Sustainable Computing: Informatics and Systems 45 (2025) 101080

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

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

Optimizing task offloading with metaheuristic algorithms across cloud, fog, and edge computing networks: A comprehensive survey and state-of-the-art schemes

Amir Masoud Rahmani [a,1], Amir Haider [b,1], Parisa Khoshvaght [c,d], Farhad Soleimanian Gharehchopogh [e,f], Komeil Moghaddasi [e], Shakiba Rajabi [e], Mehdi Hosseinzadeh [g,h,*]

[a] *Future Technology Research Center, National Yunlin University of Science and Technology, Yunlin, Taiwan*
[b] *Department of Artificial Intelligence and Robotics, Sejong University, Seoul 05006, Republic of Korea*
[c] *Institute of Research and Development, Duy Tan University, Da Nang, Vietnam*
[d] *School of Engineering & Technology, Duy Tan University, Da Nang, Vietnam*
[e] *Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran*
[f] *Applied Science Research Center, Applied Science Private University, Amman, Jordan*
[g] *School of Computer Science, Duy Tan University, Da Nang, Vietnam*
[h] *Jadara University Research Center, Jadara University, Irbid, Jordan*

A R T I C L E I N F O

Keywords:
Offloading
Metaheuristic algorithms
Cloud computing
Fog computing
Edge computing

A B S T R A C T

The Internet of Things (IoT) significantly impacts various industries, enabling better connectivity and real-time data exchange for applications ranging from smart cities to healthcare. Integrating cloud, fog, and edge computing is essential for managing increased data and processing needs as IoT networks become complex. Cloud computing provides extensive storage and powerful computing capabilities but can experience delays due to the distance data must travel. Fog computing addresses these delays by processing data closer to its source, while edge computing reduces them even further by processing data directly on IoT devices. Effective management of these computing layers requires strategic task offloading, which involves moving tasks to the most appropriate computing layer to balance latency, energy consumption, and operational efficiency. Several strategies have been developed to optimize network communication and task offloading, with metaheuristic algorithms emerging as promising approaches. Inspired by natural processes, these algorithms are skilled at searching complex spaces to find near-optimal solutions for efficient and dynamic task offloading. This review provides a detailed analysis of how metaheuristic algorithms optimize task offloading. It evaluates their effectiveness in improving system performance, managing resources, and reducing costs. The review also identifies the current challenges in this area and suggests future research directions to advance this field.

1. Introduction

The IoT is transforming many industries by improving how they connect and share data [1]. IoT is used in various areas like the Internet of Vehicles (IoV), Industrial Internet of Things (IIoT), healthcare, smart farming, smart cities, home automation, and environmental monitoring [2]. These sectors use IoT to make their operations more efficient, help with decision-making, and manage their operations in real time [3]. However, as IoT networks expand, they face significant data volume, processing needs, and latency challenges, necessitating advanced computing solutions [4]. To address these challenges, integration with cloud computing offers vast storage capacities and powerful processing capabilities, enabling extensive data analysis and management [5]. Meanwhile, fog computing brings cloud computing capabilities closer to the IoT devices at the network edge, facilitating faster response times and improved connectivity in environments with limited bandwidth [6].

* Corresponding author at: School of Computer Science, Duy Tan University, Da Nang, Vietnam.
E-mail addresses: rahmania@yuntech.edu.tw (A.M. Rahmani), Amirhaider@sejong.ac.kr (A. Haider), parisakhoshvaght@duytan.edu.vn (P. Khoshvaght), k.moghaddasi@ieee.org (K. Moghaddasi), Shakiba.rajabi@ieee.org (S. Rajabi), mehdihosseinzadeh@duytan.edu.vn (M. Hosseinzadeh).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.suscom.2024.101080
Received 31 July 2024; Received in revised form 24 November 2024; Accepted 30 December 2024
Available online 3 January 2025

Moreover, edge computing and Mobile Edge Computing (MEC) push data processing directly onto devices where data is generated, minimizing latency even more and allowing for real-time data processing, which is crucial for applications requiring immediate responses, such as autonomous vehicles and industrial automation systems [7,8]. Each computing layer contributes uniquely, ensuring IoT systems are more responsive, efficient, and tailored to specific industry needs.

