which neither offered dynamic rescheduling [23] nor considered power consumption, which was a concern for edge clouds [24].

5.4. Ant Colony Optimization (ACO)

ACO is a metaheuristic that was inspired by ant foraging behavior, in which virtual ants probabilistically search task allocations with the aid of pheromone trails that stand for the quality of previous solutions found. Its main strength resides in improving multi-objective scheduling in a dynamic edge cloud environment, thanks to its flexibility in adapting to changing network conditions [22]. In contrast, ACO also has some weaknesses. For instance, its most elementary form remains capable of addressing just a relatively small problem instance, such as the Traveling Salesman Problem with a small number of graph vertices, with high memory complexity, usually $O(n^2)$ [25]. In addition, ACO is parameter-sensitive, not scalable for large-scale data, and not suitable for real-time deployment [26].

5.5. Particle Swarm Optimization (PSO)

PSO approaches task scheduling as a problem involving swarm intelligence, with particles indicating possible solutions that move in a search space with an understanding of both personal and social learning [20]. PSO has already been validated for edge-cloud architectures, including task scheduling with latency and energy considerations, as well as for large-scale or dynamically changing workloads [18]. One major reason for its widespread adoption is its ease of use while handling non-linear, multi-modal functions, as well as its flexibility in being applicable to different scheduling, resource assignment, and feature selection problems [24]. Nevertheless, its computational complexity varies with respect to its number of particles, dimension, and iterations, having a time complexity of $O(n \times d \times t)$ [27].

5.6. Min-Min and Max-Min Heuristics

These are common but straightforward list scheduling methods:

- Min-Min algorithm selects a task with minimum execution time for a resource, making it useful in scenarios that consist of more smaller tasks, as it helps in reducing waiting time as well as speedy execution [28, 29]. However, its major drawback might result in large tasks being delayed, as they are usually given lower priorities, which leads to a queue of larger tasks being formed [29].
- Max-Min: This algorithm chooses a task with maximum minimum completion time, which will be allocated to a node that completes it first. It prevents larger tasks from being perpetually blocked by smaller tasks, making it more suitable for a heterogeneous environment with nodes having different capabilities, with a variety of task resource needs. However, it faces inefficiencies if there are mostly smaller tasks, which may get excessively delayed [29].

Such techniques are most useful in situations where workload sizes and complexities are varied, thus being applicable for heterogeneous edge cloud settings [18]. In most cases, such algorithms rely on static decision-making, meaning they are not able to adjust to changes in task rates, availability, or system load [29]. Besides, such heuristics may rely on static task execution time estimates, independent of device power state or energy, hence being ineffective, for instance, in settings that require task execution under certain power levels.

5.7. Greedy Resource-Aware Heuristics

Some research has introduced custom greedy algorithms that schedule the tasks according to system measurements like the load on the CPU, the usage of memory, or energy levels. Some heuristics, for example, schedule tasks on the closest node with enough resources for the purpose of reducing the latency of data transfer. These approaches are lightweight and suitable for real-time edge scheduling [16].

Greedy Resource-Aware Heuristics (GRAH) are more energy-efficient due to their dynamic awareness of node conditions and adaptive, proximity-aware decisions, making them ideal for energy-sensitive fog and edge environments. In general, they are valued for their simplicity and speed, as they base decisions solely on the current system state without considering future tasks or conditions [29]. Yet, in a dynamically changing context such as fog computing, they may find it difficult to produce overall globally optimal results since they concentrate more upon immediate gains, with no attention towards changing system conditions [29]. This notwithstanding, they perform best for real-time scheduling and straightforward scenarios, providing practical and efficient solutions in many edge-cloud contexts [13].

All such heuristic strategies represent a reflection of different approaches that work towards task scheduling optimization in edge cloud computing. Though they present

some individual merits as well as pitfalls, they collectively form a major development phase for more efficient scheduling approaches.

6. Comparative Analysis of Heuristic Algorithms

This section presents a comparative study of different workload scheduling algorithms based on heuristics for edge clouds. This will help to assess different strengths and weaknesses that exist between various algorithms with respect to meeting edge cloud requirements. The comparison will be done with respect to some key factors that are important in edge-cloud systems, such as latency awareness, energy efficiency, scalability, adaptability, computational cost, and simplicity of implementation [18, 30].

