Cluster Computing
https://doi.org/10.1007/s10586-024-04442-2

AI-based & heuristic workflow scheduling in cloud and fog computing: a systematic review

Navid Khaledian[1] · Marcus Voelp[1] · Sadoon Azizi[2] · Mirsaeid Hosseini Shirvani[3]

Received: 24 December 2023 / Revised: 28 February 2024 / Accepted: 18 March 2024

Abstract

Fog and cloud computing are emerging paradigms that enable distributed and scalable data processing and analysis. However, these paradigms also pose significant challenges for workflow scheduling and assigning related tasks or jobs to available resources. Resources in fog and cloud environments are heterogeneous, dynamic, and uncertain, requiring efficient scheduling algorithms to optimize costs and latency and to handle faults for better performance. This paper aims to comprehensively survey existing workflow scheduling techniques for fog and cloud environments and their essential challenges. We analyzed 82 related papers published recently in reputable journals. We propose a subjective taxonomy that categorizes the critical difficulties in existing work to achieve this goal. Then, we present a systematic overview of existing workflow scheduling techniques for fog and cloud environments, along with their benefits and drawbacks. We also analyze different workflow scheduling techniques for various criteria, such as performance, costs, reliability, scalability, and security. The outcomes reveal that 25% of the scheduling algorithms use heuristic-based mechanisms, and 75% use different Artificial Intelligence (AI) based and parametric modelling methods. Makespan is the most significant parameter addressed in most articles. This survey article highlights potentials and limitations that can pave the way for further processing or enhancing existing techniques for interested researchers.

Keywords Fog computing · Cloud computing · Workflow scheduling · Artificial intelligence · Systematic survey

Abbreviations

AI	Artificial intelligence
HEFT	Heterogeneous earliest finish time algorithm
GA	Genetic algorithm
KHA	Krill herd algorithm
EC2	Elastic compute cloud
AHA	Artificial hummingbird algorithm
ML	Machine learning
DVFS	Dynamic voltage and frequency scaling
IoT	Internet of things
SMO	Spider monkey optimization
LE	Low energy
VM	Virtual machines
MCC	Mobile cloud computing
VCPU	Virtual CPU
MEC	Mobile edge computing
PPR	Performance-to-power ratio
DAG	Directed acyclic graph
FOA	Fruit fly optimization
T	Set of tasks
FFA	Farmland fertility algorithm
A	Set of arcs
MHDA	Multi-objective hybrid dragonfly algorithm
QoS	Quality of service
SOS	Symbiotic organisms search
EPC	Event-driven process chain
GOA	Grasshopper optimization algorithm
PSO	Particle swarm optimization

✉ Navid Khaledian
navid.khaledian@uni.lu

Marcus Voelp
Marcus.voelp@uni.lu

Sadoon Azizi
S.azizi@uok.ac.ir

Mirsaeid Hosseini Shirvani
mirsaeid_hosseini@iausari.ac.ir

[1] Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Esch-sur-Alzette, Luxembourg

[2] Department of Computer Engineering and IT, University of Kurdistan, Sanandaj, Iran

[3] Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

Published online: 08 May 2024

CA	Cultural algorithm
FRM	Fog computing resource management
EPO	Emperor penguin optimizer
LCS	Longest common subsequence
HMM	Hidden Markov model
LOA	Lion optimization algorithm
HHO	Harris hawk optimization
NSGA	Non-dominated sorting genetic algorithm
OSA	Owl search algorithm
DDQN	Double deep Q-network
SDN	Software-defined network
CNN	Convolutional neural networks
GGCN	Gated graph convolution network
MLIP	Mixed integer linear programming
ILP	Integer linear programming
DEWS	Deadline energy-aware workflow scheduling
MTGP	Multi-tree genetic programming

1 Introduction

Workflow scheduling is a crucial process in many domains and applications involving complex and interdependent tasks that must be executed on distributed and heterogeneous resources in cloud and fog environments. Workflow scheduling aims to optimize various objectives and criteria, such as execution time, costs, energy consumption, quality of service, and fault tolerance. However, workflow scheduling is also a challenging problem that requires sophisticated algorithms and techniques to cope with the dynamic and uncertain nature of resources and workloads. Workflow scheduling in cloud-fog environments presents unique challenges due to their hierarchical nature and complex tradeoffs [1]. Fog provides low-latency computing capabilities closer to the network edge and IoT devices [2].

In contrast, the cloud offers high-capacity computing in centralized data centers. Effective scheduling must balance the tradeoffs between latency-sensitive fog resources and computation-intensive cloud resources to meet application requirements [3]. There are also multiple stakeholders like subscribers, infrastructure providers, platform operators, and application providers whose interests must be accommodated. The optimal usage of cloud-fog resources is evaluated based on cost and energy consumption. Since various stakeholders exist in the system, such as providers and subscribers, the optimal use of cloud-fog resources is assessed based on cost, energy consumption, and other related metrics.

To this end, several single- or multi-objective scheduling algorithms were extended in literature to favor users, providers, or balance between both. Therefore, some research is focused on these parameters and challenges [4]. For example, the cost issue involves various parameters such as processing power, system performance, system capacity, processing time, operation delay, and communication costs. A comprehensive study and evaluation of task scheduling methods is essential in the fog environment to achieve the mentioned objectives. According to the mentioned issues, it is necessary to have a survey in this field that can categorize the existing methods, identify the most critical parameters, and, finally, ezylana the tools and techniques of simulation, algorithms, and practices. For instance, Keshanchian et al. proposed a new genetic algorithm (NGA) for solving task scheduling algorithms in a cloud environment [5]. They verified their proposal on the Microsoft Azure platform using C#. In addition, Durillo et al. proposed an energy-efficient multi-objective workflow scheduling algorithm tested in a real Amazon EC2 platform [6]. Other state-of-the-art works utilize simulation tools such as Python, MATLAB, WorkflowSim, etc.

Our main argument is that workflow scheduling is an important and active research area that explores new concepts and techniques to improve the efficiency and effectiveness of the scheduling process. This survey provides a comprehensive overview of the state-of-the-art workflow scheduling methods for different environments. We cover the following topics:

- Workflow scheduling concepts: Present a concise tutorial on requirements, modelling, design techniques, workflow architectures and datasets.
- Workflow scheduling models and frameworks: How are workflow scheduling problems formulated and represented in a cloud-fog environment?
- Workflow scheduling algorithms and techniques: How are fog and cloud workflow schedules generated and evaluated?
- Workflow scheduling challenges and opportunities: How do workflow scheduling methods deal with various issues and limitations?
- Workflow scheduling simulations: The most recent and diverse simulation techniques frequently employed in fog and clouds are also discussed and contrasted.

The subsequent sections of this paper will delve into a review of prior works. Section 2 will focus on Related Work, examining the evolution of workflow scheduling methodologies and highlighting critical contributions in the field. Section 3 will examine the research method. In Sect. 4, we propose a taxonomy, and we describe concepts and background. Then, Sect. 5 will thoroughly analyze selected articles, meticulously categorizing and comparing

the methodologies, criteria, and parameters utilized in their evaluation. Finally, in Sect. 6, we aim to address the research questions posed, discussing the associated difficulties and open challenges within the field.

2 Related works

This section meticulously analyses recent survey papers, reviews, and related works focusing on workflow scheduling. By meticulously examining their strengths and weaknesses, this analysis aims to provide a comprehensive understanding of the current landscape in the field. Amidst the paramount challenge of resource management, numerous scholarly articles have explored quality parameters and their profound impact on workflow scheduling.

In the paper [7], Kaur focuses on qualitative parameters, investigating critical challenges, including energy management, resource cost, reliability, and security. Service delivery hinges significantly on reliable execution; while some businesses may tolerate slower response times, they cannot afford service disruptions, ultimately leading users to abandon unreliable service providers. As cloud/fog resources are susceptible to different kinds of failure, engaging reliable resources is critical for reliable execution to subscribers [8]. This article reached the classification of other methods in the control of workflow, and based on it, it classified, reviewed, and analyzed the articles, which is also influential in determining the simulation tools. The main issue that can be considered for the development of this article is its limitation to cloud environments that do not cover the domain of fog or the combination of cloud and fog. It is also necessary to make the definitions of quality control parameters more precise and transparent. It should be stated so that there is a unified definition during the analysis of the articles. Ahmad et al. [9] had a shared vision with this article, with the difference that they are also limited in terms of the type of workflow by having worked in the field of scientific workflow only.

Hilman et al. [10] adopt a comprehensive approach, focusing primarily on multi-tenant distributed environments, explicitly focusing on cloud and grid environments. The central vision of this work is based on essential challenges related to workflow management, and parameters such as workload volume, reliability, cost, and security have been considered according to the needs of distributed environments. In addition to the fact that several qualitative parameters have been considered and different solutions and algorithms have been analyzed based on meht, the part related to simulation has yet to be deemed essential and is left or future work. On the other hand, Yassir et al. [11] offer a more detailed examination of parameters, methodologies, and simulation tools within cloud environments. They raise pertinent questions concerning reviewing and evaluating results stemming from proposed methods. Notably, their exploration of timing encompasses dynamic and static forms, prompting the expansion of research inquiries rooted in these temporal dynamics. However, their taxonomy remains limited and warrants further elaboration to achieve comprehensiveness.

In their study [12], Versluis et al. presented a step-by-step taxonomy for resource management and workflow management, and the article's explanation based on this point of view is completed. Based on this classification, the assignment of workflow and the supply of resources is divided, classified, and described, and in the next step, the evaluation part is also checked. Allocation techniques have a broader category than previous works, and articles have been selected and analyzed based on heuristic algorithms and mathematical models. However, there is a need to specify acceptance, development, and evaluation criteria within this framework to ensure clarity and consistency. Additionally, expanding the scope beyond cloud environments and exploring similar environments is essential to provide a comprehensive understanding of resource and workflow management practices.

Hosseinzadeh et al. [13] extensively examine multi-objective scheduling methods for cloud computing, employing metaheuristic optimization techniques. The article meticulously categorizes, analyzes, and reviews various solutions within this domain, offering detailed insights into their characteristics and functionalities. It systematically organizes these solutions based on the types of optimization algorithms employed and elucidates their application in addressing scheduling challenges. Additionally, the article conducts comparative assessments among these methods and outlines potential avenues for future research. The findings and contributions of the study are summarized, highlighting its significance in advancing understanding within the field. This article thoroughly examines and categorizes solutions, analyzes metaheuristic algorithms in depth, and classifies quality parameters such as execution time, cost, energy consumption, and error prevention. However, the review of the simulation aspect still needs to be completed, necessitating further analysis and categorization of the tools and techniques utilized in this context.

Due to advancements in artificial intelligence-based solutions, there has been a notable shift in workflow management approaches. Kumar et al. [14] gave a comprehensive overview of the existing machine learning methods for energy-resource allocation, workflow scheduling, and live migration in cloud computing, as well as a taxonomy of their essential challenges. Machine learning is a branch of artificial intelligence that allows systems to learn from data and improve performance

Table 1 State-of-the-art comparisons based on merits and limitations

References	Main topic	Year	Limitation	Our contribution
[7]	Quality of service in workflow management	2019	It is limited to cloud infrastructure	We investigate workflow management in different environmental conditions and combined cloud and fog environments
[9]	Quality of service in scientific workflow management	2021	It is limited to cloud infrastructure and even scientific workflow management	we investigate workflow management in different environmental conditions and combined cloud and fog environment
[10]	Workflow scheduling in distributed systems	2020	The evaluation section must be considered based on different criteria and simulation tools	We added questions about tools, evaluation criteria, and techniques
[11]	Techniques, evaluation parameters, and methods for workflow management	2019	Taxonomy and view must be considered, and it is limited to cloud infrastructure	We proposed a comprehensive taxonomy, investigated workflow management in different environmental conditions, and combined cloud and fog environments
[12]	Workflow and resource management in the cloud environment	2021	Evaluation criteria must be considered, and this paper is limited to cloud environment	we investigate workflow management in different environmental conditions and combined cloud and fog environment
[13]	Workflow and resource management in the cloud environment	2020	Evaluation criteria must be considered, and this paper is limited to cloud environment	we investigate workflow management in different environmental conditions and combined cloud and fog environment
[14]	Workflow management in a cloud environment based on machine learning	2022	limited to the cloud environment, and evaluation is not considered in detail	We added questions about tools, evaluation criteria, and techniques. We investigate workflow management in different environmental conditions and combined cloud and fog environments
[15]	Workflow management in the cloud with tools focusing	2022	There is no classification or analytical diagram for the tools and evaluation part. It is limited to the cloud environment	We added questions about tools, evaluation criteria, and techniques. We investigate workflow management in different environmental conditions and combined cloud and fog environments

without explicit programming. Machine learning can help to optimize various aspects of cloud computing, such as energy consumption, resource utilization, service quality, and reliability. The article introduces a taxonomy categorizing crucial challenges, such as resource heterogeneity, workload dynamism, uncertainty, and optimization. It subsequently provides a systematic overview of existing machine learning methods for energy-resource allocation, workflow scheduling, and live migration in cloud computing, detailing their merits and drawbacks. The evaluation encompasses diverse criteria: performance, cost, reliability, scalability, and security. Menaka et al. [15] addressed a commonly overlooked aspect in many articles by offering a systematic overview of existing workflow scheduling tools and methods in cloud computing, along with their respective advantages and disadvantages. The paper divides the tools and techniques into four groups: heuristic-based, metaheuristic-based, machine learning-based, and hybrid methods. Table 1 compares literature based on the main topic, shortcomings, and how our work highlights points.

According to Table 1, there are three main limitations in the previous articles:

- Limitation 1: Focus on a specific domain, cloud, or fog [7, 9, 12, 14].
- Limitation 2: No classification or taxonomy for workflow scheduling methods [11, 12].
- Limitation 3: Lack of examination of software solutions, systems, algorithms, methods, and evaluation parameters [10, 12, 14, 15].

In this review paper, we avoid these weaknesses and cover the limitations of the research questions.

3 Research methodology

This section outlines our research approach, methods employed, and the selection criteria for the papers included in our study. The process involved four key steps: defining research questions, selecting relevant databases, specifying search terms, and filtering papers. Each step is detailed below.

4 Research questions

This paper is structured to address the following research questions:

RQ1 How do the different methods of scheduling workflows in a fog-cloud environment work, and what are their categories?

RQ2 Which of the various workflow scheduling algorithms and datasets have been suggested, and which have been more popular?

RQ3 What tools have been employed to simulate and implement the methods that have been studied?

RQ4 What parameters are used to assess the performance of workflow scheduling methods in a fog-cloud environment?

RQ5 What are the future challenges and open issues for workflow scheduling in a fog-cloud environment?

4.1 Databases

The papers have been selected from authentic and well-reputed publications to prepare the current survey study—Figs. 1 and 2 show selected papers per database and year. We selected papers from four different databases and focused on papers published in recent years. Our criteria were strict, emphasizing recent publications to maintain relevance and timeliness.

We sourced papers from four databases known for their academic excellence and relevance to the field. This approach aimed to capture various perspectives and insights on the topic.

4.2 Terms and principles

The following search terms were utilized in our database search:

- "Workflow scheduling" AND "Cloud"
- "Workflow scheduling" AND "Fog"
- "Workflow scheduling" AND Fog" AND "Cloud"
- "Workflow" AND "Cloud"
- "Workflow" AND "Fog"
- "Workflow" AND "Fog" AND "Cloud"
- "Workflow" AND "mathematical" AND "Fog" OR "Cloud"

The principles guiding our inclusion criteria are:

- Papers must be published in English.
- Selection preference is given to journal papers.
- The articles considered were published from 2020 to March 2024.

Exclusion principles encompass:

- Papers need more transparency in simulation methodologies and more precise evaluation criteria.
- Articles addressing general scheduling without direct mention of workflow.
- Conference papers.

The application of these criteria served as the basis for paper selection. The research methodology according to the PRISMA template is shown in Fig. 3.