Integrating IoT with cloud, fog, and edge computing introduces its own set of challenges, including issues with energy consumption, latency, and operational costs [9]. Cloud computing, while powerful, often results in higher latency due to the physical distance data must travel, which can be problematic for time-sensitive applications [10]. Additionally, the cost of data transmission and the energy required to operate cloud data centers are significant [11]. Conversely, fog computing reduces latency by processing data closer to the source but still faces challenges managing the energy use of numerous local nodes scattered across the network [12]. Edge computing offers the lowest latency by processing data directly on IoT devices. Yet, it may involve increased costs and energy demands due to the need for more sophisticated hardware at each node [13]. Offloading strategies are employed to optimize these computing paradigms [14]. Offloading involves transferring tasks to the most appropriate computing layer, whether cloud, fog, or edge, based on processing needs, energy efficiency, and cost-effectiveness, enhancing overall system performance and sustainability [15]. The research community has developed many methods to optimize offloading in IoT systems, including several machine-learning techniques. One such method treats task offloading as a parallel-machine scheduling issue, which helps distribute computing loads efficiently across edge servers. This approach is particularly beneficial for mobile crowdsensing services, split into opportunistic and participatory types. Efficient offloading improves data processing at the edge, reducing latency and server load and boosting system responsiveness. mobile crowd sensing, for example, uses mobility prediction to allocate tasks intelligently, keeping workers on their usual routes while minimizing data congestion and processing demands on central servers. Similarly, UAV-assisted data collection in areas affected by disasters also employs offloading strategies. Offloading computational tasks to edge nodes allows UAVs to reduce their computational burden, extend their operational time, and concentrate on gathering critical data [16–19]. Particularly noteworthy is the application of metaheuristic algorithms, which have significantly improved offloading efficiencies. The novelty of our study lies in its thorough and systematic examination of metaheuristic algorithms for task offloading problems across cloud, fog, and edge computing networks. This subject has not been explored in existing literature. Our research critically evaluates various metaheuristic strategies and analyzes their application and effectiveness in diverse settings