6.1. Evaluation Criteria

Table 1 defines six evaluation criteria used to assess the selected algorithms.

Table 1. Comparison criteria

Criterion	Abbreviation	Description
Latency Awareness	LA	Ability to minimize task response time and satisfy real-time performance constraints [30].
Energy Efficiency	EE	Capacity to reduce and optimize energy consumption, particularly at resource-constrained edge nodes [28].
Scalability	S	Capability to maintain performance as the number of tasks or nodes increases [30].
Adaptability	A	Flexibility to operate effectively under dynamic workloads and changing network conditions [30].
Computational Cost	CC	Overhead imposed by the algorithm itself (low cost is preferred at the edge [28].
Implementation Ease	IE	Practicality and simplicity of integration into real-world systems

6.2. Systematic Summary of Heuristic Scheduling Methods

This subsection gives a structured synthesis of the representative heuristic algorithms described in Section 5, summing up their most important performance criteria.

All algorithms are evaluated with respect to six key factors: Latency Awareness, Energy Efficiency, Scalability, Adaptability, Computational Cost, and Implementation Ease. Table 2 synthesizes insights gleaned from more recent studies, involving a set of qualitative assessments that are backed by references with a description of algorithm behavior under a hybrid edge-cloud environment.

This enables a clear understanding of how various classes of heuristics, including simple, metaheuristic, or hybrid, handle the multi-objective nature of scheduling, besides illuminating typical weaknesses related to being static or having high computational overhead. For a complementary perspective to this summary based on scientific evidence, Table 3 below provides a straightforward comparative table, which translates descriptive results into a standardized scale of ratings (very low, low, moderate, good, very good) for a more comparable result between algorithms. This standardized scoring system will form a basis for analysis in Section 6.4.

6.3. Comparison of Scheduling Algorithms in Edge-Cloud Systems

In this section, each algorithm has been assigned a value on a scale from very low to very good per criterion, where very low represents the weakest performance and very good indicates the strongest.

This scoring approach provides a structured and consistent way to assess the strengths and limitations of each algorithm across key factors relevant to hybrid edge–cloud computing, and it is based on evidence reported in the literature. Table 3 summarizes the performance ratings.

Table 2. Systematic comparison of heuristic algorithms across key performance criteria in edge–cloud scheduling (based on Section 5)

Algorithm	LA	EE	S	A	CC	IE
HEFT	Minimizes makespan effectively (moderate) [18]	Ignores energy (low) [24]	Well-suited for heterogeneous systems (good) [18]	Static, no rescheduling (very low) [23]	Ranking-based, efficient (good) [18]	Simple & widely adopted (good) [18]
ACO	Multi-objective optimization reduces	Energy-aware variants exist	Limited scalability (good) [25]	Adapts to changing environments (very good)	High $O(n^2)$ complexity (very low) [25]	Parameter-sensitive, complex (low) [26]

	latency (very good) [22]	(very good) [22]		[22]		
PSO	Reduces task delays (reasonable) [18]	Efficient allocation (good) [18]	Large-scale applicability (very good) [24]	Moderately adaptive (moderate) [27]	O(n × d × t) cost (moderate) [27]	Relatively simple (reasonable) [20]
Min-Min/Max-Min	Handles small/large tasks but unbalanced (moderate) [28]	No energy model (low) [29]	Useful in heterogeneity (moderate) [29]	Static & rigid (very low) [29]	Very lightweight (very good) [28]	Extremely easy (very good) [28]
GRAH	Proximity-aware scheduling (very good) [16]	Energy-aware & dynamic (very good) [29]	Scales to large real-time systems (very good) [13]	Adapts to state but not long-term (moderate) [29]	Lightweight & efficient (good) [16]	Simple but case-specific (moderate) [13]

Table 3. Comparative evaluation of heuristic scheduling algorithms in edge–cloud systems

Algorithm	LA	EE	S	A	CC	IE
HEFT	moderate	low	good	Very low	good	good
ACO	Very good	Very good	good	Very good	Very low	low
PSO	good	good	Very good	moderate	moderate	good
Min-Min / Max-Min	moderate	low	moderate	Very low	Very good	Very good
GRAH	very good	Very good	Very good	moderate	good	moderate