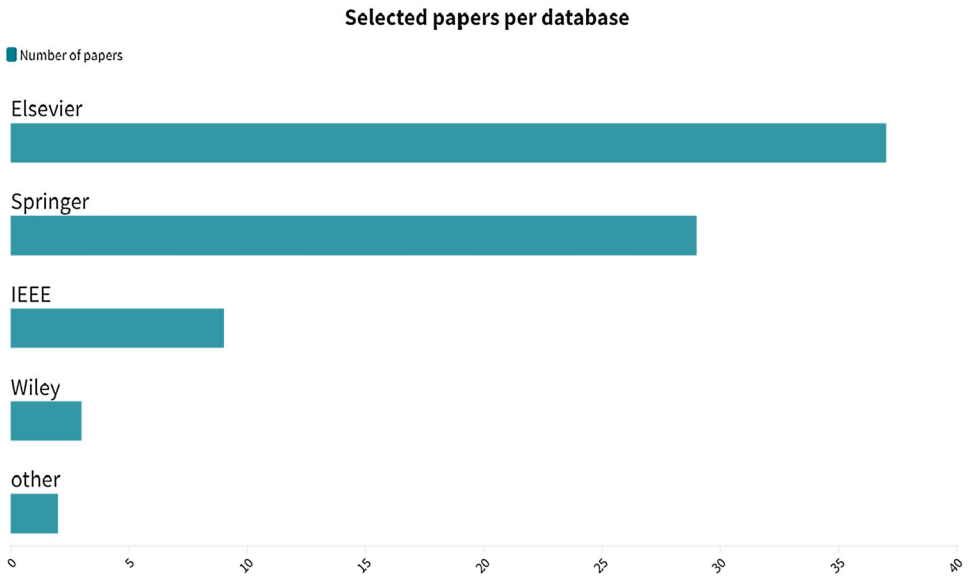


Fig. 1 Selected papers per database

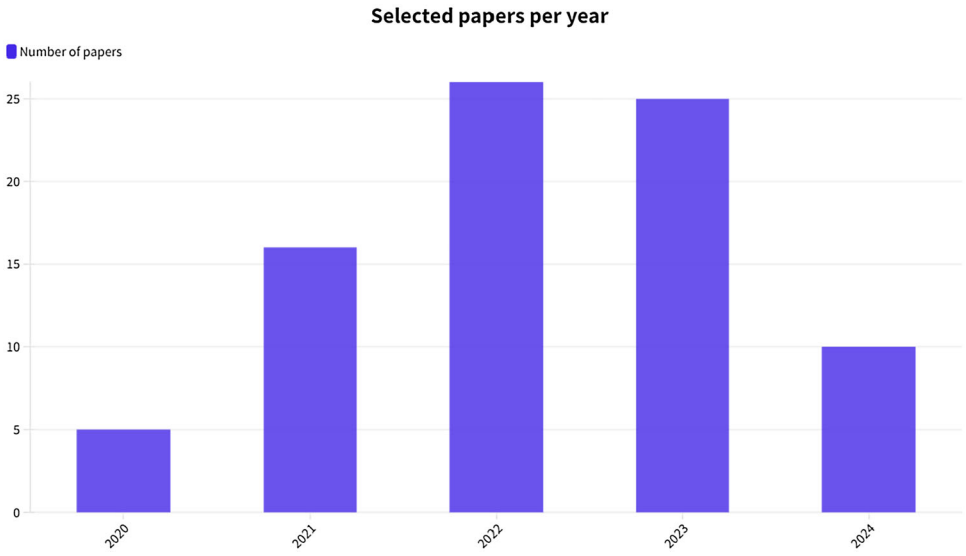


Fig. 2 Selected papers per year

5 Concepts and background

This section presents a taxonomy and a concept review of workflow scheduling in distributed environments. Initially, we scrutinize the architectures and properties of cloud and fog environments. Subsequently, we introduce a taxonomy tailored for workflow management, employing it to elucidate concepts in greater detail. Furthermore, we evaluate existing systems in this domain, assessing their alignment with the proposed taxonomy.

5.1 Cloud fog environments

The Internet of Things (IoT) refers to a network of interconnected devices capable of communication and data sharing via the Internet. In a cloud-fog environment, cloud and fog computing technologies are leveraged to manage and store data originating from IoT devices. Cloud computing operates as a centralized service, providing high performance and storage capabilities, albeit potentially encountering high latency and cost implications [16]. On the other hand, fog computing functions as a distributed service, offering lower latency and facilitating local processing, though it may have limitations in capacity and reliability [17]. Cloud-fog environments aim to optimize some criteria, such as latency, energy consumption, cost, or quality of service, by considering the characteristics and constraints of tasks and resources [18]. A cloud-fog environment can be used for applications and services that do not fit the paradigm of the cloud, including connected vehicles, smart grids, smart cities, and wireless sensors and actuator networks.

In the realm of IoT, resources are categorized based on distinct layers and technological types. The physical layer comprises connected devices and equipment at the foundational level. The intermediary layer is the network and communication layer, encompassing network protocols and standards capable of executing real-time operations. The application and service layer is positioned atop, conceptualized as a cloud or fog layer, tailored to address diverse requirements [19]. Within a cloud-fog environment, there exists a potential to enhance the performance and responsiveness of applications. This improvement is facilitated by processing data close to both the data source and the end user, thereby minimizing latency and bolstering overall efficiency.

A cloud-fog environment can extend across a vast area, employing a network of interconnected fog nodes communicating with each other and the cloud. This setup facilitates device and user mobility accommodation, with fog nodes adapting to evolving network conditions. We can manage many devices and users by using fog nodes that can grow and shrink according to the demand. This environment can support real-time applications that have strict requirements on latency, reliability, and quality of service, but this may need advanced scheduling and optimization algorithms [20]. Figure 4 illustrates a layered architecture representing the distributed environment, incorporating elements like IoT, cloud computing, and fog computing.

In addition to fog and cloud environments, there is also an edge environment. Application execution is facilitated by fog and edge computing when data sources are nearby. To be more specific, Edge computing handles processing at IoT device gateways. One example of an IoT gadget would be a smartwatch. Typically, end users utilize Bluetooth Low Energy (LE) networking to link smartwatches to smartphones to get mobile notifications while driving or walking. In this case, smartphones serve as wristwatch IoT

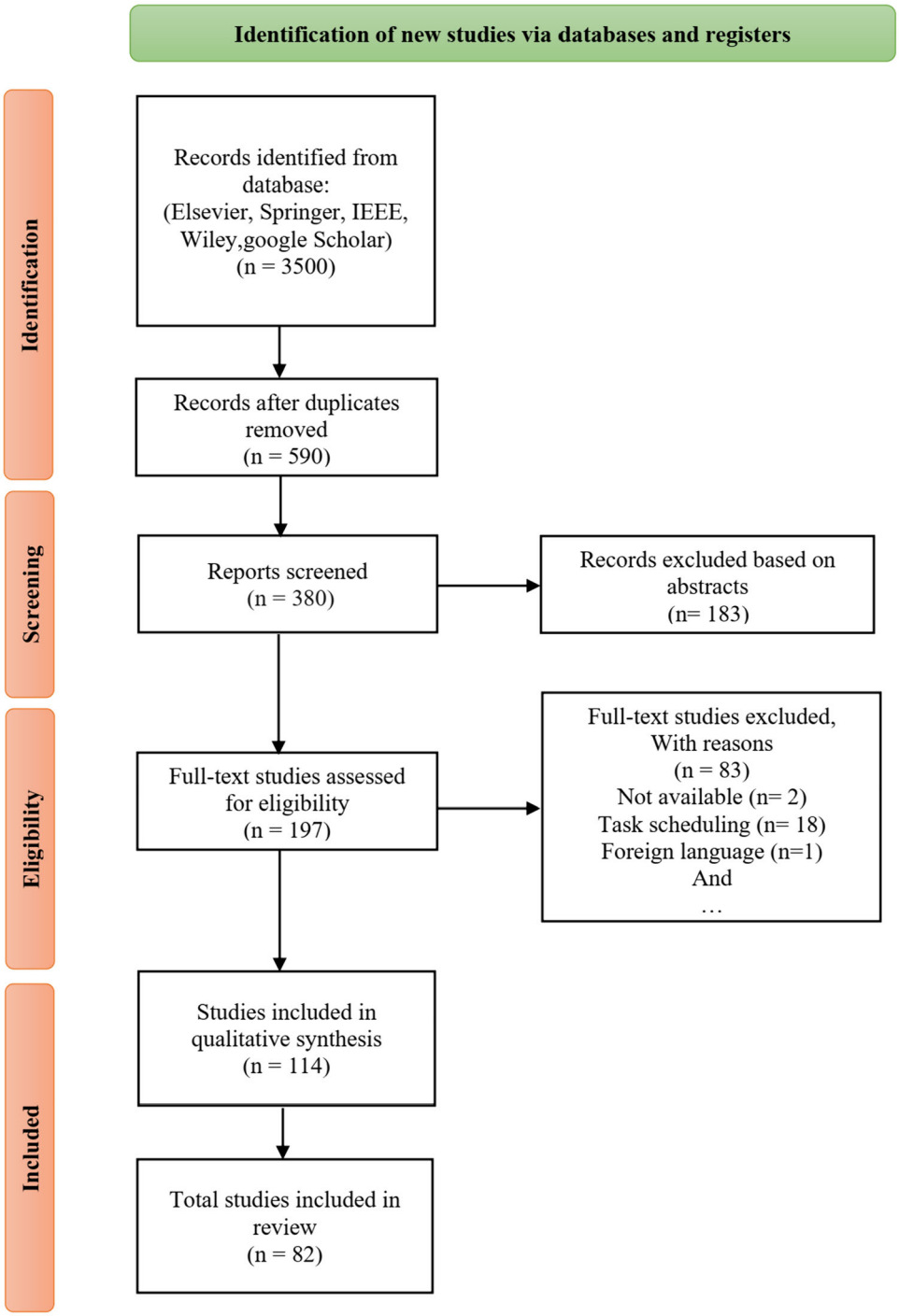


Fig. 3 Research methodology based on the PRISMA template

gateways. Smartwatches simultaneously monitor users' heart rates, blood pressure, and oxygen saturation levels. Edge computation occurs when a wristwatch sends data to a smartphone application for processing.

Comparatively speaking, fog computing processes IoT data by utilizing IoT gateways and additional edge network computing components, including smart routers, PCs, Raspberry Pi devices, and even micro-datacenters. Edge computing has some drawbacks despite being able to address many IoT-related problems. They are less capable of running complicated, large-scale applications over extended periods. Nonetheless, Edge node management primarily focuses on the user, integrating only reactive fault-tolerant features [21, 22]. By utilizing relatively strong resources at the user premises level and reducing the workload associated with resource and application service management from the users, fog computing gets over these Edge restrictions.

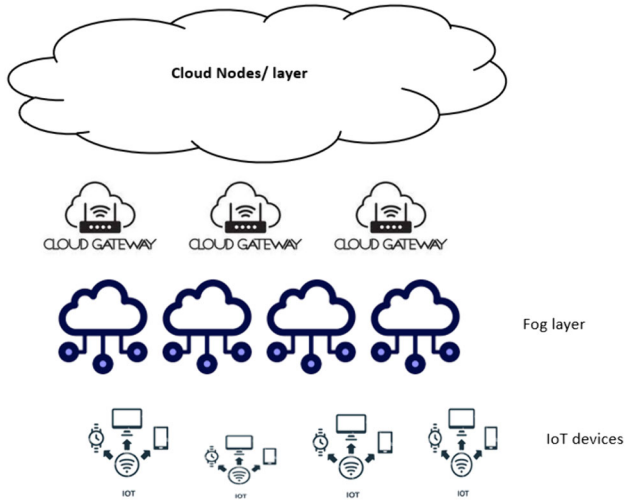


Fig. 4 IoT layered architecture as a cloud-fog environment application

Table 2 Edge and fog [21]

Facts	Edge	Fog
Place of operation	Gateway devices	Specialized networking and computing machines
Elementary hardware	Programmable logic controller	Single-board computer
Wireless standard	Bluetooth and Wi-Fi	Wi-Fi and LTE
Policy manager	users	Service providers
Application deployment	Installed by user	Request by a user to a service provider
Resource assignment	shared	Shared or virtualized
Application user mapping	Multiple application, single user	Multiple applications, multiple users
Resource orientation	Peer-to-peer, ad hoc	Cloud of Things
Cloud communication	Event-driven	seamless
Fault tolerance techniques	User-defined exception handling	Proactive and reactive
Extended from	Personalized computing environments	Cloud computing

Additionally, fog computing keeps up smooth connectivity with cloud data centers, which ultimately provide a vast platform for IoT application execution. Table 2 lists the significant distinctions between Edge and Fog computing [21]. In some research works, Edge computing is viewed as a superset that includes all paradigms where the computation is moved to the edge network, such as Fog computing, Mobile Cloud computing (MCC), and Mobile Edge Computing (MEC) [23]. In other research works, however, Edge computing is considered a subset of Fog computing [24]. Additionally, Chiti et al. [25] provide additional instances in which Fog and Edge computing are utilized interchangeably. Furthermore, Edge computation is sometimes considered a service model provided by several paradigms, such as Fog, Mist, and Dew computing. However, fog computing is considered one of the most viable modern paradigms because of its broad support for Internet of Things applications.

5.2 Workflow and scheduling concepts

A typical workflow comprises a set of tasks with potential dependencies between each pair of tasks. These workflow applications are commonly represented as directed acyclic graphs (DAG) [26]. In other words, a workflow (W) is modelled in $W = < T, A >$, in which T is a set of tasks, and A is a set of arcs that indicates a dependency between tasks. In addition, a set A is determined in $A = \{(t_i, t_j) | t_i, t_j \epsilon T$ *and* t_j*'s execution depends on* t_i*'s output*$\}$. For example, scientific workflow applications like SIPHT, LIGO, and Cybershake follow this model [27, 28]. Figure 5 illustrates a Molecular workflow application used in Physic dynamics for simulations of molecular movement [29].

It is important to note that each t_i within the set of tasks T represents a block of code determined during compile time. A vital question arises regarding the efficient execution of workflow applications. "How can we distribute workflows and tasks among computing resources in a fog-cloud environment?" This is the question that work scheduling in cloud fog tries to answer. Fog computing extends cloud computing to the network's edge, where devices like sensors, cameras, or smartphones can handle some tasks locally. In contrast, others can be offloaded to the cloud for processing. Work scheduling in cloud fog aims to find the best way to assign tasks to resources based on various factors, such as latency, energy consumption, cost, or quality of service, and consider the features and limitations of the tasks and the resources.

In this article, we focus on workflow management. It is essential to distinguish between workflow management and task scheduling, primarily regarding the extent and timing of the work involved. The problem of high-level uncertainty in the workflow parameters affects the execution of lengthy workflow. This is because more than the information at run time and the structural information of workflow is needed for the scheduling algorithms. Workflow scheduling is a process of assigning tasks to resources in a way that optimizes some criteria, such as latency, cost,

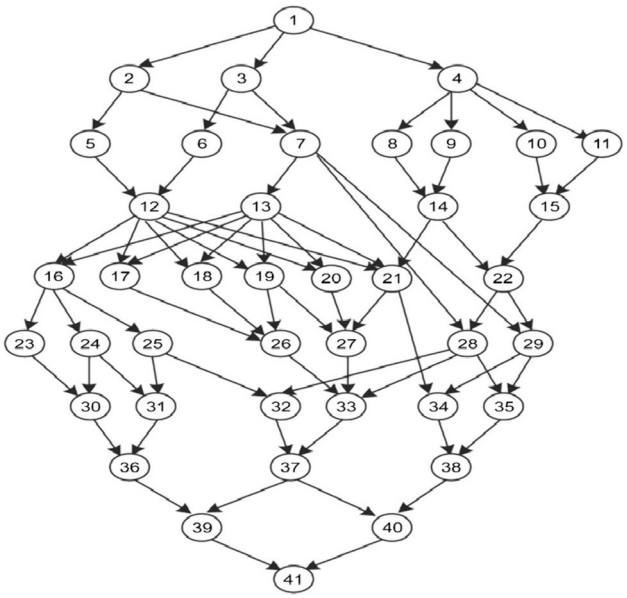


Fig. 5 Molecular workflow with 41 tasks and 72 arcs [30]

or quality of service. Workflow scheduling requires more work upfront, but all the work is completed simultaneously.

The processing and scheduling of workflows can vary depending on their type, often combining stream processing, which is highly sensitive to delay, and batch processing, which involves less time sensitivity but requires more intensive calculations for extensive data analysis. For this article, we will provide a general example. As depicted in Fig. 6, the workflow consists of ten tasks processed based on their work sequence, resource allocation, and scheduling. Initially, a sequence of tasks is determined according to the desired algorithm. Subsequently, based on the objective function (such as delay sensitivity, energy consumption, or completion time), the tasks are scheduled onto resources within the fog or cloud environment. Workflows sensitive to delay are prioritized for execution on fog nodes, while others may not be sent to the cloud. Determining task sensitivity is the responsibility of the Fog Broker, which centrally manages task scheduling and creates the most suitable workflow schedule.

In the field of workflow scheduling, several critical considerations must be addressed. It is possible to schedule workflows and resource allocation with different constraints based on various requirements and quality of service parameters. Also, the issue of cost and energy consumption should be taken into consideration in allocating resources. Figure 8 introduces a taxonomy for workflow management in cloud fog environments, delineating parameters, constraints, tools, criteria, and a general classification of methods. This taxonomy serves as a foundational framework elaborated upon in subsequent sections of the article by analyzing various solutions.

Single-Objective Optimization focuses on optimizing a single objective function to find the best solution. This approach excels in scenarios where the optimization goal is clear and unambiguous. For instance, minimizing production costs while adhering to constraints in manufacturing scheduling exemplifies single-objective Optimization. This approach is straightforward, as the superiority of one solution over another is determined by comparing their objective function values.

Multi-Objective Optimization, on the other hand, involves optimizing multiple conflicting or complementary objectives simultaneously. Rather than directly comparing solutions, dominance determines their goodness. A solution is Pareto optimal if no other solution improves one objective without worsening another. Multi-objective Optimization offers a broader perspective, considering trade-offs and diverse objectives. For example, in vehicle design, the objectives include maximizing performance while minimizing fuel consumption and emissions [31].

Both techniques find applications in various domains, including cloud computing. While single-objective Optimization is more straightforward and faster, multi-objective Optimization provides a comprehensive exploration of trade-offs, leading to informed decision-making and superior outcomes. This approach helps consider various factors such as execution time, cost, energy consumption, and quality of service simultaneously, leading to more informed decision-making and better optimization outcomes in cloud environments and beyond.