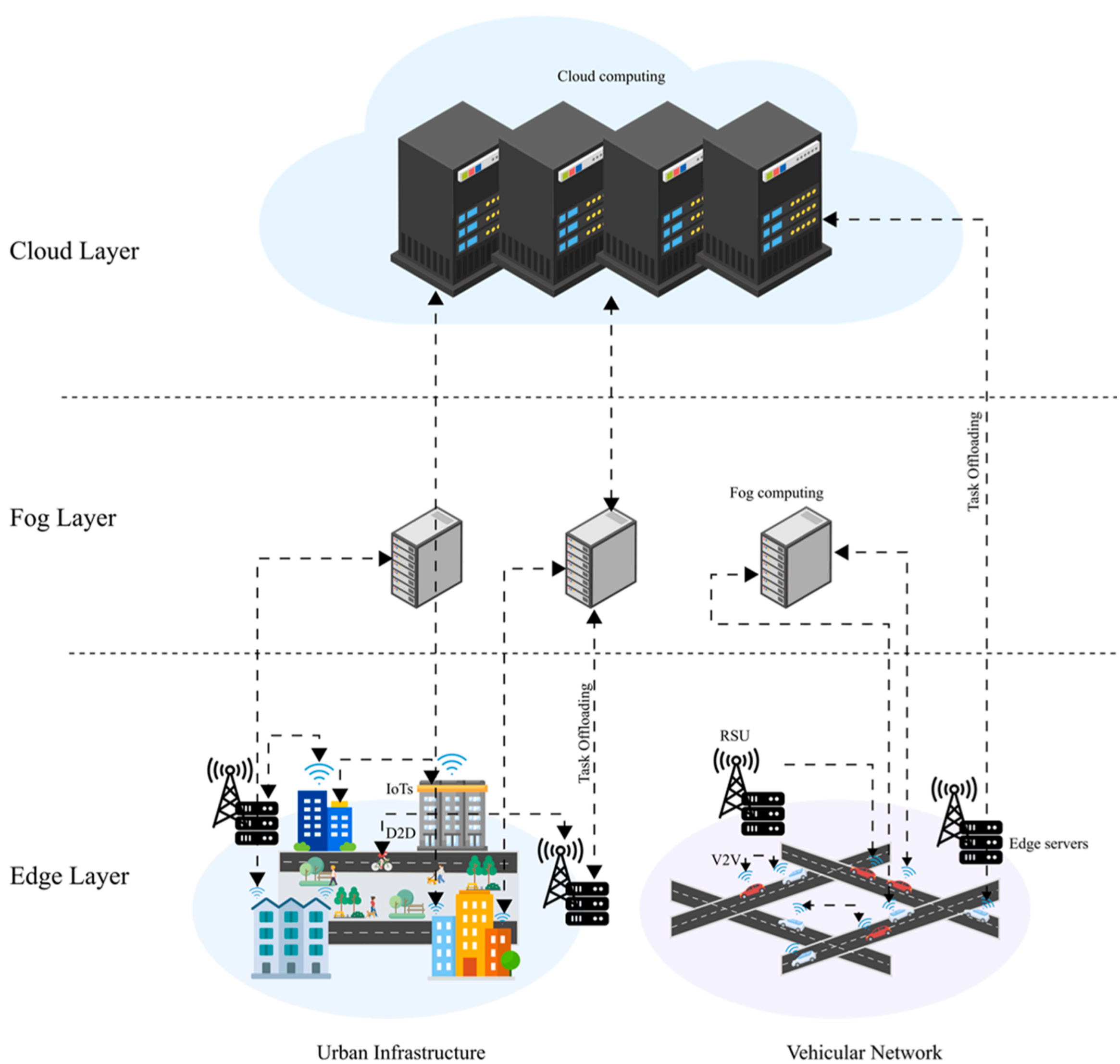

Fig. 1. Edge/Fog/Cloud offloading architecture in IoT and Vehicular networks.

[20]. Our review highlights the strengths and limitations of these methods and explores their scalability and implementation challenges. This provides valuable insights into each method's practical utility and effectiveness, bridging a significant gap in current research. Additionally, our work proposes directions for future research that could improve the efficiency, scalability, and adaptability of offloading processes—essential for advancing distributed computing technologies. We also perform a cross-domain assessment, considering key aspects like security, energy efficiency, cost-effectiveness, and adaptability. This enhances our understanding of optimizing these algorithms to meet the needs of next-generation networks. Thus, our study not only fills a significant research gap by examining offloading through the lens of metaheuristics but also helps future inquiries, marking a significant advancement in the study of computational offloading strategies. Fig. 1 depicts the hierarchical structure of computing layers in urban and vehicular networks, showcasing data flow between edge, fog, and cloud layers for task offloading.

Although DRL algorithms are widely used for optimizing offloading in edge computing, as cited as [21–25], metaheuristics provide versatile solutions for addressing complex optimization challenges in dynamic environments. Metaheuristic algorithms are advanced computational techniques designed to solve complex optimization problems that are otherwise too challenging for traditional methods. These algorithms are inspired by natural phenomena and behaviors [26]. By exploring a broad search space, metaheuristics efficiently find near-optimal solutions to intricate issues, making them ideal for dynamically optimizing where and how tasks are offloaded in cloud, fog, and edge computing environments [27]. Their flexibility and robustness allow for effective adaptation to changing network conditions and varying system demands, enhancing performance and resource utilization. Table 1 includes the existing abbreviations in this study.

1.1. Scope of the study

This study evaluates and compares various metaheuristic algorithms to optimize task offloading in IoT environments incorporating cloud, fog, and edge computing. As more IoT devices come into use, they put more pressure on network resources, making it vital to process data efficiently. Metaheuristic algorithms are promising because they help manage resources better and improve how tasks are offloaded between cloud, fog, and edge computing. This study explores these algorithms further, hoping to find new ways to make IoT systems more sustainable and perform better. By doing this, we could help shape future technology and how it is implemented in the fast-changing world of IoT.

Table 1
Applied abbreviations.