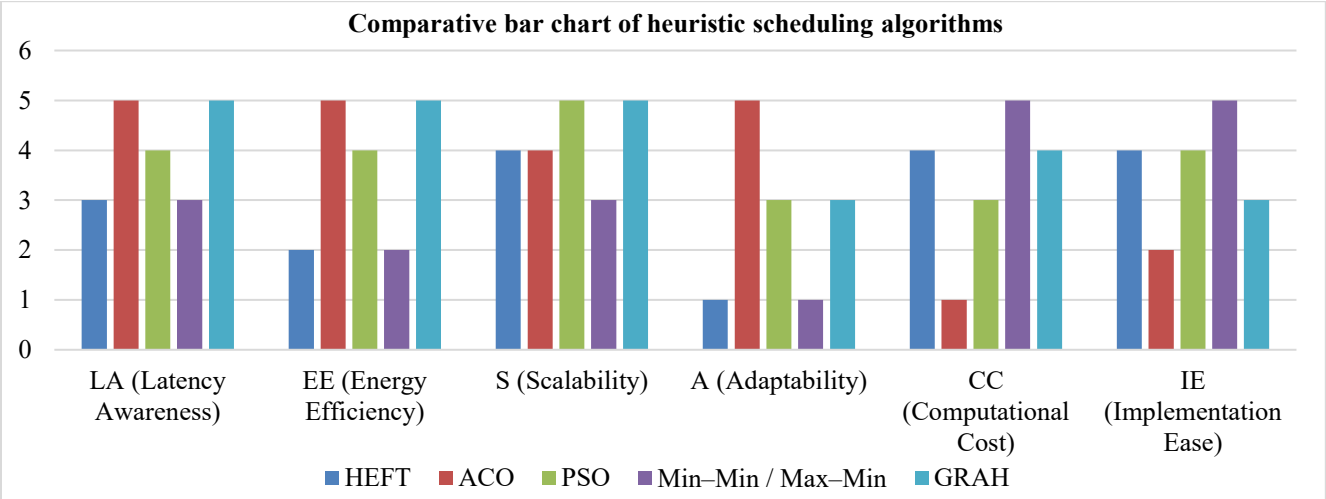

Fig. 2 Comparative bar chart of heuristic scheduling algorithms through six performance metrics

6.4. Graphical Representation of Comparative Results

For improving clarity as well as meeting visual analysis needs, Figure 2 plots a bar graph for a comparative result of Table 3. This graphical representation helps in speedy identification of tradeoffs in addition to easy understanding of strengths relevant to each algorithm related to hybrid edge clouds, as indicated by six criteria with an ordinal scale rating of 1–5, starting with Very low as 1 and culminating with Very good as 5. As shown in Figure 2, both GRAH and ACO have high performance with respect to latency, energy, and scalability, whereas PSO shows high scalability with balanced results. On the other hand, although HEFT and Min-Min/Max-Min remain simple with low computational overhead, they are non-adaptive and lack energy-awareness in dynamic environments.

6.5. Observations and Insights

Based on the comparison, Greedy Resource-Aware Heuristics demonstrate the most balanced performance across all criteria, combining strong latency awareness, energy

efficiency, and scalability, while maintaining a comparatively low computational cost and moderate implementation complexity. ACO performs well on latency and energy optimization and can adapt easily to different conditions. However, due to a high level of computational complexity and a lower ease of implementation, it may not be appropriate for edge settings. PSO performs with a stable level of quality; more specifically, energy efficiency and good scalability are achieved to a great extent. But for proper operation under varying conditions, precise tuning of PSO algorithms is needed. HEFT can also be applicable for handling static workloads due to acceptable computational efficiency and simple implementation. Its inadequacies lie in its lack of awareness about energy consumption and flexibility. Lastly, Min-Min and Max-Min are the least complicated out of the four algorithms presented. They achieved the best results in terms of both computational efficiency and implementation complexity. They received the lowest results for both flexibility and power efficiency.