5.2.1 Workflow scheduling types

5.2.1.1 Dynamic workflow scheduling The dynamic scheduling process in workflow scheduling involves allocating tasks to resources in response to the changing conditions and requirements of the workflow and the environment. Unlike static scheduling, which assumes fixed and known information about tasks and resources beforehand, dynamic scheduling is more suitable for cloud computing environments where resources and workloads are diverse, uncertain, and dynamic. It aims to optimize various criteria, such as execution time, cost, energy, reliability, or quality of service, by considering the features and limitations of workflows and resources [32]. Some essential steps in the dynamic scheduling process include [32, 33]:

- The workflow is represented as a directed acyclic graph (DAG), where the nodes represent the tasks and the edges represent the dependencies and data transfers between the tasks.

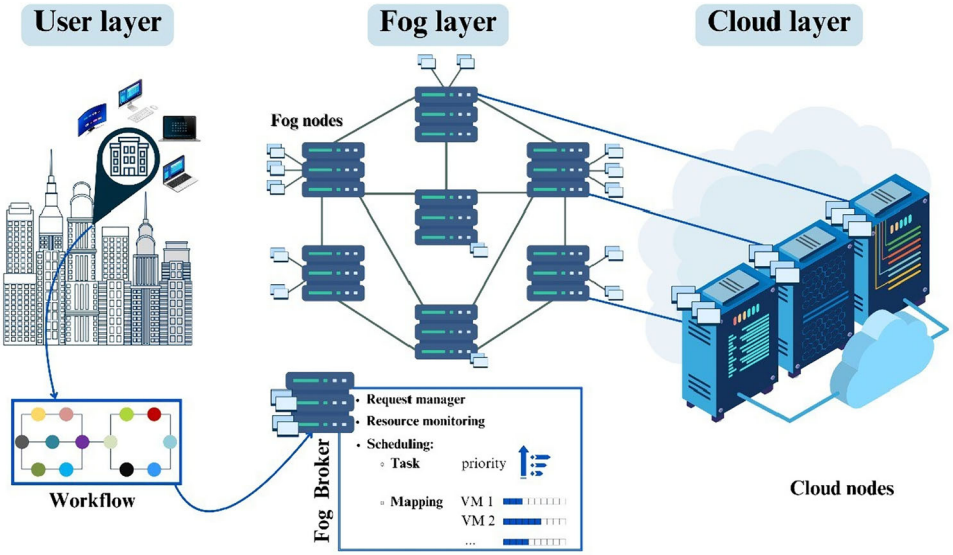


Fig. 6 An example of scheduling a DAG in fog computing

- They are estimating the parameters of the tasks and the resources, such as the execution time, the data size, the resource capacity, the resource availability, and the resource cost. These parameters may vary over time and may not be known precisely in advance.
- It is choosing a scheduling algorithm that can deal with the unpredictability and variability of the parameters and balance the trade-offs between the optimization criteria.
- They perform the tasks on the resources according to the scheduling algorithm and monitor the progress and performance of the workflow. If there are any changes or failures in the tasks or the resources, the scheduling algorithm may need to reschedule or move the tasks to other resources.
- They are evaluating the results and the performance of the workflow and the scheduling algorithm and comparing them with the expected outcomes and the optimization criteria.

5.2.1.2 Static workflow scheduling Static workflow scheduling is an algorithm that assigns tasks to resources based on fixed and known information. This approach assumes that task and resource parameters, such as execution time, data size, resource capacity, availability, and cost, remain constant. By leveraging mathematical models and optimization techniques, static workflow scheduling aims to optimize criteria like execution time, cost, energy, reliability, or quality of service. It is well-suited for environments characterized by uniform, predictable, and stable resources and workloads. In contrast to dynamic workflow scheduling, static scheduling often outperforms scenarios where workflow-level scheduling decisions are

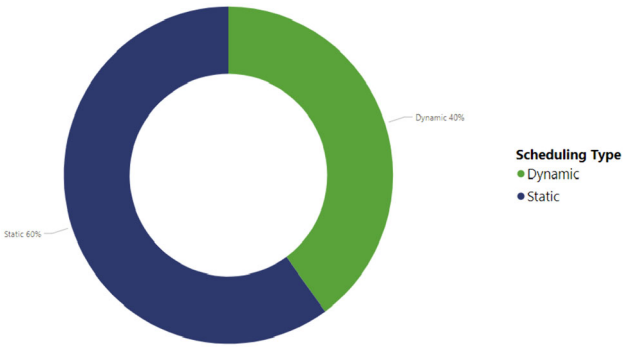


Fig. 7 Distribution of scheduling types: dynamic vs. static

crucial. However, it comes with potential drawbacks, including higher overhead and complexity, attributed to the necessity of solving large-scale optimization problems [32].

Figure 7 visualizes the proportional distribution between these two scheduling types.

5.2.2 Workflow scheduling architectures

A workflow scheduling architecture is a system design and structure that aims to optimize the execution of workflows across various computing resources. These architectures can be classified into different types based on several criteria, including the level of abstraction, degree of distribution, mode of operation, and optimization approach. Below are examples of workflow scheduling architecture types [32].

Centralized vs. distributed: In a Centralized Architecture, a singular scheduler assumes control over resource allocation and task execution across the workflow. Conversely, a Distributed Architecture employs multiple

schedulers, which may operate autonomously or collaboratively, managing distinct workflows or sub-workflows. These contrasting approaches offer unique advantages and considerations in optimizing workflow execution across diverse computing environments.

5.2.3 Workflow modelling

Workflow models serve as unique representations of the tasks and activities within a workflow management system. A workflow could be a set of forbidden assignments that must be executed in a particular arrangement to realize a specific objective. A workflow captures the basic properties of the errands, such as their inputs, yields, preconditions, post-conditions, activities, exceptional cases, and traits. A workflow also characterizes the connections and conditions among the assignments, such as their grouping, parallelism, synchronization, and branching.

Various diagrams and graphs are utilized to depict different aspects of workflows:

5.2.3.1 Petri net Directed arcs connect the two types of nodes that make up a Petri net: places and transitions. Tokens indicate the system's current state and can be stored in areas. If sufficient tokens exist in the transitions' input locations, they can initiate, generate, and consume tokens following the arc weights. Transitions can simulate concurrency, synchronization, choice, and iteration because they fire in a nondeterministic manner. This kind of workflow modelling displays a workflow's state and progression. It uses bars for transitions (events) and circles for locations (states). Petri nets are helpful in simulating workflow concurrency and synchronization, as well as conflicts and deadlocks. This model makes it easier to specify workflow applications and acts as a firm.

5.2.3.2 Dataflow graph This kind of workflow modelling displays the computation of a workflow together with the data. It employs edges for data (values) and nodes for operations (functions). Dataflow graphs are helpful in simulating workflow dependencies and parallelism, as well as data processing and Optimization. The assignment and coordination of computational modules among processing resources, or scheduling, is crucial in dataflow-based design processes that influence real-world performance metrics like latency, throughput, energy consumption, and memory requirements. A formal abstraction for scheduling in dataflow-based design processes is offered by dataflow schedule graphs (SDGs). With the DSG abstraction, schedule designers can represent a schedule as a distinct dataflow graph, giving rise to a formal, abstract, and language- and platform-independent representation of the schedule [17].

5.2.3.3 Event-driven process chain (EPC) This workflow chart shows a workflow's control and work. It employs circles for occasions (states) and hexagons for capacities (exercises). EPCs are valuable for modelling a workflow's rationale and semantics, as well as the exemptions and varieties. An EPC chart may be a graphical and scientific demonstration that can be utilized to depict and analyze the behaviour of concurrent and disseminated frameworks. This could capture the conditions, conditions, circles, and parallelism among the exercises of a handle [4].

5.2.3.4 Directed-acyclic graph (DAG) It could be a chart in which the edges have a course, and there are no cycles, meaning that no vertex can reach itself through an arrangement of advantages. DAGs are commonly utilized to speak to complex connections between errands in a workflow, such as the conditions, conditions, circles, and parallelism among the activities of a handle. A workflow chart includes a single source vertex and a single sink vertex; each vertex goes from the source to the sink. A workflow chart can capture the causal structure among the factors included in a handle and give a basis for choosing bewildering aspects to alter when assessing causal impacts.

6 Task description

Task Descriptions play a pivotal role in workflow scheduling. They provide a written explanation of each workflow step's purpose, inputs, outputs, dependencies, roles, and deadlines. A task description can make the process more transparent and beneficial for all stakeholders. Bridging the gap between abstract workflow models and practical execution contributes to efficient task completion and successful project outcomes.

Figure 8 is designed based on our proposed taxonomy. Other features of the taxonomy become more detailed throughout our paper.

7 Workflow scheduling methods and algorithms

A general classification of methods can be found according to the methods and algorithms used in various articles. A group of methods uses heuristic algorithms; another uses modelling and other AI-based methods and parametric modelling for workflow scheduling. Parametric modelling typically denotes approaches where parameters or mathematical relationships characterize the problem.

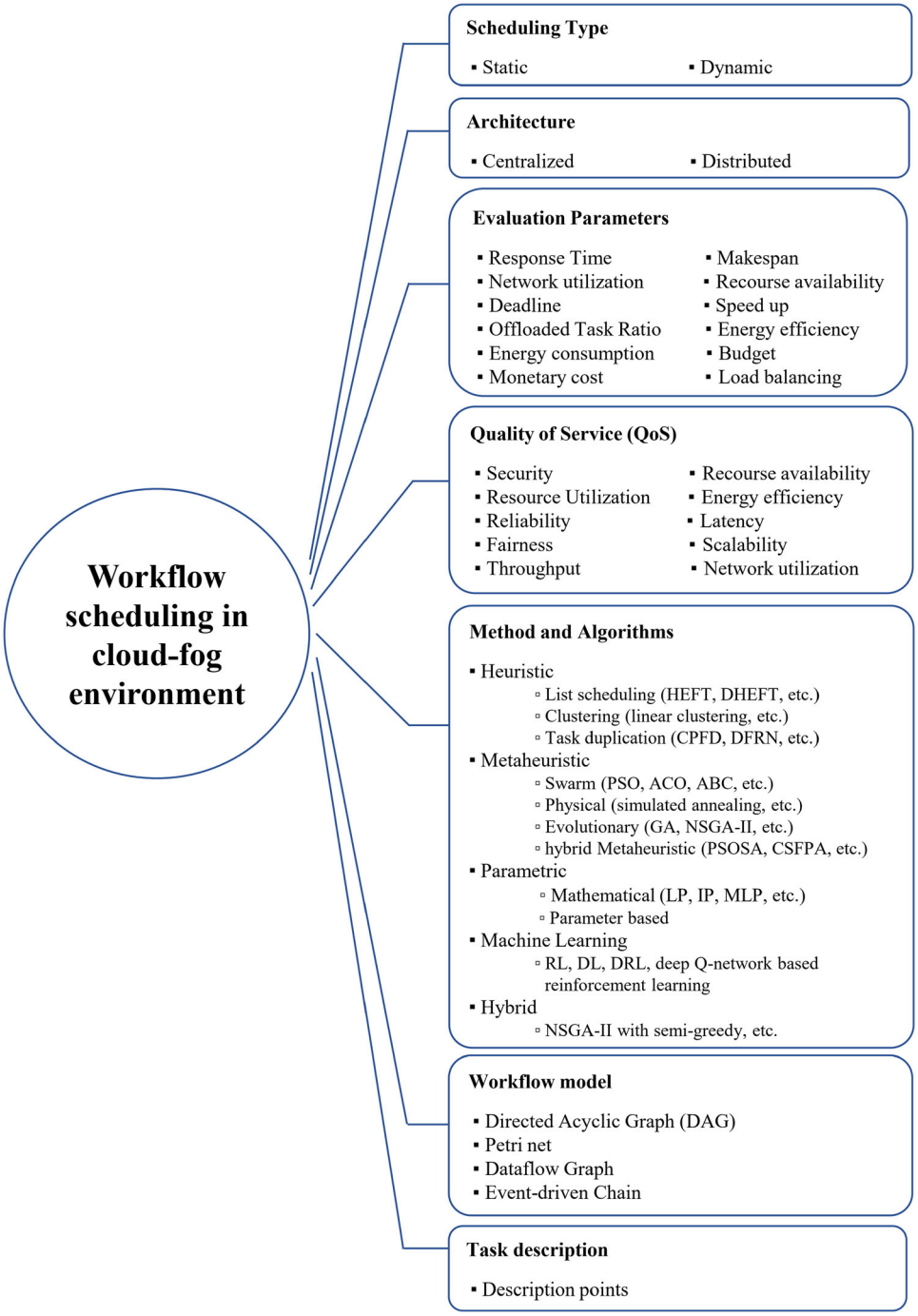


Fig. 8 Proposed subjective taxonomy for workflow scheduling management

7.1 Heuristic algorithms

Heuristic workflow scheduling is an algorithm that uses simple rules and strategies to allocate tasks to resources in a cloud computing environment. Heuristic workflow scheduling differs from meta-heuristic workflow scheduling, which employs more advanced and complex methods, such as evolutionary algorithms, swarm intelligence algorithms, or reinforcement learning algorithms, to find the best or near-best solution for the workflow scheduling problem. Heuristic workflow scheduling is also different from hybrid workflow scheduling, which combines heuristic and meta-heuristic methods to enhance the performance and efficiency of workflow scheduling. In addition, Heuristic architectures use simple, fast algorithms that provide near-optimal solutions to a given problem. Meta-heuristic architectures use general-purpose, flexible algorithms that explore large search spaces and offer high-

quality solutions to various issues. Heuristic workflow scheduling can deal with large-scale and complex workflows, as it can shrink the search space and the complexity of the problem. Heuristic workflow scheduling may not balance the trade-offs between the criteria, as it may prefer one standard over another or use a fixed priority order [34].

List scheduling is a heuristic scheduling algorithm that lists all tasks in the graph according to their priorities. These algorithms have two phases. The first phase is task prioritization or selection. In this phase, tasks with the highest priority are selected and placed in a list or queue ready for execution. The second phase is processor selection. In this phase, a processor that has the lowest cost is selected. The cost can be the task completion time, monetary cost, energy, etc. Some of the methods in this group are the heterogeneous earliest finish time algorithm [35] HEFT, modified critical path, dynamic level scheduling and heuristic mapping. The Heterogeneous Earliest Finish Time (HEFT) method is used to discover the order of task flow execution [36]. Effective scheduling methods are needed to lower the energy consumption to assign tasks to the most suitable resources. This paper presents a binary model that uses a combination of the Krill Herd Algorithm (KHA) and the Artificial Hummingbird Algorithm (AHA) as Binary KHA-AHA (BAHA-KHA). KHA enhances AHA. The paper also uses the dynamic voltage and frequency scaling (DVFS) method to solve the local optimal problem for task scheduling in FC environments. The paper by Hajam et al. [37] introduces a spider monkey optimization algorithm with heuristic initialization for resource allocation and scheduling in a fog computing network. The algorithm reduces the total cost of tasks by selecting the best fog nodes. The paper suggests and compares three ways to initialize the SMO algorithm based on the longest job's fastest processor, the shortest job's fastest processor, and the minimum completion time. The paper also presents a mathematical system model to solve the optimization problem.

Among these methods, list scheduling is usually more practical and provides better performance results in less time than other groups. The algorithms in this group divide the tasks in a given graph into unlimited clusters. In each step, the tasks selected for clustering can be any task, not necessarily a ready task. In each iteration, the previous cluster is modified by merging some clusters. If two tasks are assigned to the same cluster, they will be executed on the same processor. Clustering heuristics require additional steps to produce a final schedule, the cluster merging step. To merge clusters so that the number of remaining clusters is equal to the number of processors, a cluster mapping step is required to map clusters to available processors. A task sequencing step is required to sequence the mapped tasks on each processor [35, 38]. Task duplication algorithms Task duplication scheduling attempts to reduce communication delays by executing some of the previous tasks on more than one virtual machine. Task duplication algorithms differ based on the task selection strategy for duplication [39]. The algorithms in this group are usually for an unlimited number of identical processors, and their complexity is much higher than that of algorithms in other groups. A group of researchers have used multi-objective mathematical models to cover various parameters. Noorian Talouki et al. [28] proposed a method for scheduling tasks in cloud computing, where the goal is to optimize the time of the task execution. The tasks depend on each other and must be executed in order. The method uses a new task priority strategy and a task duplication method that lowers the execution time.

In [40], Li et al. used several combined strategies. They designed a three-step model for scheduling and deploying container-based workflows in a cloud-edge environment. A cloud-edge environment is a hybrid system that combines cloud computing and edge computing. Cloud computing is a way of providing and using computing resources over the Internet. Edge computing is a way of supplying and using computing resources near the users or data sources. Container-based workflows are workflows that use containers for packaging and running tasks. Containers are lightweight and isolated units that can run on different machines. The first step of the model is to allocate a virtual CPU (vCPU) for each container so that other containers can share vCPU. The second step is to schedule the containers to virtual machines (VM), simulated machines that run on physical devices. The third step is to schedule the VM to physical machines, which are either in the cloud or at the edge.