Abbreviation	Description
IoT	Internet of Things
IoV	Internet of Vehicles
IIoT	Industrial Internet of Things
GA	Genetic Algorithm
PSO	Particle Swarm Optimization
ACO	Ant Colony Optimization
SA	Simulated Annealing
CS	Cuckoo Search
GWO	Grey Wolf Optimizer
WOA	Whale Optimization Algorithm
BA	Bee Algorithm
HHO	Harris Hawks Optimization
MFO	Moth-Flame Optimization
EGA	Evolutionary Genetic Algorithm
GOA	Genetic Optimization Algorithm
MDs	Mobile Devices
VANET	Vehicular Ad Hoc Network
UAV	Uncrewed Aerial Vehicle
QoS	Quality of Service
DRL	Deep Reinforcement Learning
MEC	Mobile Edge Computing
LEO	Low Earth Orbit
SMDs	Smart Mobile Devices
Deep Q Network	DQN
User Equipment	UE

1.2. Motivation of the study

This study is motivated by the urgent need to handle increasing data demands in IoT systems effectively. To our knowledge, no comprehensive review study focuses explicitly on optimizing task offloading with metaheuristic algorithms. This study aims to fill this gap by systematically analyzing the effectiveness of various metaheuristic strategies in this field. We explored how these algorithms enhance system performance by reducing latency, decreasing energy consumption, and managing the costs associated with data processing. We covered a detailed review of the existing literature, identifying strengths and limitations of current approaches and suggesting directions for future research to improve offloading efficiencies in distributed computing contexts.

1.3. Contributions of the study

This review comprehensively examines the landscape and inherent challenges of offloading strategies employing metaheuristic algorithms. The key contributions of this study are summarized below:

- It identifies metaheuristic algorithms that are particularly effective at improving scalability and adaptability for task offloading across fog, edge, and cloud computing environments.
- The study comprehensively analyzes the complexities and security risks of using metaheuristic algorithms in task offloading.
- It evaluates which metaheuristic algorithms offer the best cost-efficiency and energy savings in various offloading scenarios. Additionally, it examines how different metaheuristic algorithms help reduce latency and improve service quality in task offloading.
- This research explores the practical feasibility of implementing metaheuristic algorithms across different technological settings, providing insight into the challenges and potential solutions.
- It highlights this field's challenges and limitations and suggests future research directions for using metaheuristic algorithms to optimize offloading tasks.

The structure of this paper is organized as follows: Section 2 outlines the methods used for collecting and analyzing important literature related to offloading strategies that utilize metaheuristic algorithms. Section 3 presents details on the essential characteristics of the studies reviewed. Section 4 critically examines and discusses cutting-edge research, emphasizing recent innovations and techniques in the field. Section 5 addresses the research questions by analyzing the gathered studies. Section 6 explores the significant challenges that currently hinder advancements in these systems. The paper concludes with Section 7, summarizing the main findings, acknowledging the limitations of current methods, and suggesting future research opportunities that aim to overcome these obstacles and enhance system performance.

2. Methodology

The method used for this systematic review is based on the methodology developed by Kitchenham and Charters [28], which has three main phases: Planning, Conducting, and Reporting. Each phase comprises several detailed activities, as shown in Fig. 1. During the planning phase, a critical step involves defining the research questions that set the review's objectives and establishing a review protocol. The protocol is organized into six crucial stages: (1) identifying the research questions, (2) formulating the search strategy, (3) defining the selection criteria, (4) devising methods for quality assessment, (5) outlining techniques for data extraction, and (6) selecting methods for data synthesis. These stages are further detailed in Fig. 2. Additional information on each component of the review protocol will be discussed in the subsequent subsections.

2.1. Research questions

In our study, we initially set out the research questions this review article aims to address. These questions are the foundation of our investigation, directing the choice of literature and informing the analysis. The questions we intend to explore include:

- RQ1: Which metaheuristic algorithms enhance scalability and adaptability in offloading tasks across fog, edge, and cloud computing environments?
- RQ2: What complexities and security risks have been reported using metaheuristic algorithms in computational offloading implementations?