In conclusion, Greedy Resource-Aware Heuristics provide a fair optimization solution for a hybrid edge cloud computing system, with notable latency reduction, energy efficiency, and scalability with a minimal overhead cost. It shows that ACO performs well in terms of its optimizing capability, but its effectiveness will be affected since it requires high computational demands. On the other hand, PSO shows a fair level of efficiency with moderate flexibility. It was established that both HEFT and Min-Min/Max-Min perform well for non-dynamic resources, but they lack adaptability as well as energy awareness needed for a dynamic edge cloud scenario.

6.6. Cross-Criterion Insights and Tradeoffs

Table 2 and Table 3, which indicate the comparison results, indicate definite patterns as well as certain compromises between criteria in the selected heuristic algorithms. Heuristics that perform best for a certain criterion may compromise on other criteria, which indicates that no algorithm performs best for all criteria.

- Latency vs. Computational Cost: Algorithms such as ACO and GRAH have been shown to achieve significant latency reduction due to their adaptive and search-based design. However, this enhancement comes with an increased computational overhead. Conversely, simple heuristics such as HEFT and Min–Min are computationally efficient but demonstrate reduced efficacy in minimizing delay under dynamic workloads.
- Energy Efficiency vs. Ease of Implementation: It has been proven that the algorithms that integrate dynamic or energy-aware mechanisms, such as ACO, PSO, or GRAH, perform well in terms of energy efficiency. But such algorithms require more complicated parameter tuning or system feedback, which may result in adverse effects on deployment ease.
- Adaptability vs. Simplicity: On one side, metaheuristics and hybrid approaches are described as being flexible, adapting to workload changes, but they also bring complexity with regard to design. On the other hand, simple heuristics, being easy to implement, fail to handle changes in real time effectively.
- Scalability and Stability: PSO maintains stable performance as the system size grows, proving effective for heterogeneous or federated edge-cloud systems. Despite its robust performance in smaller-scale settings, ACO may be susceptible to parameter sensitivity as system size increases.
- Scalability and Stability: Both PSO and GRAH show stable outcomes in large-scale scenarios, so they could also be appropriate in heterogeneous or federated edge-cloud infrastructures. However, in small-scale scenarios, ACO could yield good results, but in large-scale scenarios, it could also face stability issues regarding parameters.

As an aggregate, these tradeoffs imply that criterion-driven selection for heuristics is required in the sense that static workload scenarios could be better suited for either HEFT or Min–Min, while other scenarios could be appropriate for either ACO, PSO, or GRAH algorithms. The aforementioned interpretation transcends specific scores in understanding each type of heuristic in relation to its operational scenarios in edge-cloud architecture.

6.7. Comparison with Existing Surveys

Almost all the previous literature reviews on workload scheduling in cloud, fog, or edge setups [3-8] investigated different paradigms in terms of algorithms, including heuristics, meta-heuristics, and AI-based approaches, with a wide-ranging taxonomy. Still, they discussed heuristics in a supporting role only, lacking a separate performance-centric comparison in their scope. The current literature review is different in the sense that it concentrates only on heuristics for workload scheduling in edge-cloud hybrid setups, with an organized synthesis on six different criteria: latency, energy efficiency, scalability, adaptability, computational cost, and implementation ease.

7. Challenges and Future Directions

Despite the significant advancements in edge-cloud computing using heuristics for the scheduling approaches, there remain some open issues yet to be addressed. The following issues represent rich areas for research with respect to improving adaptability, efficiency, and scalability in managing edge cloud workloads.

7.1. Key Challenges

- Less adaptability in dynamic Environments: Heuristics that belong to the classical category, for instance, HEFT, Min–Min, and Max–Min, were traditionally designed for static environments only. Edge-cloud computing, on the

other hand, is a dynamic environment with fluctuating workloads, varying levels of network connectivity, changing workloads, and variable resource availability, limiting its effectiveness in such settings.