The model considers multiple goals, such as reducing the time, the imbalance, and the energy use for both the cloud-edge resources and the container-based workflows. The model uses three evolutionary strategies and two multi-objective algorithms to find optimal solutions. Bugingo et al. [41] worked on cost optimization and energy consumption in scheduling workflows. This article proposes a heuristic algorithm which determines a weight for resource selection and allocation criteria in two steps. First, the resource is selected according to the capacity and frequency of the CPU. This method works by adjusting the voltage and frequency of the processor to reduce power and energy consumption. In [42], Khaleel proposed a distributed algorithm for scheduling scientific workflows on cloud and fog resources. The algorithm aimed to optimize two objectives: energy consumption and scheduling reliability, which affect the system efficiency. The algorithm used a reliability-aware technique that adjusted the machines' performance-to-power ratio (PPR) based on their utilization. PPR measures how much work a machine

can do with a given power. The algorithm ensured that the cloud and fog machines operated at the optimal utilization level to minimize energy consumption. They used a static workflow scheduling method that assumes the workflow structure and the resource availability are known in advance.

Since the financial budget is essential for individuals and enterprises utilizing cloud/fog resources, a new bi-objective optimization workflow scheduling algorithm was developed by Shirvani and Noorian to cope with task scheduling problems from a monetary and time costs perspective [43]. A new framework was proposed, which logs all of the underlying resource failures and calculates how the resource is reliable for task execution. Finally, the Pareto set of solutions, which makes a trade-off between makespan and reliability, is returned. Since the financial budget is crucial for individuals and enterprises utilizing cloud/fog resources, a new bi-objective optimization workflow scheduling algorithm was developed by Shirvani and Noorian to cope with task scheduling problems from a monetary and time costs perspective [43].

As seen in the articles, researchers have discussed various parameters such as performance, time, cost, and energy and have provided solutions based on heuristic algorithms that, according to the dimensions of cloud and fog-based systems, can Cover several parameters simultaneously and on a large scale. Another group of articles has used the combination of mathematical models with other intelligence-based solutions, which we will examine further.

7.2 AI-based methods

Hybrid workflow scheduling is a method for running complex applications with stream and batch processing tasks on different systems. It tries to improve the performance and cost of running hybrid workflows with additional and distributed resources at the edge and cloud layers. Hybrid workflow scheduling has two main steps: resource estimation and task scheduling. Resource estimation entails gauging the requisite amount and type of resources—such as CPU, memory, bandwidth, and budget—necessary for executing a hybrid workflow effectively. By capitalizing on the distinct advantages of stream and batch processing, this method strives to achieve optimal outcomes in low latency, high throughput, and scalability. However, the dynamic nature of the edge cloud environment introduces fresh challenges and trade-offs in both resource estimation and task scheduling, including uncertainties, heterogeneity, and dynamism, which necessitate careful consideration [44]. Qin et al. [45] proposed a technique based on the Fruit Fly optimization algorithm (FOA) for scheduling workflows in cloud computing, where the goal is to optimize both the time and the cost of the workflow execution. FOA uses a cluster strategy based on reference points to divide the solutions into groups. FOA also uses some operators designed for the specific workflow scheduling problem. To create the initial solutions, FOA uses three heuristics, one of which is based on a non-linear weight vector to improve the distribution of the solutions. To improve the solutions, FOA uses three neighbourhood search operators that work together in the smell-based foraging phase. FOA also uses the vision-based foraging step, extending the solutions based on the sub-groups and the crossover operator to speed up the convergence. Li et al. [46] proposed an improved Farmland Fertility algorithm by combining Mutation strategies and Dynamic Objective strategies for scheduling workflows in the cloud to optimize cost under deadline constraints. Dynamic objective is a process that dynamically adjusts the objectives' weights, such as minimizing the execution time, the cost, and the energy consumption. Dynamic objectives can help the algorithm balance the trade-offs among the objectives and achieve better performance. They considered scalability and extensive experiments were conducted on well-known scientific workflows with different types and sizes. This method is not considered in the detailed analysis of the complexity and scalability of the proposed method, which may limit its applicability to large-scale and dynamic cloud systems. From a complete point of view, some have investigated user needs in heterogeneous distributed environments instead of service parameters. Mokni et al. [47] addressed the problem of allocating workflow tasks to fog and cloud nodes, considering multiple objectives such as execution time, energy consumption, cost, and reliability. The paper used an adapted Genetic Algorithm for workflow scheduling in minimal time. The article claims that the proposed method can achieve better trade-offs among the objectives and improve the quality of service for the users.

A hybrid task scheduling scheme based on an evolutionary algorithm was proposed by Bay Wu et al. [48] to solve workflow executions in edge and fog scenarios. They modelled given projects as a directed acyclic graph (DAG) and incorporated a partition operator to split DAG into an ordered queue of tasks; then, it maps each of them to appropriate available servers, guaranteeing the quality of service.

Abualigahet al. [49] used the Dragonfly algorithm, which mimics the swarming behaviours of dragonflies, to solve optimization problems in IoT cloud computing applications. This method decreases the makespan and increases resource utilization. They focused on significant data task scheduling. The authors improved the two features in their heuristic method: mutation and dynamic objective. Mutation is a process that randomly changes

some parts of the technique, such as the number of tasks, the rate of mixing, and the rate of switching. Mutation can help the method avoid getting stuck in the wrong solutions and find new solutions. Dynamic objective is a process that changes the importance of the goals, such as minimizing the time, the cost, and the energy use. Dynamic objectives help the method balance the trade-offs among the goals and achieve better performance. The type of workflow scheduling used is a hybrid approach that combines a bio-inspired optimization algorithm and a local search technique. The proposed method, named MHDA, optimized the energy consumption and the makespan of the workflow tasks while meeting the deadline constraint.

Mollajafari et al. [46] worked on time and performance parameters and optimized the cost by modelling cloud features and the time and cost of data storage. The paper by Mohammadzadeh et al. [50] presented a new HDSOS-GOA method that uses a hybrid chaotic algorithm to schedule tasks in fog computing environments. HDSOS-GOA uses Symbiotic Organisms Search (SOS) and Grasshopper Optimization Algorithm (GOA) algorithms and chooses between them based on the probability computed by the learning automata. The paper aimed to minimize the makespan, cost, and energy consumption of scientific workflows on fog nodes and cloud servers. The paper also uses the Dynamic voltage and frequency scaling (DVFS) approach to lower energy consumption. The HDSOS-GOA algorithm assigns the tasks to the most appropriate VMs and selects the optimal DVFS-level VMs. The paper's goal is to reduce the energy consumption and the scheduling time of workflow scheduling. The paper also reduces the latency and bandwidth consumption by using the fog nodes near the end-users. The article by Singh et al. [51] presented a hybrid GA-modified PSO method for resource assignment. The technique aims to minimize the makespan, cost, and energy consumption of tasks that depend on each other on various resources in cloud-fog computing settings. The method uses slow VMs to prevent the job execution from being delayed. The technique uses the same weights for the makespan, energy consumption, and execution cost in the fitness function to achieve the goal. The paper shows that the proposed method is better than other existing algorithms regarding the makespan, cost, and energy consumption of running scientific workflows on cloud and fog resources. The paper [52] introduced a hybrid multi-criteria decision model that addresses the reliability challenge mentioned in articles [53] and [50]. The model enhances the timing reliability and optimizes the time-related parameters of the workflow execution. The model can adjust to the changes and uncertainties of cloud-fog resources and workflow applications.

Metaheuristic algorithms, including genetic algorithms and simulated annealing, provide robust solutions for tackling intricate optimization dilemmas. Unlike exact approaches guaranteeing optimal outcomes but struggling with extensive solution spaces, meta-heuristics efficiently explore these expansive domains, particularly suitable for NP-hard problems or situations with incomplete data. These algorithms, such as genetic algorithms, simulated annealing, and ant colony optimization, are aptly suited for NP-hard scheduling predicaments with incomplete information. Although they do not assure global optimality, their minimal time complexity makes them applicable for real-time endeavours with adaptable constraints. Demonstrating efficacy in workflow scheduling, particularly in multi-objective contexts, they adeptly navigate intricate Pareto fronts by intelligently sampling vast solution spaces. However, while predominantly employed in single-objective workflow scheduling, the substantial potential exists to be harnessed for reconciling trade-offs and constraints in cloud-centric workflow environments, furnishing near-optimal outcomes at manageable computational costs [4, 54].

Some hybrid methods have been presented on cost and energy parameters, and their superiority over using a specific heuristic algorithm is to improve various parameters at the system level and provide a universal solution. Bacanin et al. [55] proposed a method for scheduling workflows in cloud computing, where the goal is to optimize the time of the workflow execution. The process uses an improved firefly algorithm, a technique that imitates the behaviour of fireflies, where each individual (firefly) moves and communicates with others to find the best solution. The paper claims that the improved firefly algorithm can solve the problems of the original firefly algorithm, such as slow convergence and early convergence, by using a mutation operator and a population renewal mechanism. The paper also uses a task clustering technique, which divides the tasks into groups based on their data dependencies and communication costs to lower the data transfer time and the network bandwidth consumption. The paper introduces a new way to represent the solutions using fireflies, where each firefly shows the type and the number of cloud and edge resources needed by each task group. The paper also explains a decoding procedure that transforms a firefly into a scheduling solution, where the tasks are allocated to the resources based on their priorities and availability. The paper also uses a repair method to correct the invalid solutions that may happen because of the randomness of the firefly algorithm. The article shows that the technique can lower the workflow execution time by selecting the most appropriate cloud and edge resources for each task group and using the unused periods on the resources to increase resource utilization.

As mentioned earlier, reliable service delivery is vital for providers and requesters. The reliability of service delivery is a reputational parameter for service providers

and simultaneously makes service requesters highly adhere to it because their business continuity strongly depends on reliable execution. To this end, Asghari et al. proposed a bi-objective workflow scheduling algorithm based on a cuckoo search optimization algorithm [56]. A new framework was proposed, which logs all of the underlying resource failures and calculates how the resource is reliable for task execution. Finally, the Pareto set of solutions, which makes a trade-off between makespan and reliability, is returned.

Some researchers worked on time parameters for workflow prioritization and allocation of resources. Hafsi et al. [57] have gone for a combined solution using a genetic algorithm and particle swarm optimization. The main advantage of this method is its high scalability in terms of demand and time-constrained problems, which is also considered in simulation. They used an accurate number coding based on a random key with a limited value range, which can improve the search efficiency and avoid premature convergence. Dynamic adaptive decoding can decode the actual number coding into a feasible scheduling solution using an active mapping table that can adjust to the changing cloud environment.

Xie et al. [58] worked on cost management and suggested a new genetic algorithm for cloud workflow scheduling to assign available cloud resources based on performance and cost optimization. They designed an iterative backward scheduler that can improve the solution by repeatedly scheduling workflow tasks while considering resource availability and task dependency. This method has a complete solution space. It can also efficiently search the entire solution space and find the optimal solution in theory. This method has proper scalability. It can also adapt and improve the environmental variables by developing dynamically.

Similarly, article [59] proposed a method called CEPO for scheduling resources in cloud computing, where the goal is to optimize the cloud system's performance and cost. CEPO combines two techniques: cultural algorithm (CA) and emperor penguin optimizer (EPO). CA is a technique that uses shared memory to store and update the best solutions. EPO is a technique that simulates the behaviour of emperor penguins, where each penguin moves and communicates with others to find the best solution. The paper claims that using CA can improve the efficiency of EPO by enhancing its exploitation ability.

Systems operating within hybrid cloud-edge environments necessitate tailored structures and strategies to address each layer's unique demands and capabilities. A cloud-edge environment seamlessly integrates resources from cloud and edge computing domains, with the cloud offering centralized, scalable services and the edge providing distributed, low-latency services. Within edge cloud systems, heterogeneous resources span across edge and cloud layers, boasting diverse capabilities and cost structures. Fog computing, as a subset of edge cloud systems, leverages fog nodes and intermediate servers to furnish computation and storage services to end devices, optimizing proximity and performance in a study by Khaledian et al.

In a study by Khaledian et al. [60], a novel approach was proposed to schedule complex applications with multiple tasks within edge cloud systems to enhance performance and cost-effectiveness. This approach, rooted in the krill herd algorithm—a meta-heuristic algorithm inspired by the collective behaviour of krill swarms—aims to optimize workflow scheduling in fog-cloud environments. Moreover, integrating dynamic voltage and frequency scaling (DVFS) techniques further enhances energy efficiency by adjusting voltage and frequency levels in fog nodes. This method has two main steps: population initialization and individual improvement. Population initialization creates an excellent initial population by using the level information of the workflow tasks and some heuristic algorithms, such as the dynamic heterogeneous earliest finish time (DHEFT). Individual improvement is improving the individuals by using the krill herd algorithm, which can search the search space and find the optimal or near-optimal solutions. The authors enhanced the individual solutions by using some novel strategies for personal improvement and local search, such as task swapping and task insertion operators, which can balance the workload among the fog nodes and reduce the communication overhead among the workflow tasks and the simulated annealing algorithm, which can explore the neighbourhood solutions and escape from local optima.

Kamanga et al. [61] developed a scheduling algorithm for cost-time minimization, and they specified CPU frequency for each task. With this scheduling, using the lowest CPU frequency may not be cost-effective since increasing the execution time may raise the cost, too [19]. They made sure that the CPU frequency selection for each activity is not based just on cost or execution time but also allows for flexibility in both execution time and cost. In [61], a method is also proposed to allocate resources similarly. The proposed method selects the computing resource and CPU speed according to the highest performance score calculated by Manhattan and Euclidean distance. Paper [62] has taken a step ahead of [63] and has improved the use of data centers with a more comprehensive view of user requirements. The paper [62] proposed a new method called PBMO-DALO, which uses an ant lion optimization algorithm to schedule workflow tasks in cloud data centers. Workflow tasks depend on each other and have different needs and limits, such as how long they take, how much they cost, and how well they perform. The method aims to

reduce the time and energy used for the workflow execution simultaneously. The method uses a new way of coding and moving the ants and ant lions, inspired by nature. The method also compares and chooses the best solutions based on multiple goals, such as time, cost, and energy use. The method can find solutions that offer different trade-offs among the objectives. The method can also change the importance of the goals based on the situation and the user preference. The paper tested the method on different workflow applications and showed that the method performed better than other methods in terms of time, cost, energy use, and quality of service.

Another group has dealt with more parameters and established a trade-off point between the parameters. The paper [64] introduced a fog computing resource management (FRM) model with three primary solutions. First, it computes the resource availability based on the average execution time of each task. Second, it improves the load balancing by using a hybrid method that mixes the multi-agent load balancing algorithm and the throttled load balancing algorithm. Third, it schedules the tasks based on priority, resource availability, and load balancing. The FRM model has three levels of processing: personal agents (Pas), fog node agents (FNAs), and cloud. It also has two levels of control: master private agents (MPAs) and master fog node agents (MFNAs).

The FRM model uses four parameters: task initial priority, task assignment to the fog node, resource availability calculation in the fog computing tier, and task migration to the cloud. This method can deal with real-time and emergency healthcare applications with different priorities and deadlines. It can distribute the workload and the resource utilization among fog nodes and cloud servers. Javaheri et al. [65] proposed a 3-tier scheduling scheme based on the Hidden Markov Model (HMM) model for managing scientific workflows in multi-fog computing environments, in which a broker node in the IoT layer chooses the suitable fogs based on the availability of fog computing providers to submit the IoT workflows. The authors improved the performance and cost of running complex applications on multi-fog systems using an enhanced discrete Harris hawk optimization algorithm (IDHHO). This meta-heuristic algorithm imitates the hunting behaviour of Harris hawks. IDHHO uses a discrete encoding scheme based on the position and velocity of the hawks, which can enhance search efficiency and avoid premature convergence.

Qiu et al. [66] suggested a method for scheduling tasks on edge cloud systems that need to optimize both the execution time and the energy consumption simultaneously, which is a kind of evolutionary algorithm that uses multiple sub-populations with different genetic operators to search for the best solutions. This paper divided the population into three groups: superior, ordinary, and inferior, which are based on the quality of the individuals. The superior group uses a genetic operator based on the superior individuals' longest common subsequence (LCS), which can keep some good gene blocks and speed up the convergence. The ordinary group uses the usual genetic operators, such as crossover and mutation, which can preserve the diversity and balance of the population. The inferior group uses a genetic operator based on the LCS of the flawed individuals, which can remove some lousy gene blocks and help the individuals escape from local optima. The paper also designed a dynamic mechanism that can change the size and composition of the groups according to the fitness landscape and the generation number. In each generation, the non-dominated sorting and crowding distance ranking methods are used to evaluate the quality of each individual. This method can change the size and composition of the groups according to the fitness landscape and the generation number by using a dynamic mechanism, which can adapt to the changing and uncertain environment of edge cloud systems.