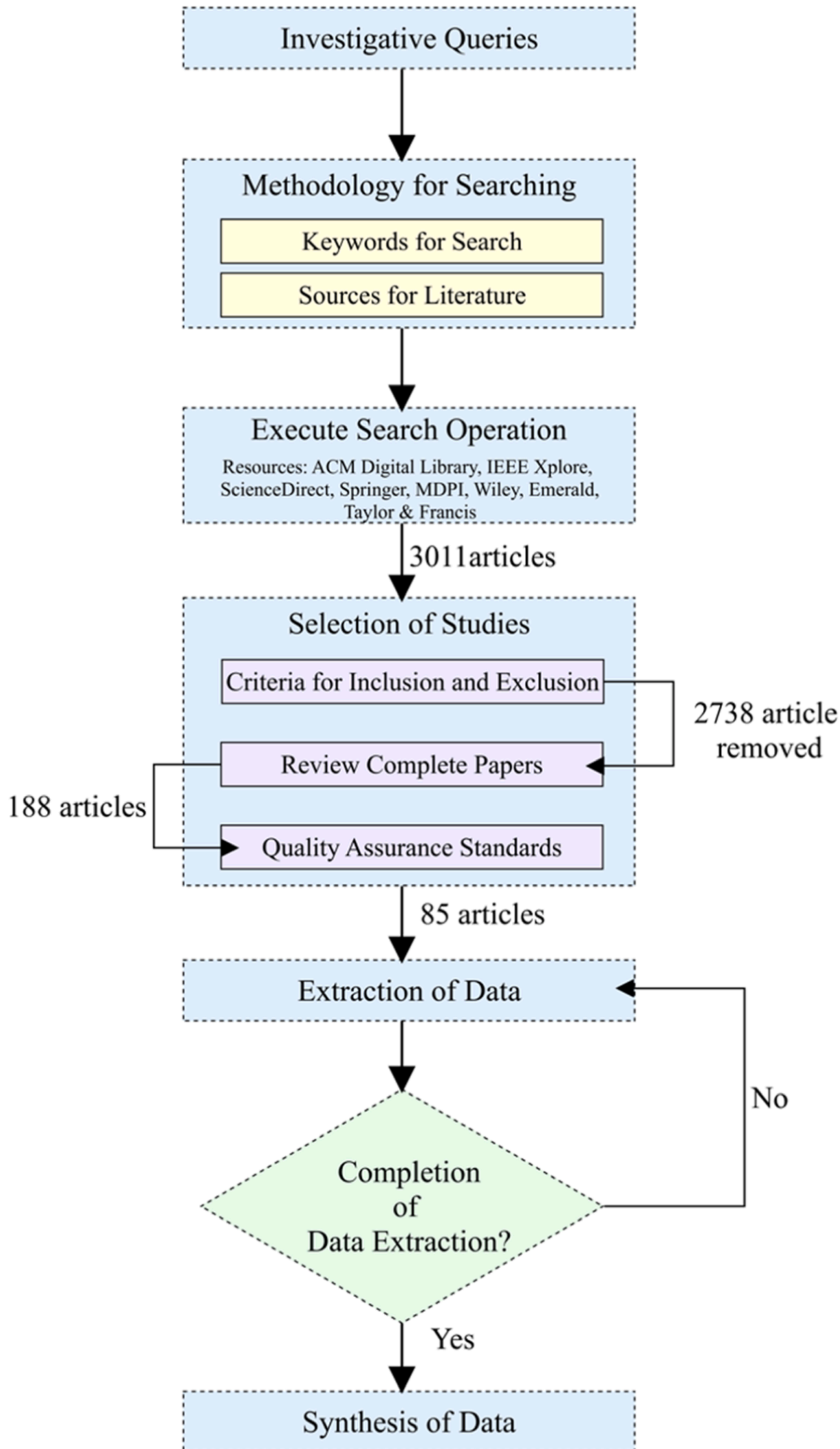

Fig. 2. Review process.

- RQ3: Which metaheuristic algorithms are recognized as the most cost-effective and energy-efficient for offloading across various environments?
- RQ4: Which metaheuristic algorithms contribute to improved latency and quality of service in offloading tasks for different systems?
- RQ5: How feasible is the implementation of metaheuristic algorithms for computational offloading in diverse technological environments?