- Suboptimal global scheduling from local decisions: Greedy heuristics are appreciated for their simplicity and low computational complexity, but they commonly make locally optimal decisions based only on their states in the nodes in question. In large-scale edge-cloud systems, locally made choices could potentially cause suboptimal global systems' performance in terms of unbalanced load distribution, resource underutilization, and starvation in some nodes.
- Higher computational cost in Metaheuristics: Metaheuristics like PSO and ACO offer strong flexibility and multi-objective optimization capabilities, but typically come with high computational and memory demands. This makes them unsuitable for deployment on lightweight, resource-constrained edge nodes-especially in real-time or latency-sensitive applications-unless adapted or hybridized for efficiency.

7.2. Future Research Directions

- Design of Adaptive Heuristics: In future research, designing context-aware and adaptive heuristics for handling varying levels of workloads and availability nodes while incorporating user mobility into the design would be beneficial to enhance the responsiveness and robustness of edge cloud systems.
- Lightweight Metaheuristic Variants: Research should optimize and simplify metaheuristics like ACO and PSO by including techniques like parameter pruning or tuning with a set of greedy rules. This may help retain performance with reduced overhead costs for edge device deployment.
- Hybrid Scheduling Approaches: One of the promising areas that can be explored is the design of hybrid schedulers with multiple objectives by combining the power of diverse heuristic algorithms; for instance, a combination of Min-Min with PSO or HEFT with Greedy Priorities.
- Energy and SLA-Aware Extensions: Enhancing traditional heuristics with capabilities like energy-awareness, battery management, and Service Level Agreement (SLA) compliance would significantly improve their practicality and long-term viability in real-world edge computing environments.

In conclusion, despite the immense potential of heuristic algorithms for scheduling in future edge cloud designs, real-world challenges such as adaptability, scalability, and resource awareness necessitate further research.

This includes developing lightweight, adaptive algorithms, modifying metaheuristics for edge environments, and integrating energy and SLA awareness into existing heuristic frameworks. This precise guidance creates a distinct heuristic structure that can adapt and schedule according to the complex performance demands of emerging edge–cloud systems.

7.3. Ethical and Societal Considerations

With the increasing complexity of edge–cloud computing infrastructure, ethical concerns become increasingly significant. Scheduling tasks influences where and how sensitive data-such as personal or medical information—is processed, thereby raising concerns about privacy and security. It is essential that heuristic algorithms adopt secure data-handling protocols to uphold user trust and meet regulatory standards.

An additional ethical dimension pertains to energy consumption and sustainability. It should be noted that edge devices and data centers contribute to carbon emissions; therefore, the design of energy-efficient scheduling is both a technical necessity and an ethical obligation toward environmentally responsible computing.

In addition, heuristic and hybrid scheduling algorithms might unintentionally inject bias or unfairness in resource allocation, benefiting some users, devices, or regions at the expense of others. It is imperative that scheduling policies be transparent and auditable, particularly in mission-critical domains such as healthcare and autonomous transportation.

Finally, as edge-cloud infrastructures operate across jurisdictions, schedulers must comply with regulatory and compliance frameworks, among which are the General Data Protection Regulation (GDPR) or the Health Insurance Portability and Accountability Act (HIPAA). To address these concerns, interdisciplinary research is necessary. This research must combine technical efficiency with ethical accountability. The objective is to ensure trustworthy workload orchestration in real-world deployments.

8. Conclusion

Heuristic workload scheduling is among the most critical performance optimization aspects in edge-cloud computing platforms. Due to the increasing requirements for ultra-low latency, energy efficiency, and scalability, the development of lightweight yet efficient scheduling algorithms has become increasingly challenging. This paper presents a structured review of heuristic scheduling strategies. The heuristic scheduling strategies are categorized into three classes: simple, metaheuristic, and hybrid. The paper also analyzes several representative algorithms, including HEFT, ACO, PSO, Min–Min/Max–Min, and GRAH. These algorithms are analyzed across six performance criteria: latency, energy consumption, scalability, adaptability, computational cost, and implementation ease. The comparative results indicate that GRAH achieves the most balanced performance, while ACO and PSO offer strong adaptability and energy

optimization at the expense of higher computational cost. In contrast, classical heuristics such as HEFT and Min–Min remain practical for static workloads but are less effective under dynamic or resource-constrained conditions. Despite these advantages, heuristic-based methods still face persistent challenges in real-time adaptability, multi-objective tradeoffs, and energy- and SLA-awareness within heterogeneous systems.