On the other hand, it can need help to balance the trade-offs between performance and cost, which may lead to conflicting and inconsistent objectives and preferences. Wang et al. [67] proposed a method for scheduling workflows in cloud computing, where the goal is to optimize the execution cost of the workflow while respecting a deadline. The technique combines particle swarm optimization (PSO) and idle time slot-aware rules. The paper claims that using these rules can increase resource utilization and save the execution cost of the workflow. The paper introduced a new way to represent the solutions using particles, where each particle shows the type of cloud resource needed by each task and the order of functions. The paper also explains a decoding procedure that transforms a particle into a scheduling solution using idle time slot-aware rules. The paper also used a repair method to correct the invalid order of tasks that may happen because of the randomness of PSO. The paper showed that the technique can find reasonable solutions using PSO, which can search an ample space and communicate with other particles to find the best solution. The paper also shows that the method may not work well for workflows with more than one objective or constraint, such as energy consumption, reliability, security, etc., as it only focuses on the cost and the deadline as the primary objective and discretion. Article [30], in determining the objectives of the article, directly mentions both the response time management in the fog and the cost optimization in the allocated cloud elastic resources, as well as the proposed approach as a multi-agent approach in the relevant environment. It is simulated to the Internet of Things. In this approach, reliability, cost, and availability are also covered according to the defined parameters. However, it is necessary to complete some of them

according to user needs in the natural environment. As seen in the review of these articles, a group of papers has presented models to improve important service parameters, and another group has used intelligence-based solutions to adapt to mixed environments.

The improved Owl search algorithm (OSAM) [68] is a method for scheduling workflows in cloud computing, where the goal is to minimize the makespan (the total execution time) of the workflow while respecting the budget limit. OSAM uses a new mutation strategy to increase the diversity of the solutions and avoid getting stuck in local optima. OSAM also uses a population update mechanism that adapts the step parameter, β, according to the current best solution (CBS) and the number of iterations. This helps OSAM to converge faster and find the near-optimal solution. However, OSAM only considers one objective (makespan) and one constraint (budget) in a single cloud environment.

Li et al. [69] proposed a new method called PSO + LOA, which combines two optimization techniques: particle swarm optimization (PSO) and lion optimization algorithm (LOA). The method reduces the time needed to execute workflow tasks in cloud computing while staying within a budget limit. The process uses several features, such as a distance-based particle repositioning algorithm, an adaptive search strategy, and a balance between exploration and exploitation, to improve the efficiency and effectiveness of the optimization process. The paper tested the method on large-scale workflow applications and showed that the method performed better than previous methods regarding solution quality and budget constraints.

To increase efficiency, the authors in [47] have gone to the grey wolf algorithm and particle swarm optimization to optimize the time parameters and have faster convergence. In [70], Shirvani has considered a different solution from other researchers, which uses parallel processing to optimize the time parameters. This research has presented a hybrid optimization algorithm based on particle swarm optimization, which, like [47] and faster convergence with parallel processing, also improves the system's overall performance.

Javanmardi et al. [71] suggested a method for scheduling complex applications on IoT systems that need to optimize both the security and the performance simultaneously, which uses fuzzy-based anomaly detection algorithms to find and block the source of attacks from malicious requestors. They also used an NSGA-III scheduler optimization method to balance the load and the delay for resource management. The paper also considers the security aspect of the scheduling problem, as IoT devices and fog nodes may be exposed to various attacks, such as data tampering, eavesdropping, or denial of service. This method has four main steps: task clustering, task mapping, task scheduling, and task migration. Task clustering is the step of grouping the tasks of an application into clusters based on their similarity and dependency. Task mapping assigns the clusters to the fog nodes based on their resource requirements and security levels—task scheduling orders tasks within each cluster based on their priority and dependency. Task migration is moving tasks from one fog node to another in case of resource shortage or security breach. This method can improve the performance and cost of running complex applications on SDN-based IoT-Fog networks by using a secure workflow scheduling method, which is a technique that assigns and orders the tasks of an application to the available resources while satisfying the quality-of-service constraints, such as deadline, throughput, and reliability.

A group of articles have addressed the parameters that have been paid attention to in less research. The approach of the study [72] is slightly different from other articles, and by using the combination of the optimal algorithm of the Sari particle swarm and also the ant-lion algorithms, it has been able to manage time parameters and solve the problem of privacy and security, which is one of the required quality parameters. Services, tasks, and workflows are introduced, and a data encryption technique has been applied to provide a more secure framework. However, it has been used for austere cloud environments, and the parameters related to multi-cloud environments should also be included.

Machine learning (ML), especially reinforcement learning and deep reinforcement learning, presents promising capabilities for cloud workflow scheduling. For instance, Wang et al. in [73] proposed a deep reinforcement learning-based optimization scheduling algorithm for solving workflow scheduling problems with the aim of load-balancing and reducing response in edge and fog platforms. Since miscellaneous IoT applications request low-time computing resources, their load is forwarded toward edge and fog servers. To have load balance and minimum response time, the deep learning-based reinforcement algorithm was extended to have adaptive and flexible task scheduling commensurate with the underlying heterogeneous distributed platform. Deep Q-learning empowers scheduling agents to glean effective strategies from past workflow executions and resource utilization patterns [74]. ML methods aid in optimizing objectives like time and cost while adhering to constraints concerning budget, deadlines, and reliability.

Applying ML to scheduling poses challenges such as sample complexity, hyperparameter calibration, and managing sizeable discrete action spaces. Integrating metaheuristics and ML with simulation-based training can mitigate these hurdles. ML holds substantial promise in

facilitating adaptive, resilient scheduling while complementing human expertise.

Reducing energy consumption is one way to lower cloud providers' costs. Intelligent task-scheduling algorithms can help to assign user-deployed jobs to servers in an energy-efficient way. Hunter Plus [53] is a new CNN-based resource scheduling method that builds on the existing GGCN scheduler (HUNTER) and develops a new CNN scheduling model. The authors aimed to optimize makespan, cost, energy consumption, and throughput. The CNN model also exhibits stable behaviour by evenly and consistently allocating and migrating tasks. It can handle large-scale and complex task-scheduling problems with multiple objectives and constraints. In addition to energy, Saif et al. [75] have also worked on delay, and similar to the previous paper, it has used a multi-objective function to cover users' and service needs, and it can cover constraints at a large-scale level.

A new method called WDDQN-RL was introduced by Li et al. [76] to schedule many workflows in the cloud while reducing both the time and the cost. The technique uses a WDDQN, a weighted version of a double deep Q-network, to avoid errors in estimating the value of each action that can happen in DQN and DDQN. The method also uses pointer networks to deal with different sizes of task sets that can be chosen and a dynamic sensing mechanism to change the focus on each objective depending on the current situation. The method can correct the errors in estimating the value of each action that can happen in DQN and DDQN, making the learning process more stable and accurate.

Chen et al. [77] suggested a method for scheduling different types of workflows in cloud computing using deep reinforcement learning. This method can deal with the problem of poor service quality caused by the lack of coordination among different types of workflows and the interruption of task execution in cloud computing situations. It can adjust to edge cloud systems' changing and uncertain environment by using deep reinforcement learning, which can learn from online feedback and update its policy accordingly. It can weigh the trade-offs between performance and cost using a multi-objective reward function, which considers the execution time, energy consumption, and monetary cost of different types of workflows. It can manage the heterogeneity and diversity of workflows and resources using a collaborative scheduling strategy. It splits the different workflows into sub-workflows and assigns them to various agents, each in charge of scheduling a sub-workflow on a subset of resources. This method has different steps. First, it extracts the structure and time sequence features for the dynamic scheduling process and builds a reasonable feature set to support the scheduling decision. Second, it designs a time-step adaptive scheduling mechanism to reduce redundant information in the scheduling process and enables the agent to achieve efficient learning. In addition, it uses equilibrium, priority, and preference scheduling strategies, a compound reward mechanism that combines immediate and delayed rewards, and a hybrid action that switches between scheduling and waiting to harmonize the agent's learning objectives and actual scheduling requirements.

Parameter-based approaches typically denote approaches where parameters or mathematical relationships characterize the problem. Mathematical methods, such as linear programming, address problems through mathematical relationships or optimization, while parametric methods involve managing or improving a specific parameter, such as energy or cost. They often solve the problem and the desired parameter without relying on a specific algorithm. We have clarified that articles considering mathematical methods and parameter-based approaches fall under the umbrella of parametric methods.

Mathematical optimization, also referred to as programming (MP) and mentioned as parametric modelling, serves as a valuable tool for tackling intricate problems involving an objective function and mathematical constraints. This approach simplifies decision-making processes by helping determine the best choice from a range of options. Applied mathematics plays a role in this field, enabling individuals to identify the optimal solution within given constraints. Mathematical programming finds application, in addressing planning and scheduling challenges optimizing resource utilization efficiently to achieve specific objectives within reasonable time frames. Various techniques, including Integer Linear Programming (ILP) and Mixed Integer Linear Programming (MILP), are employed to achieve these goals [78, 79].

Chakravarthi et al. [80] proposed a method called NRBWS for scheduling workflows in cloud computing, where the goal is to optimize the reliability and the time of the workflow execution under a given budget limit. A workflow is a set of tasks that depend on each other and must be executed in a specific order. NRBWS uses a–max normalization process and a calculation of the expected reasonable budget (erb) to choose the best resource for each task. NRBWS also considers the reliability of the resources and the tasks and assigns the tasks to the most reliable resources with the shortest finish time within the budget. Cost parameters must be extended to different parameters like energy consumption.

Xie et al. [81] proposed a new two-stage multi-population genetic algorithm with heuristics for workflow scheduling. This study presented a mathematical model for workflow scheduling in Heterogeneous Distributed Computing Environments, a complete, solvable, and extensible integer programming problem with constraints on the order

and the resources of the tasks. This algorithm uses a two-stage multi-population coevolution method with novel techniques for initializing, operating, decoding, and improving the populations. The algorithm is applied to various situations based on accurate and random workflows to test its effectiveness. This study only considers the static scheduling of workflow, where the exact task execution time and communication time for any valid task-resource assignment are known before scheduling.

By presenting a multi-objective algorithm, Hussain et al. [63] proposed a method for scheduling workflow tasks to different servers in cloud data centers, considering cost, time, and energy consumption challenges. The DEWS method has four steps: ordering the tasks, finding the best data center, adjusting the task order, and finding the best virtual machine. The method also uses a technique called DVFS, which can adjust the voltage and frequency of the servers to reduce the energy cost. The technique considers data centers that are located in different places.

Xu et al. [82] presented a new problem model and simulator for Dynamic Workflow Scheduling in Fog Computing (DWSFC). The authors proposed a new Multi-Tree Genetic Programming (MTGP) method that can generate scheduling heuristics and make real-time decisions on different decision points. The MTGP method with multiple trees can deal with both routing and sequencing decision points at the same time. However, this method may not account for the reliability and security issues of the fog nodes, which may impact the quality of service.

Table 3 below is a general review of the articles studied in this article, and in the discussion section, we will discuss their statistics and answer the research questions.

8 Discussions

This section comprises an analytical discussion of the existing approaches to workflow scheduling in cloud and fog environments. The analytical reports are based on our questions:

RQ1 How do the different methods of scheduling workflows in a fog-cloud environment work, and what are their categories?

Figure 9 shows a statistical view of the work scheduling approaches based on our taxonomy. We proposed two work scheduling approaches classification: heuristic workflow scheduling and hybrid workflow scheduling. Hybrid workflow scheduling has the highest percentage of the scheduling approaches, with 75% based on selected papers analyses. As expected, given that the articles of recent years have been selected, the use of artificial intelligence methods in research is more. In response to the next question, techniques and algorithms are categorized.

RQ2 Which of the various workflow scheduling algorithms and datasets that have been suggested has been more popular?

In Figure 10, we present a method distribution derived from current papers focused on workflow scheduling in cloud and fog environments. Through a meta-analysis of 14 research studies, it was observed that authors frequently utilize a blend of diverse heuristic algorithms to address crucial parameters in resource allocation and workflow planning. The analysis provides insights into the algorithms mentioned in the examined articles, revealing a diverse landscape. Algorithms are mentioned in the analysis written for the articles.

Figure 11 depicts the composition of workflow scheduling datasets. It is divided into two categories: “Real” and “Random”. Real datasets account for a larger share than randomly selected datasets, constituting 55.1% of the pie chart.

RQ3 What tools have been employed to simulate and implement the methods that have been studied?

In our selection process, we specifically focused on articles that addressed distributed environments across various layers, including fog, cloud, and edge computing. This deliberate selection allowed us to analyze the papers based on the environments in which they were evaluated, thus identifying both the challenges associated with real-time implementation and making the simulation environment and tools more transparent for future solutions.

Figure 12 provides a detailed illustration of the tools utilized for simulation and implementation across different environments. Through this analysis, we aim to shed light on the methodologies and technologies employed in evaluating the effectiveness and feasibility of workflow scheduling methods in diverse computing environments.

Based on Figure 13, it is evident that 29% of the method simulations were conducted using the WorkflowSim tool. WorkflowSim is designed explicitly for simulating workflows in distributed environments and is built as an extension of CloudSim, a well-known cloud computing simulation tool. Additionally, 25% of the papers utilized the CloudSim tool for simulation purposes, although specific details were not provided.

Table 3 provides a comprehensive breakdown of the simulation tools used in each article, allowing for a detailed comparison of the tools employed across different studies. This information enhances transparency and facilitates a deeper understanding of the methodologies adopted in the simulation and implementation of workflow scheduling methods in various computing environments.

Table 3 General analysis of selected papers

Papers	Advantage	Disadvantage	Simulation tools	Environment
[56]	Worked on multi-cloud methods	Other distributed environments must be considered	CloudSim	Cloud
[43]	A multi-objective method is designed for performance and cost	Reliability and security must be considered	CloudSim	Cloud
[57]	Worked on scalability and time-constraint problems	Complexity	WorkflowSim in CloudSim	Cloud
[81]	Cover heterogeneous distributed computing environments	Complexity	Not considered	Cloud-Fog (distributed)
[66]	By using dynamic methodology, it can adapt to the changing and uncertain environment	They must identify the trade-off between performance and cost	CloudSim	Cloud
[71]	Resource allocation optimization, security improvement	Computational overhead	Not considered	Fog in IoT
[58]	The iterative forward–backward scheduling can further improve the individual by iteratively scheduling the workflow tasks from the beginning to the end and from the back to the beginning while considering the resource availability and the task dependency	They used a GA-based algorithm but in static workflow scheduling	C + +	Cloud
[65]	Search efficiency improvement and the communication overhead optimization	High complexity	IFogSim	Fog in IoT
[83]	They proposed scheduling time constrained Workflows on hybrid Fog/Cloud environments	Workflow scalability must be considered	Velociraptor simulator-Python	Fog/Cloud
[44]	Worked on scalable workflow scheduling	The cost must be considered	CloudSim	Edge cloud
[60]	Worked on Edge and Cloud simultaneously	Complexity and overhead	WorkflowSim in CloudSim	Edge cloud
[48]	Focused on energy efficiency in different workflow scheduling	The cost must be considered for cloud and fog servers	CloudSim	Fog in IoT
[73]	Complex and stochastic optimization for workflow scheduling	Complexity for computation	Python	Fog-Cloud
[47]	They covered different characteristics of heterogeneous environments	Real-time workflow scheduling and user goals must be considered	WorkflowSim in CloudSim	Fog-cloud
[42]	This research has performance and energy improvement simultaneously	Communication cost and delay between the fog and cloud nodes must be considered for time parameters	IFogSim	Fog-cloud
[46]	They used a dynamic multi-objective algorithm	The paper assumes that the cloud resources are homogeneous and have fixed prices and capacities. This assumption may not be realistic in practice, as cloud resources vary in their types, prices, and availability	WorkflowSim in CloudSim and Java	Cloud
[40]	Dynamic objectives can help the algorithm balance the trade-offs among the goals and achieve better performance	The paper does not consider the communication cost and delay between the container-based tasks and the cloud-edge resources, which may affect the performance and reliability of the workflow execution	Python	Edge cloud
[63]	A heuristic approach that considers both the energy consumption and the deadline of the workflow tasks	Fixed execution time and resource requirements are assumed for workflows	Java	Cloud
[49]	They designed a dynamic objective process to help the algorithm balance the trade-offs among the objectives and achieve better performance	Cost and scalability must be considered	CloudSim	Cloud in IoT
[69]	Time parameters are minimized	The cost must be considered	Java	Cloud
[46]	Cost optimization based on environmental parameters modeling	Reliability and energy consumption must be considered	MATLAB	Cloud

Table 3 (continued)