2.2. Review process

A systematic review aims to gather many primary studies that respond to defined research questions using a fair and unbiased search strategy. This process includes creating search terms and choosing databases to source applicable articles. In the study selection phase, articles are screened and reduced based on specific inclusion and exclusion criteria. This step is succeeded by an in-depth review of the complete texts and the implementation of quality assessment measures to narrow down the choices further. Finally, the relevant articles are thoroughly analyzed and synthesized to deliver detailed responses to the research questions and to summarize the essential findings. To achieve this objective, we carried out a literature search on Google Scholar to find review papers and survey articles that focus on offloading strategies using metaheuristic algorithms. We refined our search using the following keywords:

- Offloading strategies AND Metaheuristic algorithms Review
- Offloading strategies AND Metaheuristic algorithms Survey
- Offloading AND Metaheuristic Review
- Offloading AND Metaheuristic Survey

These keywords included the names of various metaheuristic algorithms, terms related to bio-nature-inspired algorithms, and others relevant to the study's focus. We did not find relevant articles from this search, and even with thorough analysis, we did not find clear answers to our research questions. This shows a gap in the existing research, which motivates us to conduct a new study in this area.

2.3. Literature resources

Eight digital databases were employed to collect the initial set of articles: IEEE Xplore, ScienceDirect, ACM Digital Library, Springer, MDPI, Wiley, Emerald, and Taylor and Francis. The search spanned from January 2018 to April 5, 2024. Using specified search terms across these platforms, the initial search yielded 5011 articles, including their titles, abstracts, and keywords. Table 2 summarizes the number of studies extracted from each database following the initial search.

To handle the large amount of work involved in reviewing articles, we used the systematic review software Rayyan and Covidence. This software helped us screen titles and abstracts, remove duplicates, and review full texts. It improved our efficiency in assessing the relevance and quality of each study based on our set criteria, ensuring our analysis of the literature was thorough and organized. As shown in Fig. 2, the procedure for choosing primary studies consisted of three straightforward steps: (1) removing any duplicate studies, (2) applying specific inclusion and exclusion criteria to select articles relevant to the research questions, and discarding those that are not, and (3) carrying out a quality assessment to make sure only high-quality studies were considered. The criteria for including and excluding studies are outlined in Table 3.

We thoroughly scanned each selected study during the review process to ensure all research aspects were relevant and thoroughly examined. This scrutiny was essential to evaluate the relevance and completeness of the content. After this, we applied quality assessment criteria to each study to check its reliability and scientific merit, ensuring that only high-quality research was considered. Following this evaluation, we extracted data, gathering essential information from each paper in a standardized format. This step was crucial for ensuring accurate comparison and thorough analysis across studies. Finally, we synthesized the extracted data to develop a cohesive understanding of the collective findings. This synthesis was key in addressing the research questions and drawing well-founded conclusions based on the aggregated evidence from the reviewed studies (Tables 4–12).

Fig. 3 shows a breakdown of the number of publications and conferences produced by different publishers. IEEE stands out with 30 publications and eight meetings, highlighting its dominant role in this area of research. Following IEEE, Springer has published 25 works, and Elsevier has 20. On the other hand, MDPI and Wiley have less output, with Wiley at the bottom with only a single publication. This illustration clearly shows how these publishers compare in their contributions to the field.

Fig. 4 illustrates the progression in the number of publications from 2018 through 2024, highlighting a marked rise in research output starting from 2019. The data peaks in 2023 with 28 publications, signifying a robust interest and substantial advancements within the field. Although the count for 2024 currently stands at 15 publications, this number illustrates the continued active research and contributions to the domain.

Fig. 5 presents the distribution of publications by country, with China leading significantly with 40 publications, underscoring its dominant role in the field. China, Egypt, and India contributed substantially, with 6 and 10 publications. Countries like Iran and Saudi Arabia have contributed 5 and 6 publications, respectively. Notably, several countries, such as Brazil, Indonesia, Jordan, and others, maintain a minimal presence with only one publication each. This distribution highlights the varying levels of engagement and research output across different countries in this academic area.

3. Research background

This section will briefly discuss metaheuristic algorithms, system

Table 2
Number of the studies in each database.