To address these gaps, future research should:

- Develop context-aware and adaptive heuristic frameworks capable of responding dynamically to workload fluctuations, mobility, and resource variability in real time.
- Design lightweight variants of metaheuristics such as ACO and PSO by applying parameter tuning, dimensionality reduction, or integration with greedy strategies to make them feasible for resource-constrained edge nodes.
- Integrate heuristics with AI-driven mechanisms, such as deep reinforcement learning, to enable predictive and self-optimizing scheduling that anticipates changes rather than merely reacting to them.
- Incorporate energy-, SLA-, and fairness-aware extensions to ensure not only performance efficiency but also sustainability and equitable task distribution across distributed infrastructures.
- Establish standardized benchmarking and simulation platforms for evaluating heuristic models under realistic hybrid edge–cloud scenarios, enabling consistent comparison and reproducibility.

By pursuing these directions, future work can transform heuristic scheduling into a more adaptive, sustainable, and intelligent discipline, bridging the gap between lightweight decision-making and context-aware orchestration. This evolution will pave the way for robust, ethically responsible, and high-performance workload management of emerging edge–cloud systems.

References

[1] Kanagarla Krishna Prasanth Bra, "Edge Computing and Analytics for IoT Devices: Enhancing Real-Time Decision Making in Smart Environments," *International Journal for Multidisciplinary Research (IJFMR)*, vol. 6, no. 5, pp. 1-9, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[2] Nasir Abbas et al., "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450-465, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[3] Mahsa Paknejad et al., "A Reliable and Efficient 5G Vehicular MEC: Guaranteed Task Completion with Minimal Latency," *2025 IEEE International Conference on Communications Workshops (ICC Workshops)*, Montreal, QC, Canada, pp. 566-571, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[4] Hossein Ahmadvand, and Fouzhan Foroutan, "Latency and Privacy-Aware Resource Allocation in Vehicular Edge Computing," *arXiv Preprint*, pp. 1-6, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] Alexandru Rancea, Ionut Anghel, and Tudor Cioara, "Edge Computing in Healthcare: Innovations, Opportunities, and Challenges," *Future Internet*, vol. 16, no. 9, no. 1-28, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[6] Yazeed Yasin Ghadi et al., "Enhancing Patient Healthcare with Mobile Edge Computing and 5G: Challenges and Solutions for Secure Online Health Tools," *Journal of Cloud Computing*, vol. 13, no. 1, pp. 1-13, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Dinesh Sahu et al., "Optimizing Energy and Latency in Edge Computing Through a Boltzmann Driven Bayesian Framework for Adaptive Resource Scheduling," *Scientific Reports*, vol. 15, no. 1, pp. 1-26, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[8] Nasim Soltani, Behzad Soleimani, and Behrang Barekatain, "Heuristic Algorithms for Task Scheduling in Cloud Computing: A Survey," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 9, no. 8, pp. 16-22, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[9] Harshala Shingne, and R. Shriram, "Heuristic Deep Learning Scheduling in Cloud for Resource-Intensive Internet of Things Systems," *Computers and Electrical Engineering*, vol. 108, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Zhiyu Wang et al., "Deep Reinforcement Learning-based Scheduling for Optimizing System Load and Response Time in Edge and Fog Computing Environments," *Future Generation Computer Systems*, vol. 152, pp. 55-69, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[11] Hassan Asghar, and Eun-Sung Jung, "A Survey on Scheduling Techniques in the Edge Cloud: Issues, Challenges and Future Directions," *arXiv Preprint*, pp. 1-19, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[12] Quyuan Luo et al., "Resource Scheduling in Edge Computing: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 4, pp. 2131-2165, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[13] Deafallah Alsadie, "Advancements in Heuristic Task Scheduling for IoT Applications in Fog-Cloud Computing: Challenges and Prospects," *PeerJ Computer Science*, vol. 10, pp. 