Papers	Advantage	Disadvantage	Simulation tools	Environment
[47]	Faster convergence	Energy consumption must be considered	WorkflowSim in CloudSim	Cloud
[70]	Parallel workflow scheduling	Cost optimization must be considered	WorkflowSim in CloudSim	Cloud-Fog (distributed)
[72]	The multi-objective method with reliability consideration	Multi-cloud parameters must be added	CloudSim	Cloud
[30]	Multi-objective and multi-layer coverage	Some Quality of Services must be evaluated in real-world scenarios	WorkflowSim in CloudSim	Cloud-Fog (IoT)
[62]	Better convergence	Complexity on a large scale in real-time scenarios	CloudSim	Cloud
[59]	Work on different sizes of workflows	The cost must be considered in the evaluation	TensorFlow framework	Cloud
[41]	Multi-objective based on cost, energy consumption, and deadline constraint	Complexity in real data must be considered and evaluated	Java	Cloud
[68]	A new heuristic method is used for cost management	It is a single objective method for a single cloud environment	WorkflowSim in CloudSim	Cloud
[45]	The multi-objective method is designed based on resource allocation requirements in a distributed system	Hybrid and multiple clouds are not covered in the real environment	Java	Cloud
[67]	Scalability and workflow variety coverage	Computation costs must be considered	CloudSim	Cloud
[80]	It uses a hybrid approach to manage cost and reliability. It improved QoS in different areas	The cost factor must be considered	CloudSim	Cloud
[59]	Different parameters of cost are considered	The fault must be considered based on the resource allocation method in future work. Complexity of simulation	CloudSim	Cloud
[61]	Multi-objective model for different requirements	Complexity in simulation	Java	Cloud
[55]	Resource utilization improvement	It only focused on the makespan and the cost	CloudSim and MATLAB	Edge cloud
[61]	A method for complex workflow scheduling	Scalability must be considered	WorkflowSim	Cloud
[28]	Prioritizing the tasks based on different parameters	Computational time and resources must be considered	CloudSim	Cloud
[53]	Multiple object functions based on different requirements of users	Scalability and reliability must be considered	COSCO framework	Fog-cloud
[75]	Delay constraints added to energy management for cost-aware workflow scheduling	The other parameters that are related to cost must be considered	MATLAB	Fog-cloud
[82]	Multi-objective are formulated	Complexity must be considered	WorkflowSim	Fog-cloud
[77]	Optimizes multi-objective scheduling makespan, cost, fairness, and continuity	Potentially limited to smaller workflows and static environments	Python	Cloud
[50]	Handling large-scale and complex task scheduling problems with multiple objectives and constraints	Security and reliability must be considered	iFogSim-CloudSim	Fog-cloud
[51]	Reducing the latency and bandwidth consumption	Several data centers must be considered	WorkflowSim	Fog-Cloud
[52]	Handling real-time interactive services with time constraints and reliability requirements	Cost and energy consumption must be considered	iFogSim-CloudSim	Fog-Cloud
[36]	Handling large-scale and complex task-scheduling problems	Reliability and security issues must be considered	WorkflowSim	Fog-Cloud
[37]	It is multi-objective formulated for large-scale tasks	Load balancing must be considered	SMO-WorkflowSim	Fog
[64]	Performance and availability for emergency services	Cost and energy consumption must be considered	iFogSim	Fog
[2]	Predictive energy-efficient scheduling and optimized workflow efficiency	Implementation complexity, computational overhead, and limited validation scope	iFogSim, MATLAB	Fog

Table 3 (continued)

Papers	Advantage	Disadvantage	Simulation tools	Environment
[84]	Efficient workflow scheduling outperforms existing methods	Limited scalability discussion; potential complexity in diverse scenarios	Python	Fog-Cloud
[85]	Enhanced adaptability and efficient resource utilization	The potential trade-off between objectives requires careful consideration of makespan and resource utilization	CloudSim	Cloud
[86]	Optimizing execution time and communication costs	Limited empirical validation and comparison, potentially restricting generalizability	none	Cloud
[87]	Optimizes real-time cloud workflow scheduling, surpassing existing algorithms in response time, success rate, and cost	Limited scalability to large-scale cloud environments and potential underrepresentation of overall performance	Python	Cloud
[88]	Proposes an effective deadline-constrained and cost-efficient cloud workflow scheduling algorithm	Lacks evaluation on large-scale	CloudSim	Cloud
[89]	minimize makespan, execution time, energy consumption, and cost	Unclear generalizability and need for more discussion on security/reliability	FogBus2	Cloud
[90]	Multi-objective algorithm for cloud workflow scheduling, effectively balancing user preferences with conflicting interests	Limited to single workflows and cloud environments, needing more scalability for handling multiple workflows or cloud scenarios	WorkflowSim	Cloud
[91]	Highly effective and robust algorithm that minimizes costs for the considered problem	Limited scope and scalability	WorkflowSim	Cloud
[92]	Optimizes workflow scheduling for both users and providers in Fog-Cloud using distributed agents and fuzzy logic	It lacks real-time and QoS sensitivity and does not offer user-specific prioritization	WorkflowSim, Fuzzy logic toolbox	Fog-Cloud
[93]	Achieves better results in convergence, diversity, and critical metrics like makespan, cost, and energy	Increased execution time and complexity	MATLAB	Fog
[94]	Faster convergence, better accuracy, and Energy-efficient scheduling for real-world workflows	Tuning-hungry, potentially limited to specific workflows, needs more proof against other hybrids	IoTSim-Osmosis	IoT-Fog
[95]	Balances energy consumption and task completion time	High complexity for dense workflows	WorkflowSim	Multi-cloud
[96]	achieving better results for makespan, cost, energy, and load balancing	Higher computational cost	fog workflow sim	Cloud
[97]	Reduces cost under deadline constraints and adapts to cloud environment fluctuations	Higher scheduling time	MATLAB	Fog-Cloud
[98]	Reduces execution time and cost	Limited in considering load balancing, fault tolerance, diverse workflow types, and user satisfaction	FogWorkflowSim	Fog-Cloud
[99]	Efficient cloud workflow scheduling with reduced cost and execution time, outperforming existing methods on real-world datasets	Limited to addressing data center power consumption and geographically distributed environments	Java	Cloud
[100]	Significant performance improvement in response time, resource utilization, and web service combination	Limited exploration of effectiveness for different workflow sizes	WorkflowSim	Cloud
[101]	efficient workflow scheduling with minimized task execution time	Unclear how it handles real-world dynamic uncertainties	WorkflowSim, CloudSim	Cloud
[17]	Energy efficiency and fast execution	Needs reliability considerations	D-JStorm	Cloud
[3]	Better performance in terms of makespan and energy consumption	high complexity	CloudSim	Cloud
[102]	Improves efficiency and solution quality	Limited testing, security considerations, and future work plans could be more specific	iFogSim	Fog
[103]	Reduces energy consumption and cost through task clustering, deadline constraints and DVFS for energy efficiency	Relies on accurate execution time estimation and may not consider all relevant parameters in real-world environments	CloudSim	Cloud

Table 3 (continued)

Papers	Advantage	Disadvantage	Simulation tools	Environment
[104]	Multiple objectives and energy efficiency	Computationally expensive	CloudSim	Cloud
[105]	Achieves better cost and makespan through a hybrid approach, balancing workload and diverse population	Higher complexity	WorkflowSim	Cloud
[106]	Optimizes execution time, cost, and energy consumption	Lacks security and privacy considerations, and effectiveness compared	WorkflowSim	Cloud
[107]	Improved performance, balanced exploration/exploitation, real-world applicability, and resource efficiency	Limited real-world testing, parameter tuning challenge, specific problem focus, and limited comparison to existing solutions	CloudSim	Cloud
[108]	Outperforms existing methods in energy, time, and throughput, offering platform adaptability and balanced multi-objective optimization	Lacks real-world validation and communication cost consideration, potentially favoring throughput reduction in specific cases	CloudSim	Cloud
[109]	Optimizes cost & time simultaneously for improved resource utilization	Limitations in generalizability, complexity, and specific drawbacks	CloudSim	Cloud
[110]	Uncertainty-aware optimization	Limited applicability in constrained scenarios and increased computational demands	Python	Cloud
[111]	Better cost, compilation time, and constraint satisfaction	Slow convergence, limited evaluation on diverse datasets and goals	WorkflowSim	Cloud
[112]	Reduces execution time and cost	Prone to local optima, slow convergence, needs improvement for complexity	Python	Cloud

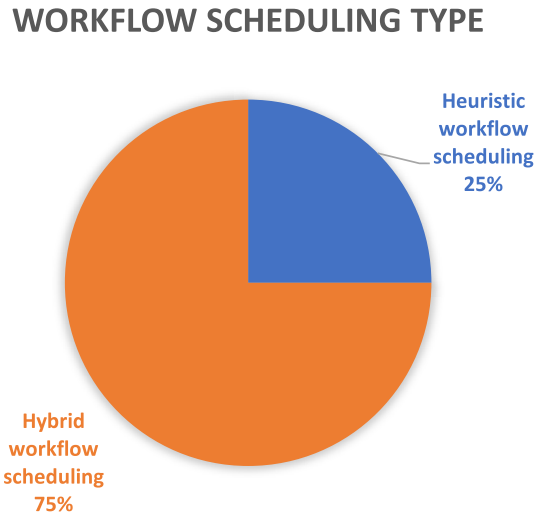


Fig. 9 Method types of distribution based on taxonomy

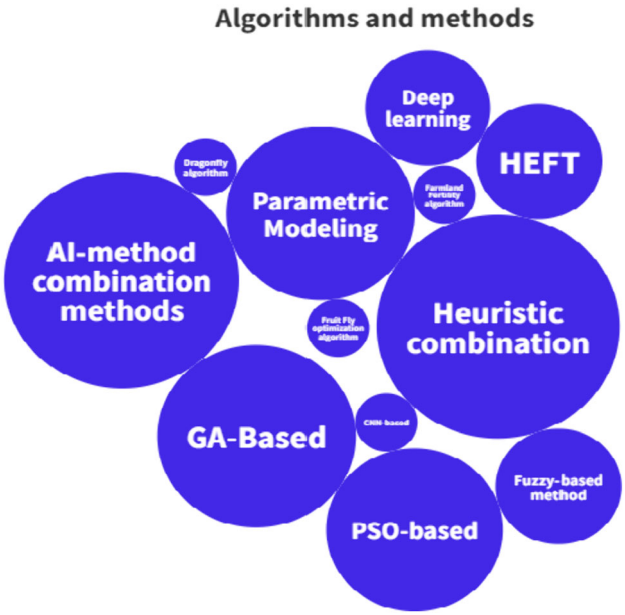


Fig. 10 Algorithms and approaches in selected paper

RQ4 What parameters are used to assess the performance of workflow scheduling methods in a fog-cloud environment?

Table 4 categorizes the essential parameters for scheduling workflows and tasks as outlined in each article, providing a comprehensive overview of the factors considered in workflow scheduling methodologies. Meanwhile, Fig. 14 presents the percentage of parameter usage across all the analyzed articles, shedding light on the relative importance of each parameter in the context of workflow scheduling.

Utilizing various cloud and fog-based architectures in innovative city applications underscores the critical importance of these architectures in serving communities effectively. Furthermore, the inherent resource heterogeneity emphasizes the significance of parameters related

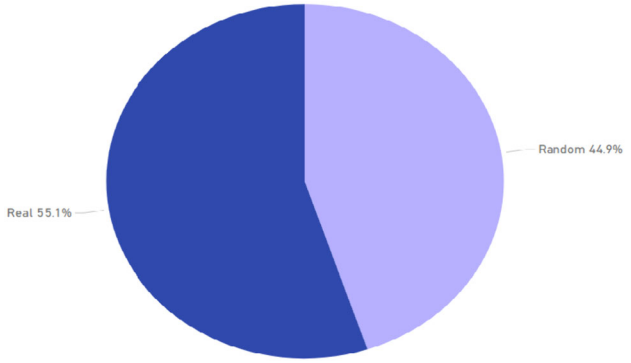


Fig. 11 Distribution of real and random datasets in workflow scheduling research

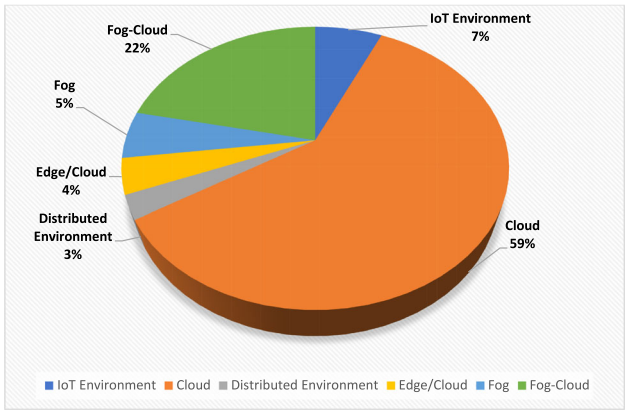


Fig. 12 Simulation environment in selected papers

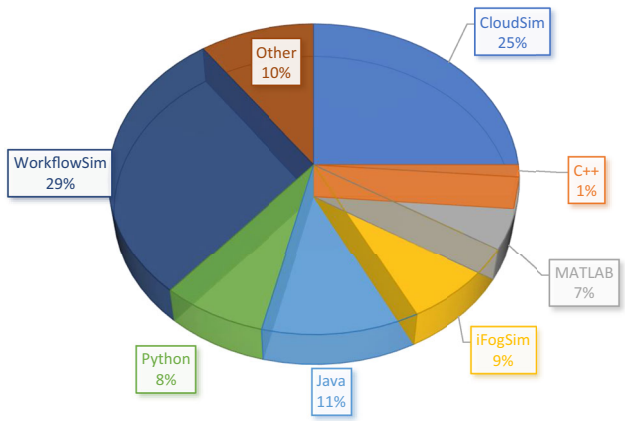


Fig. 13 Simulation tools on selected papers

to performance and makespan, reflecting their pivotal role in optimizing workflow scheduling. Additionally, parameters such as energy consumption and cost emerge as critical considerations in method design, reflecting the imperative to utilize resource capacities while minimizing operational costs efficiently.

RQ5 What are the future challenges, open issues, and research gaps for workflow scheduling in a fog-cloud environment?

According to the studies, a group of challenges are listed:

- ✔ Resource heterogeneity: Workflow scheduling faces difficulties when dealing with resource heterogeneity, which refers to resources that differ in processor speed, memory capacity, bandwidth, cost, availability, and even structural differences depending on the architectural layer. Because of this, assigning jobs to resources in an optimal or nearly optimal way that will meet workflow goals and restrictions like decreasing makespan, cost, or energy usage can be challenging. Due to resource heterogeneity, scheduling algorithms must be robust and flexible to manage the unpredictability and fluctuation of resource availability and performance.
- ✔ Resource dynamism: Workflow scheduling is made more difficult by resource dynamism, which implies that resources' performance, cost, and availability can all fluctuate over time. Because of this, assigning jobs to resources in an optimal or nearly optimal way that will meet workflow goals and restrictions like decreasing makespan, cost, or energy usage can be challenging. Resource dynamism also calls for resilient and adaptive scheduling algorithms that can manage the ambiguity and unpredictability of resource conditions and modify scheduling choices as necessary.
- ✔ Resource utilization: Workflow scheduling is complicated by resource utilization, which measures how sound resources are employed to carry out process activities. How resources are used can impact the workflow's efficiency, affordability, energy usage, and the contentment of both users and providers. Therefore, it might be challenging to establish an optimal or nearly optimal task-to-resource assignment that maximizes resource use, particularly in dynamic and heterogeneous contexts like cloud and fog computing [114].
- ✔ Conflicting objectives: It is a challenge in workflow scheduling since different goals go to be optimized simultaneously, but they are frequently inconsistent or incompatible. For case, minimizing the execution time and minimizing the execution fetched of a

Table 4 The main parameters in the selected paper

References	Makespan	Cost	Fairness	Continuity	Energy	Response Time	Network Utilization	Scheduling Time	Wait Time
[2]	*	*		*	*				
[3]	*	*			*				
[56]		*				*			*
[43]	*	*							
[57]		*							
[77]	*	*	*	*					
[66]	*				*				
[71]						*	*		
[58]	*								
[65]	*								
[44]	*					*		*	
[60]	*	*			*				
[48]		*			*				
[73]						*			
[47]	*	*							
[49]		*							
[40]	*				*				
[63]		*							
[69]	*	*							
[46]	*	*						*	
[47]	*								
[70]									
[72]	*	*			*				
[30]	*	*				*			
[28]	*								
[61]	*	*						*	
[59]	*				*	*			
[67]		*						*	
[45]	*	*							
[41]	*	*							
[76]	*	*							
[62]	*				*				
[68]	*	*							
[55]	*	*						*	
[61]	*	*	*			*			
[53]		*			*				
[75]		*			*	*			*
[50]	*	*			*			*	
[51]	*	*			*				*
[52]						*			
[36]	*				*	*		*	
[37]		*				*		*	
[64]								*	
[80]									
[81]	*						*		
[82]	*	*				*			
[83]	*				*				
[84]	*	*			*				
[85]	*	*				*			

Table 4 (continued)

References	Makespan	Cost	Fairness	Continuity	Energy	Response Time	Network Utilization	Scheduling Time	Wait Time
[86]	*	*			*				
[87]		*			*	*			*
[88]	*	*	*	*	*				
[89]	*	*			*				
[90]	*	*			*				
[91]	*	*							
[92]	*	*							
[93]	*	*			*				
[94]	*	*			*				
[95]	*	*			*				
[96]	*	*			*				
[97]	*	*						*	
[98]	*	*							
[99]	*	*							
[100]		*				*		*	
[101]	*								
[17]	*	*			*				
[102]	*	*			*				
[103]	*	*			*				
[104]	*	*			*	*		*	
[105]	*	*			*				
[106]	*	*			*	*			
[107]	*	*			*				
[108]		*			*				
[109]	*	*			*	*		*	
[110]	*	*							
[111]	*	*			*				
[112]	*	*			*				
[113]	*	*			*				

References	Deadline	Throughput	Imbalance Degree	Resource Utilization	Violation Cost	Communication Cost	Speed up	Efficiency	Reliability
[2]				*		*		*	
[3]							*	*	
[56]								*	*
[43]								*	
[57]									
[77]									
[66]									
[71]									
[58]									
[65]	*								
[44]									
[60]							*	*	
[48]									
[73]						*			
[47]			*						
[49]									
[40]			*						

Table 4 (continued)

References	Deadline	Throughput	Imbalance Degree	Resource Utilization	Violation Cost	Communication Cost	Speed up	Efficiency	Reliability
[63]	*			*					
[69]									
[46]									
[47]									
[70]							*	*	
[72]			*						*
[30]									*
[28]							*	*	
[61]									
[59]									
[67]									
[45]									
[41]									
[76]									
[62]									
[68]									
[55]						*			
[61]									
[53]		*							
[75]									
[50]									
[51]						*			
[52]	*								*
[36]									
[37]									
[64]						*		*	
[80]									*
[81]									
[82]				*				*	
[83]									*
[84]	*			*				*	
[85]				*		*		*	
[86]						*		*	*
[87]									*
[88]	*							*	
[89]	*			*				*	*
[90]	*			*			*	*	
[91]	*							*	*
[92]	*							*	*
[93]		*					*	*	*
[94]						*	*	*	*
[95]	*					*			
[96]				*		*	*	*	*
[97]	*	*		*				*	*
[98]	*			*				*	
[99]	*							*	*
[100]	*			*			*		*
[101]							*		

Table 4 (continued)

References	Deadline	Throughput	Imbalance Degree	Resource Utilization	Violation Cost	Communication Cost	Speed up	Efficiency	Reliability
[17]	*	*				*			
[102]	*					*		*	*
[103]	*	*					*		*
[104]	*			*		*	*	*	
[105]	*							*	*
[106]	*			*					
[107]							*	*	*
[108]		*		*				*	
[109]	*	*		*		*		*	*
[110]	*							*	
[111]	*			*					
[112]						*			
[113]					*				

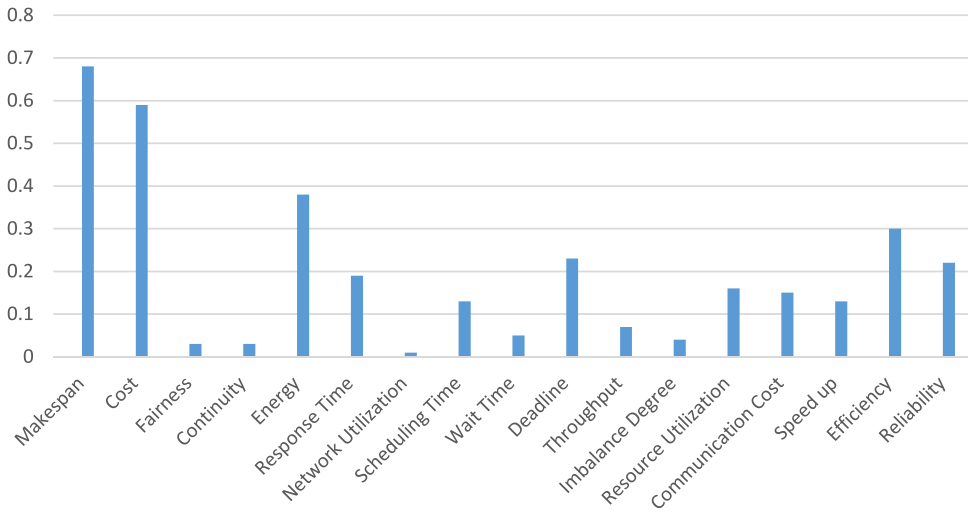


Fig. 14 distribution on different parameters in selected papers

workflow are two common targets, but they - more often than not require distinctive trade-offs and compromises. Hence, finding an ideal or near optimal task of errands to assets that can satisfy all the goals and imperatives of the workflow may be a troublesome issue, particularly in heterogeneous and energetic situations such as cloud and fog computing.

✔ Scalability: it is critical for workflow scheduling since it influences the execution, productivity, and unwavering quality of the framework. The complexity and measure of the workflows, which may include hundreds or thousands of assignments with different conditions, imperatives, and prerequisites. The heterogeneity and dynamism of the assets, which may have diverse characteristics and accessibility, take a toll and may alter over time due to disappointments, vacillations, or competition. The trade-off between clashing destinations may require distinctive trade-offs and compromises among the benefits quality, asset utilization, and client fulfilment. Hence, finding an ideal or near-optimal task of assigning assets that can fulfil all the targets and limitations of the workflow and scale well with the increasing workload and asset pool

may be a challenging issue, particularly in heterogeneous and energetic situations such as cloud and fog computing [66].

- ✔ Complexity: Complexity is a challenge in workflow scheduling because it refers to the difficulty of finding an optimal or near-optimal solution to an NP-hard workflow scheduling problem. Complexity is influenced by several factors, such as the size and structure of the workflow, the number and characteristics of resources, the workflow's goals and constraints, and the environment's uncertainty and variability. Therefore, finding effective and efficient scheduling algorithms to address the complexity of the workflow scheduling problem is challenging, especially in heterogeneous and dynamic environments such as cloud computing and fog.
- ✔ Estimation: Time estimation for tasks and workflow is a challenge in workflow scheduling because it requires predicting the completion time of each task and the entire workflow based on available information about tasks, resources, and the environment. It is challenging to find accurate and robust time estimation methods that can cope with the complexity, variability, heterogeneity, dynamics, uncertainty, and randomness of working on flow scheduling problems, especially in heterogeneous and dynamic environments such as cloud computing and fog.
- ✔ Research gaps: Lack of scheduling mechanisms optimized for emerging edge computing paradigms like cloudlets and nano data centers. Existing work focuses on cloud/fog environments. More research is needed on scheduling for serverless platforms and Functions as a Service (FaaS), which have different constraints. There is a need for a more extensive evaluation of real-world traces and workloads at scale rather than simulations. Insufficient attention to incremental deployment in brownfield environments alongside legacy systems. Lack of standardization around APIs, abstractions, and interfaces for cross-platform workflow management. Shortage of solutions focused on industry verticals like healthcare, smart cities, and augmented reality.

9 Conclusions

Workflow scheduling is an essential problem in distributed environments, especially in cloud and fog computing. Many researchers have tackled this problem using artificial intelligence-based solutions. We decided to review the recent literature (from the recent four years) on the methods and techniques used in these environments. This paper provides a conceptual overview, a classification, and a taxonomy of the methods, algorithms, and architectures for workflow management in cloud and fog environments. We also discuss the future directions of this field. We found that there are many algorithms for workflow scheduling, and they vary in the factors and parameters they consider for scheduling. We examine these factors and their related challenges and issues, such as resource utilization, performance metrics, and cost management. We also observed that workflow scheduling is an NP-hard problem, so many researchers used heuristic algorithms or hybrid approaches that combine artificial intelligence and modelling. We categorized some properties that pose challenges and open issues in workflow scheduling. Considering these aspects can enhance the quality and flexibility of workflow scheduling methods.

Author contributions NK: Conceptualization, methodology, Writing—original draft preparation, validation. MV: Supervising, Data curation, writing—reviewing and editing, validation. SA: Visualization, investigation. MHS: Writing—reviewing and editing, validation.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data availability The dataset used and analyzed during the current study is available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Stavrinides, G.L., Karatza, H.D.: A hybrid approach to scheduling real-time IoT workflows in fog and cloud environments. Multimed. Tools Appl. **78**, 24639–24655 (2019)
2. Nazeri, M., Soltanaghaei, M., Khorsand, R.: A predictive energy-aware scheduling strategy for scientific workflows in fog computing. Expert. Syst. Appl. **247**, 123192 (2024)
3. Xia, X., Qiu, H., Xu, X., Zhang, Y.: Multi-objective workflow scheduling based on genetic algorithm in cloud environment. Inform. Sci. **606**, 38–59 (2022)

4. Noorian Talouki, R., Hosseini Shirvani, M., Motameni, H.: A hybrid meta-heuristic scheduler algorithm for optimization of workflow scheduling in cloud heterogeneous computing environment. J. Eng., Design Technol. **20**(6), 1581–1605 (2022)
5. Keshanchi, B., Souri, A., Navimipour, N.J.: An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: formal verification, simulation, and statistical testing. J. Syst. Softw. **124**, 1–21 (2017)
6. Durillo, J.J., Nae, V., Prodan, R.: Multi-objective energy-efficient workflow scheduling using list-based heuristics. Future Gener. Comput. Syst. **36**, 221–236 (2014)
7. Kaur, S., Bagga, P., Hans, R., Kaur, H.: Quality of Service (QoS) aware workflow scheduling (WFS) in cloud computing: a systematic review. Arab. J. Sci. Eng. **44**, 2867–2897 (2019)
8. Hassan, H.O., Azizi, S., Shojafar, M.: Priority, network and energy-aware placement of IoT-based application services in fog-cloud environments. IET Commun. **14**(13), 2117–2129 (2020)
9. Ahmad, Z., et al.: Scientific workflows management and scheduling in cloud computing: taxonomy, prospects, and challenges. IEEE Access **9**, 53491–53508 (2021)
10. Hilman, M.H., Rodriguez, M.A., Buyya, R.: Multiple workflows scheduling in multi-tenant distributed systems: a taxonomy and future directions. ACM Comput. Surv. (CSUR) **53**(1), 1–39 (2020)
11. Yassir, S., Mostapha, Z., Claude, T.: Workflow scheduling issues and techniques in cloud computing: a systematic literature review. Cloud Comput. Big Data: Technol., Appl. Secur. **3**, 241–263 (2019)
12. Versluis, L., Iosup, A.: A survey of domains in workflow scheduling in computing infrastructures: community and keyword analysis, emerging trends, and taxonomies. Future Gener. Comput. Syst. **123**, 156–177 (2021)
13. Hosseinzadeh, M., Ghafour, M.Y., Hama, H.K., Vo, B., Khoshnevis, A.: Multi-objective task and workflow scheduling approaches in cloud computing: a comprehensive review. J. Grid Comput. **18**, 327–356 (2020)
14. Kumar, Y., Kaul, S., Hu, Y.-C.: Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: state-of-the-art survey. Sustain. Comput.: Inform. Syst. **36**, 100780 (2022)
15. Menaka, M., Kumar, K.S.S.: Workflow scheduling in cloud environment–challenges, tools, limitations & methodologies: a review. Meas.: Sens. **24**, 100436 (2022)
16. Masdari, M., ValiKardan, S., Shahi, Z., Azar, S.I.: Towards workflow scheduling in cloud computing: a comprehensive analysis. J. Netw. Comput. Appl. **66**, 64–82 (2016)
17. Ahmed, O.H., Lu, J., Xu, Q., Ahmed, A.M., Rahmani, A.M., Hosseinzadeh, M.: Using differential evolution and moth-flame optimization for scientific workflow scheduling in fog computing. Appl. Soft Comput. **112**, 107744 (2021)
18. Hoseiny, F., Azizi, S., Shojafar, M., Tafazolli, R.: Joint QoS-aware and cost-efficient task scheduling for fog-cloud resources in a volunteer computing system. ACM Trans. Internet Technol. (TOIT) **21**(4), 1–21 (2021)
19. Hosseinzadeh, M., Abbasi, S., Rahmani, A.M.: Resource management approaches to internet of vehicles. Multimed. Tools Appl. **82**, 1–34 (2023)
20. Abohamama, A.S., El-Ghamry, A., Hamouda, E.: Real-time task scheduling algorithm for IoT-based applications in the cloud–fog environment. J. Netw. Syst. Manag. **30**(4), 54 (2022)
21. Mahmud, R., Ramamohanarao, K., Buyya, R.: Application management in fog computing environments: a taxonomy, review and future directions. ACM Comput. Surv. (CSUR) **53**(4), 1–43 (2020)
22. Barik, R.K., et al.: Mist data: leveraging mist computing for secure and scalable architecture for smart and connected health. Procedia Comput. Sci. **125**, 647–653 (2018)
23. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
24. Tuli, S., Mahmud, R., Tuli, S., Buyya, R.: Fogbus: a blockchain-based lightweight framework for edge and fog computing. J. Syst. Softw. **154**, 22–36 (2019)
25. Chiti, F., Fantacci, R., Picano, B.: A matching game for tasks offloading in integrated edge-fog computing systems. Trans. Emerg. Telecommun. Technol. **31**(2), e3718 (2020)
26. Kocot, B., Czarnul, P., Proficz, J.: Energy-aware scheduling for high-performance computing systems: a survey. Energies (Basel) **16**(2), 890 (2023)
27. Shirvani, H.: A novel discrete grey wolf optimizer for scientific workflow scheduling in heterogeneous cloud computing platforms. Sci. Iranica **29**(5), 2375–2393 (2022)
28. NoorianTalouki, R., Shirvani, M.H., Motameni, H.: A heuristic-based task scheduling algorithm for scientific workflows in heterogeneous cloud computing platforms. J. King Saud Univer.-Comput. Inform. Sci. **34**(8), 4902–4913 (2022)
29. Tanha, M., Hosseini Shirvani, M., Rahmani, A.M.: A hybrid meta-heuristic task scheduling algorithm based on genetic and thermodynamic simulated annealing algorithms in cloud computing environments. Neural Comput. Appl. **33**, 16951–16984 (2021)
30. Mokni, M., Yassa, S., Hajlaoui, J.E., Chelouah, R., Omri, M.N.: Cooperative agents-based approach for workflow scheduling on fog-cloud computing. J. Ambient. Intell. Human. Comput. **13**(10), 4719–4738 (2022)
31. Pies, I., Schreck, P., Homann, K.: Single-objective versus multi-objective theories of the firm: using a constitutional perspective to resolve an old debate. RMS **15**, 779–811 (2021)
32. Kousalya, G., Balakrishnan, P., Pethuru Raj, C., Kousalya, G., Balakrishnan, P., Pethuru Raj, C.: Workflow scheduling algorithms and approaches. In: Smith, J. (ed.) Automated workflow scheduling in self-adaptive clouds: concepts algorithms and methods, pp. 65–83. Springer, Cham (2017)
33. Ismayilov, G., Topcuoglu, H. R.: Dynamic multi-objective workflow scheduling for cloud computing based on evolutionary algorithms. In: 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion), IEEE, pp. 103–108 (2018)
34. Nandhakumar, C., Ranjithprabhu, K.: Heuristic and meta-heuristic workflow scheduling algorithms in multi-cloud environments—A survey. In: 2015 International Conference on Advanced Computing and Communication Systems, IEEE, pp. 1–5 (2015)
35. Topcuoglu, H., Hariri, S., Wu, M.-Y.: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. Parallel Distrib. Syst. **13**(3), 260–274 (2002)
36. Abdalrahman, A.O., Pilevarzadeh, D., Ghafouri, S., Ghaffari, A.: The application of hybrid krill herd artificial hummingbird algorithm for scientific workflow scheduling in fog computing. J. Bionic Eng. **20**, 1–22 (2023)
37. Hajam, S.S., Sofi, S.A.: Spider monkey optimization based resource allocation and scheduling in fog computing environment. High-Conf. Comput. **3**(3), 100149 (2023)
38. Madhura, R., Elizabeth, B.L., Uthariaraj, V.R.: An improved list-based task scheduling algorithm for fog computing environment. Computing **103**, 1353–1389 (2021)
39. Alsaidy, S.A., Abbood, A.D., Sahib, M.A.: Heuristic initialization of PSO task scheduling algorithm in cloud computing.