1-58, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[14] Olha Boiko et al., "Edge-Cloud Architectures for Hybrid Energy Management Systems: A Comprehensive Review," *IEEE Sensors Journal*, vol. 24, no. 10, pp. 15748-15772, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[15] Ahmed A. Ismail et al., "A Survey on Resource Scheduling Approaches in Multi-Access Edge Computing Environment: A Deep Reinforcement Learning Study," *Cluster Computing*, vol. 28, no. 3, pp. 1-45, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[16] Zaiwar Ali et al., "A Comprehensive Utility Function for Resource Allocation in Mobile Edge Computing," *Computers, Materials and Continua*, vol. 66, no. 2, pp. 1461-1477, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] Amin Avan, Akramul Azim, and Qusay H. Mahmoud, "A State-of-the-Art Review of Task Scheduling for Edge Computing: A Delay-Sensitive Application Perspective," *Electronics*, vol. 12, no. 12, pp. 1-27, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] Nisha Devi et al., "A Systematic Literature Review for Load Balancing and Task Scheduling Techniques in Cloud Computing," *Artificial Intelligence Review*, vol. 57, no. 10, pp. 1-63, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[19] Saydul Akbar Murad et al., "SG-PBFS: Shortest Gap-Priority Based Fair Scheduling Technique for Job Scheduling in Cloud Environment," *Future Generation Computer Systems*, vol. 150, pp. 232-242, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[20] Hemant Kumar Apat et al., "A Hybrid Meta-Heuristic Algorithm for Multi-Objective IoT Service Placement in Fog Computing Environments," *Decision Analytics Journal*, vol. 10, pp. 1-17, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Mustafa Ibrahim Khaleel et al., "Combinatorial Metaheuristic Methods to Optimize the Scheduling of Scientific Workflows in Green DVFS-Enabled Edge-Cloud Computing," *Alexandria Engineering Journal*, vol. 86, pp. 458-470, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[22] Nebojsa Bacanin et al., "Modified Firefly Algorithm for Workflow Scheduling in Cloud-Edge Environment," *Neural Computing and Applications*, vol. 34, no. 11, pp. 9043-9068, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[23] Ashutosh Shankar, and Astha Kumari, "QoS-aware Scheduling of Periodic Real-time Task Graphs on Heterogeneous Pre-occupied MECs," *arXiv Preprint*, pp. 1-9, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[24] Dinesh Sahu et al., "Beyond Boundaries A Hybrid Cellular Potts and Particle Swarm Optimization Model for Energy and Latency Optimization in Edge Computing," *Scientific Reports*, vol. 15, no. 1, pp. 1-22, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[25] Rafał Skinderowicz, "Improving Ant Colony Optimization Efficiency for Solving Large TSP Instances," *Applied Soft Computing*, vol. 120, pp. 1-28, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[26] Renjbar Sh. Othman, and Ibrahim Mahmood Ibrahim, "A Review of Exploring Recent Advances in Ant Colony Optimization: Applications and Improvements," *International Journal of Scientific World*, vol. 11, no. 1, pp. 114-122, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[27] Shabariram C. Palaniappan, and Priya P. Ponnuswamy, "Task Offloading in Edge Computing Using Integrated Particle Swarm Optimization and Genetic Algorithm," *Advances in Science and Technology Research Journal*, vol. 19, no. 1, pp. 371-380, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[28] Farida Siddiqi Prity, Md. Hasan Gazi, and K.M. Aslam Uddin, "A Review of Task Scheduling in Cloud Computing Based on Nature-Inspired Optimization Algorithm," *Cluster Computing*, vol. 26, no. 5, pp. 3037-3067, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[29] Kaushik Sathupadi, "Comparative Analysis of Heuristic and Ai-Based Task Scheduling Algorithms in Fog Computing: Evaluating Latency, Energy Efficiency, and Scalability in Dynamic, Heterogeneous Environments," *Quarterly Journal of Emerging Technologies and Innovations*, vol. 5, no. 1, pp. 23-40, 2020. [Google Scholar] [Publisher Link]

[30] Sheikh Umar Mushtaq, Sophiya Sheikh, and Sheikh Mohammad Idrees, "Enhanced Priority Based Task Scheduling with Integrated Fault Tolerance in Distributed Systems," *International Journal of Cognitive Computing in Engineering*, vol. 6, pp. 152-169, 2025. [CrossRef] [Google Scholar] [Publisher Link]