J. King Saud Univer.-Comput. Inform. Sci. **34**(6), 2370–2382 (2022)
40. Li, F., Tan, W.J., Cai, W.: A wholistic optimization of containerized workflow scheduling and deployment in the cloud–edge environment. Simul. Model. Pract. Theory **118**, 102521 (2022)
41. Bugingo, E., Zheng, W., Lei, Z., Zhang, D., Sebakara, S.R.A., Zhang, D.: Deadline-constrained cost-energy aware workflow scheduling in cloud. Concurr. Comput. **34**(6), e6761 (2022)
42. Khaleel, M.I.: Multi-objective optimization for scientific workflow scheduling based on performance-to-power ratio in fog–cloud environments. Simul. Model. Pract. Theory **119**, 102589 (2022)
43. Hosseini Shirvani, M., Noorian Talouki, R.: Bi-objective scheduling algorithm for scientific workflows on cloud computing platform with makespan and monetary cost minimization approach. Complex Intell. Syst. **8**(2), 1085–1114 (2022)
44. Alsurdeh, R., Calheiros, R.N., Matawie, K.M., Javadi, B.: Hybrid workflow scheduling on edge cloud computing systems. IEEE Access **9**, 134783–134799 (2021)
45. Li, H., Wang, Y., Huang, J., Fan, Y.: Mutation and dynamic objective-based farmland fertility algorithm for workflow scheduling in the cloud. J. Parallel Distrib. Comput. **164**, 69–82 (2022)
46. Mollajafari, M., Shojaeefard, M.H.: TC3PoP: a time-cost compromised workflow scheduling heuristic customized for cloud environments. Clust. Comput. **24**(3), 2639–2656 (2021)
47. Arora, N., Banyal, R.K.: Workflow scheduling using particle swarm optimization and gray wolf optimization algorithm in cloud computing. Concurr. Comput. **33**(16), e6281 (2021)
48. Wu, C., Li, W., Wang, L., Zomaya, A.Y.: Hybrid evolutionary scheduling for energy-efficient fog-enhanced internet of things. IEEE Trans. Cloud Comput. **9**(2), 641–653 (2018)
49. Abualigah, L., Diabat, A., Elaziz, M.A.: Intelligent workflow scheduling for big data applications in IoT cloud computing environments. Clust. Comput. **24**(4), 2957–2976 (2021)
50. Mohammadzadeh, A., Akbari Zarkesh, M., Haji Shahmohamd, P., Akhavan, J., Chhabra, A.: Energy-aware workflow scheduling in fog computing using a hybrid chaotic algorithm. J. Supercomput. **79**, 1–36 (2023)
51. Singh, G., Chaturvedi, A.K.: Hybrid modified particle swarm optimization with genetic algorithm (GA) based workflow scheduling in cloud-fog environment for multi-objective optimization. Clust. Comput. **27**, 1–18 (2023)
52. Khaleel, M.I.: Hybrid cloud-fog computing workflow application placement: joint consideration of reliability and time credibility. Multimed. Tools Appl. **82**(12), 18185–18216 (2023)
53. Iftikhar, S., et al.: HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments. Internet Things **21**, 100667 (2023)
54. Konjaang, J.K., Xu, L.: Meta-heuristic approaches for effective scheduling in infrastructure as a service cloud: a systematic review. J. Netw. Syst. Manag. **29**, 1–57 (2021)
55. Bacanin, N., Zivkovic, M., Bezdan, T., Venkatachalam, K., Abouhawwash, M.: Modified firefly algorithm for workflow scheduling in cloud-edge environment. Neural Comput. Appl. **34**(11), 9043–9068 (2022)
56. Asghari Alaie, Y., Hosseini Shirvani, M., Rahmani, A.M.: A hybrid bi-objective scheduling algorithm for execution of scientific workflows on cloud platforms with execution time and reliability approach. J Supercomput **79**(2), 1451–1503 (2023)
57. Hafsi, H., Gharsellaoui, H., Bouamama, S.: Genetically-modified multi-objective particle swarm optimization approach for high-performance computing workflow scheduling. Appl. Soft Comput. **122**, 108791 (2022)
58. Xie, Y., Sheng, Y., Qiu, M., Gui, F.: An adaptive decoding biased random key genetic algorithm for cloud workflow scheduling. Eng. Appl. Artif. Intell. **112**, 104879 (2022)
59. Mansour, R.F., Alhumyani, H., Khalek, S.A., Saeed, R.A., Gupta, D.: Design of cultural emperor penguin optimizer for energy-efficient resource scheduling in green cloud computing environment. Clust. Comput. **26**(1), 575–586 (2023)
60. Khaledian, N., Khamforoosh, K., Azizi, S., Maihami, V.: IKH-EFT: an improved method of workflow scheduling using the krill herd algorithm in the fog-cloud environment. Sustain. Comput.: Inform. Syst. **37**, 100834 (2023)
61. Kamanga, C.T., Bugingo, E., Badibanga, S.N., Mukendi, E.M.: A multi-criteria decision making heuristic for workflow scheduling in cloud computing environment. J. Supercomput. **79**(1), 243–264 (2023)
62. Rani, R., Garg, R.: Pareto based ant lion optimizer for energy efficient scheduling in cloud environment. Appl. Soft Comput. **113**, 107943 (2021)
63. Hussain, M., Wei, L.-F., Rehman, A., Abbas, F., Hussain, A., Ali, M.: Deadline-constrained energy-aware workflow scheduling in geographically distributed cloud data centers. Future Gener. Comput. Syst. **132**, 211–222 (2022)
64. Mutlag, A.A., et al.: A new fog computing resource management (FRM) model based on hybrid load balancing and scheduling for critical healthcare applications. Phys. Commun. **59**, 102109 (2023)
65. Javaheri, D., Gorgin, S., Lee, J.-A., Masdari, M.: An improved discrete Harris hawk optimization algorithm for efficient workflow scheduling in multi-fog computing. Sustain. Comput.: Inform. Syst. **36**, 100787 (2022)
66. Qiu, H., Xia, X., Li, Y., Deng, X.: A dynamic multipopulation genetic algorithm for multiobjective workflow scheduling based on the longest common sequence. Swarm Evol. Comput. **78**, 101291 (2023)
67. Wang, Y., Zuo, X.: An effective cloud workflow scheduling approach combining PSO and idle time slot-aware rules. IEEE/CAA J. Automatica Sin. **8**(5), 1079–1094 (2021)
68. Li, H., Wang, D., Xu, G., Yuan, Y., Xia, Y.: Improved swarm search algorithm for scheduling budget-constrained workflows in the cloud. Soft Comput. **26**(8), 3809–3824 (2022)
69. Li, H., Wang, D., Canizares Abreu, J.R., Zhao, Q., Bonilla Pineda, O.: PSO+ LOA: hybrid constrained optimization for scheduling scientific workflows in the cloud. J. Supercomput. **77**, 13139–13165 (2021)
70. Shirvani, M.H.: A hybrid meta-heuristic algorithm for scientific workflow scheduling in heterogeneous distributed computing systems. Eng. Appl. Artif. Intell. **90**, 103501 (2020)
71. Javanmardi, S., Shojafar, M., Mohammadi, R., Persico, V., Pescapè, A.: S-FoS: a secure workflow scheduling approach for performance optimization in SDN-based IoT-Fog networks. J. Inform. Secur. Appl. **72**, 103404 (2023)
72. Valappil Thekkepuryil, J.K., Suseelan, D.P., Keerikkattil, P.M.: An effective meta-heuristic based multi-objective hybrid optimization method for workflow scheduling in cloud computing environment. Clust. Comput. **24**, 2367–2384 (2021)
73. Wang, Z., Goudarzi, M., Gong, M., Buyya, R.: Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. Future Gener. Comput. Syst. **152**, 55–69 (2024)
74. Kaur, A., Singh, P., Singh Batth, R., Peng Lim, C.: Deep-Q learning-based heterogeneous earliest finish time scheduling algorithm for scientific workflows in cloud. Softw. Pract. Exp. **52**(3), 689–709 (2022)
75. Saif, F.A., Latip, R., Hanapi, Z.M., Shafinah, K.: Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing. IEEE Access **11**, 20635–20646 (2023)

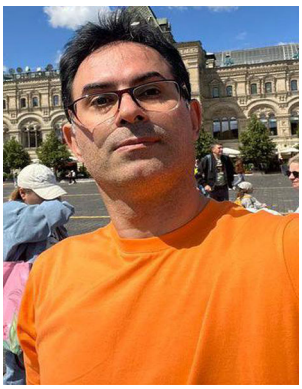
76. Li, H., Huang, J., Wang, B., Fan, Y.: Weighted double deep Q-network based reinforcement learning for bi-objective multi-workflow scheduling in the cloud. Clust. Comput. **25**, 1–18 (2022)
77. Chen, G., Qi, J., Sun, Y., Hu, X., Dong, Z., Sun, Y.: A collaborative scheduling method for cloud computing heterogeneous workflows based on deep reinforcement learning. Future Gener. Comput. Syst. **141**, 284–297 (2023)
78. "Schedule Optimization Approaches and Use Cases." Accessed: Feb. 23, 2024. [Online]. Available: https://www.altexsoft.com/blog/schedule-optimization/
79. Ziagham Ahwazi A.: Budget-aware scheduling algorithm for scientific workflow applications across multiple clouds. A Mathematical Optimization-Based Approach. May 2022, Accessed: Feb. 23, 2024. [Online]. Available: https://munin.uit.no/handle/10037/25932
80. Chakravarthi, K.K., Neelakantan, P., Shyamala, L., Vaidehi, V.: Reliable budget aware workflow scheduling strategy on multi-cloud environment. Clust. Comput. **25**(2), 1189–1205 (2022)
81. Xie, Y., Gui, F.-X., Wang, W.-J., Chien, C.-F.: A two-stage multi-population genetic algorithm with heuristics for workflow scheduling in heterogeneous distributed computing environments. IEEE Trans. Cloud Comput. **11**, 1446 (2021)
82. Xu, M., et al.: Genetic programming for dynamic workflow scheduling in fog computing. IEEE Trans. Serv. Comput. **16**, 267 (2023)
83. Davami, F., Adabi, S., Rezaee, A., Rahmani, A.M.: Distributed scheduling method for multiple workflows with parallelism prediction and DAG prioritizing for time constrained cloud applications. Comput. Netw. **201**, 108560 (2021)
84. Karami, S., Azizi, S., Ahmadizar, F.: A bi-objective workflow scheduling in virtualized fog-cloud computing using NSGA-II with semi-greedy initialization. Appl. Soft Comput. **151**, 111142 (2024)
85. Mikram, H., El Kafhali, S., Saadi, Y.: HEPGA: a new effective hybrid algorithm for scientific workflow scheduling in cloud computing environment. Simul. Model. Pract. Theory **130**, 102864 (2024)
86. Rathi, S., Nagpal, R., Srivastava, G., Mehrotra, D.: A multi-objective fitness dependent optimizer for workflow scheduling. Appl. Soft Comput. **152**, 111247 (2024)
87. Gu, Y., Cheng, F., Yang, L., Xu, J., Chen, X., Cheng, L.: Cost-aware cloud workflow scheduling using DRL and simulated annealing. Digital Commun. Netw. (2024). https://doi.org/10.1016/j.dcan.2023.12.009
88. Ye, L., Yang, L., Xia, Y., Zhao, X.: A cost-driven intelligence scheduling approach for deadline-constrained IoT workflow applications in cloud computing. IEEE Internet Things J. (2024). https://doi.org/10.1109/JIOT.2024.3351630
89. Mangalampalli, S., et al.: Multi objective prioritized workflow scheduling using deep reinforcement based learning in cloud computing. IEEE Access **12**, 5373 (2024)
90. Xie, H., Ding, D., Zhao, L., Kang, K., Liu, Q.: A two-stage preference driven multi-objective evolutionary algorithm for workflow scheduling in the Cloud. Expert Syst. Appl. **238**, 122009 (2024)
91. Lu, C., Zhu, J., Huang, H., Sun, Y.: A multi-hierarchy particle swarm optimization-based algorithm for cloud workflow scheduling. Future Gener. Comput. Syst. **153**, 125–138 (2024)
92. Mokni, M., Yassa, S., Hajlaoui, J.E., Omri, M.N., Chelouah, R.: Multi-objective fuzzy approach to scheduling and offloading workflow tasks in fog-cloud computing. Simul. Model. Pract. Theory **123**, 102687 (2023)
93. Mohammadzadeh, A., Masdari, M.: Scientific workflow scheduling in multi-cloud computing using a hybrid multi-objective optimization algorithm. J. Ambient. Intell. Human. Comput. **14**(4), 3509–3529 (2023)
94. Shukla, P., Pandey, S.: DE-GWO: a multi-objective workflow scheduling algorithm for heterogeneous fog-cloud environment. Arab. J. Sci. Eng. **14**, 1–26 (2023)
95. Ijaz, S., Munir, E.U., Ahmad, S.G., Rafique, M.M., Rana, O.F.: Energy-makespan optimization of workflow scheduling in fog–cloud computing. Computing **103**, 2033–2059 (2021)
96. Subramoney, D., Nyirenda, C.N.: Multi-swarm PSO algorithm for static workflow scheduling in cloud-fog environments. IEEE Access **10**, 117199–117214 (2022)
97. Ma, X., Xu, H., Gao, H., Bian, M.: Real-time multiple-workflow scheduling in cloud environments. IEEE Trans. Netw. Serv. Manag. **18**(4), 4002–4018 (2021)
98. Belgacem, A., Beghdad-Bey, K.: Multi-objective workflow scheduling in cloud computing: trade-off between makespan and cost. Clust. Comput. **25**(1), 579–595 (2022)
99. Aziza, H., Krichen, S.: A hybrid genetic algorithm for scientific workflow scheduling in cloud environment. Neural Comput. Appl. **32**, 15263–15278 (2020)
100. Hu, Y., Wang, H., Ma, W.: Intelligent cloud workflow management and scheduling method for big data applications. J. Cloud Comput. **9**, 1–13 (2020)
101. Dong, T., Xue, F., Xiao, C., Zhang, J.: Workflow scheduling based on deep reinforcement learning in the cloud environment. J. Ambient Intell. Human. Comput. **12**, 1–13 (2021)
102. Saeedi, S., Khorsand, R., Bidgoli, S.G., Ramezanpour, M.: Improved many-objective particle swarm optimization algorithm for scientific workflow scheduling in cloud computing. Comput. Ind. Eng. **147**, 106649 (2020)
103. Choudhary, A., Govil, M.C., Singh, G., Awasthi, L.K., Pilli, E.S.: Energy-aware scientific workflow scheduling in cloud environment. Clust. Comput. **25**(6), 3845–3874 (2022)
104. Mohammadzadeh, A., Masdari, M., Gharehchopogh, F.S.: Energy and cost-aware workflow scheduling in cloud computing data centers using a multi-objective optimization algorithm. J. Netw. Syst. Manag. **29**, 1–34 (2021)
105. Iranmanesh, A., Naji, H.R.: DCHG-TS: a deadline-constrained and cost-effective hybrid genetic algorithm for scientific workflow scheduling in cloud computing. Clust. Comput. **24**, 667–681 (2021)
106. Lakhwani, K., et al.: Adaptive and convex optimization-inspired workflow scheduling for cloud environment. Int. J. Cloud Appl. Comput. (IJCAC) **13**(1), 1–25 (2023)
107. Mohammadzadeh, A., Masdari, M., Gharehchopogh, F.S., Jafarian, A.: Improved chaotic binary grey wolf optimization algorithm for workflow scheduling in green cloud computing. Evol. Intell. **14**, 1997–2025 (2021)
108. Gu, Y., Budati, C.: Energy-aware workflow scheduling and optimization in clouds using bat algorithm. Future Gener. Comput. Syst. **113**, 106–112 (2020)
109. Sharma, G., Khurana, S., Harnal, S., Lone, S.A.: CSFPA: an intelligent hybrid workflow scheduling algorithm based upon global and local optimization approach in cloud. Concurr. Comput. **34**(23), e7176 (2022)
110. Calzarossa, M.C., Della Vedova, M.L., Massari, L., Nebbione, G., Tessera, D.: Multi-objective optimization of deadline and budget-aware workflow scheduling in uncertain clouds. IEEE Access **9**, 89891–89905 (2021)
111. Marwa, M., Hajlaoui, J.E., Sonia, Y., Omri, M.N., Rachid, C.: Multi-agent system-based fuzzy constraints offer negotiation of workflow scheduling in fog-cloud environment. Computing **105**(7), 1361–1393 (2023)
112. Akraminejad, R., Khaledian, N., Nazari, A., Voelp, M.: A multi-objective crow search algorithm for optimizing makespan and

costs in scientific cloud workflows (CSAMOMC). Computing **2024**, 1–17 (2024). https://doi.org/10.1007/S00607-024-01263-4
113. Khaledian, N., Khamforoosh, K., Akraminejad, R., Abualigah, L., Javaheri, D.: An energy-efficient and deadline-aware workflow scheduling algorithm in the fog and cloud environment. Computing **106**(1), 109–137 (2024)
114. Srikanth, G.U., Geetha, R.: Effectiveness review of the machine learning algorithms for scheduling in cloud environment. Arch. Comput. Methods Eng. **30**, 1–21 (2023)

Navid Khaledian received his Ph.D. in Artificial Intelligence from the Islamic Azad University, Sanandaj branch, Iran, in 2023. He is currently a postdoctoral researcher at the University of Luxembourg. His research interests include Artificial Intelligence (AI), Distributed Systems, and Internet of Things (IoT), focusing on task scheduling, recommender systems, and data mining.

Marcus Voelp is head of the CritiX lab of the Interdisciplinary Centre for Security, Reliability and Trust (SnT) at the University of Luxembourg. He received his Ph.D. in computer science in 2011 from Technische Universität Dresden. His research interests include methods, tools and system architectures for constructing resilient cyber-physical and embedded systems for a wide variety of application domains and at various levels of the hardware-software stack, from Industry 4.0 over autonomous vehicles to space and from microkernels and microkernel-based systems that are able to simultaneously fulfill a large range of functional and non-functional properties for the applications they run to distributed algorithms coordinating nearby and far away vehicles. Guarantees must be given concerning real-time, security (including information-flow) and dependability, in particular in terms of Byzantine fault and intrusion tolerance and resilience of unattended systems.

Sadoon Azizi is an Associate Professor with the Department of Computer Engineering and IT, University of Kurdistan, Sanandaj, Iran. He is also the director of the Distributed Computing Systems Research Laboratory (DCS Lab) and Head of the High-Performance Computing (HPC) center at the University of Kurdistan. His research interests include Cloud, Fog, Edge, and Serverless Computing, Internet of Things, and Artificial Intelligence (AI) and Machine Learning (ML) for Distributed Systems. For additional information: https://research.uok.ac.ir/~sazizi/en.



Mirsaeid Hosseini Shirvani received his B.Sc., M.Sc., and Ph.D. all in Computer Software Engineering Systems at Universities in Tehran, IRAN. He has been teaching miscellaneous computer courses in several universities in Mazandaran province of IRAN since 2001. He was the former head of computer engineering department at IAU (Sari-Branch) during 2010–2012. Currently, he serves as Professor in Computer Engineering Department at IAU (Sari-Branch). He also published several papers in authentic and world-reputed journals. Moreover, he serves as reviewer of numerous journals in IEEE, Elsevier, Wiley, Springer, Taylor & Francis, Emerald, etc. publications. His-research interests are in the areas of cloud- and fog computing, IoT, distributed systems, parallel processing, machine learning, and evolutionary computations. Google Scholar: https://scholar.google.com/citations?user=Hz8PfNQAAAAJ&hl=en.