IEEE Xplore	805
ScienceDirect	731
ACM Digital Library	449
Springer	400
MDPI	388
Wiley	83
Emerald	42
Taylor and Francis	113

Table 3
Inclusion and exclusion criteria.

Inclusion Criteria	Exclusion Criteria
The studies had to address the research questions that were formulated directly.	We excluded grey literature, abstracts, editorials, and non-peer-reviewed reports.
We only considered peer-reviewed articles and conference papers to ensure the research was valid.	Studies that did not focus on metaheuristic algorithms and offloading in computing environments were not considered.
The studies must have been published within the last six years to guarantee the information's timeliness and relevance.	Studies that provided incomplete data or only preliminary results were excluded.
The articles needed to be published in English to ensure they could be understood and analyzed consistently.	Any study that was a duplicate of another that was already considered for review was excluded.
The studies had to be fully accessible and complete for a thorough evaluation.	
Only studies demonstrating robust and appropriate methodological approaches were included.	

Table 4
Properties of the GA-based schemes.

Ref	Techs.	App.	contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[56]	GA	IoT	Optimizes task offloading, checks dependencies, reduces energy and delay.	security, cost, and real-world testing issues.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	×
[57]	GADS	IoT	Optimizes execution time and energy, improves QoS.	Optimality not guaranteed, potential hardware strain.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[58]	GA-OA	IoT	Reduces delays, improves request success rate.	Balancing optimal solutions and real-world scalability.	Low	Medium	Medium	×	×	×	✓	✓	×	×	✓
[59]	GA	Cellular Networks	Reduces latency and energy, improves offloading decisions.	Challenges in BS-dense environments.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[60]	GA	IoT	Minimizes task completion time and ensures fairness in user service.	Limited by wireless and MES resource constraints.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[61]	IGA	IoT	Reduces task latency and improves resource utilization.	Risk of resource overconsumption complexity in scheduling.	Medium	High	Medium	×	✓	×	✓	✓	✓	×	×
[62]	IGA	IoT	Reduces total overhead enhances offloading efficiency.	Complexity in real-world scalability and security concerns.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[63]	EGA	VANET	Enhances energy efficiency, supports SLA compliance, and optimizes offloading.	It may not outperform all non-SLA-aware algorithms in energy savings.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[64]	QoS-SLA-AGA	IoV	Enhances offloading speed and respects multiple SLA constraints.	Potential real-world implementation challenges.	High	High	Medium	×	✓	×	✓	✓	×	✓	×
[65]	NGA, MDP	MDs	It enhances offloading, reduces execution time, and lowers power consumption.	It relies on stable network conditions and a complex initial setup.	High	Medium	Medium	×	✓	×	✓	✓	×	×	✓
[66]	GA	IoT	Minimizes average service delay and	It may not address all aspects of	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	×

(continued on next page)

Table 4 (*continued*)

Ref	Techs.	App.	contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
			optimizes server deployment and user offloading.	complexity, limited by server number.											
[67]	GA, Halton Sequence	IoV	Enhances task completion rates ensures resource allocation.	The process may still require significant computational resources.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[68]	IPSGA	VANET	Optimizes DNN model partition and minimizes offloading delay.	Complex tuning, resource dependency on performance.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[69]	DDRL, GA	IoV	Enhances task completion rates and optimizes system utility.	Complexity in real-world implementation and scalability concerns.	High	Medium	Medium	×	×	✓	×	✓	×	×	✓

attributes, and metrics of interest. Metaheuristic algorithms are widely used in offloading to optimize complex problem-solving and improve computational efficiency.

3.1. Offloading process

Task offloading in cloud, fog, and edge computing helps shift complex computational tasks from devices with limited resources to more powerful computing systems. This is essential for improving applications' performance, reducing delays, and reducing device energy use [23]. Decisions about where and how to offload tasks are complicated because network conditions change frequently, tasks have different requirements, and the available computing resources vary [24]. Metaheuristic algorithms are well-suited for this challenge because they are good at finding optimal solutions in complex scenarios with many competing goals. These algorithms adjust their approach dynamically to balance the demands of processing power, speed, energy efficiency, and overall system performance. Using these algorithms, researchers aim to create effective offloading strategies that swiftly respond to user needs while conserving energy. This is particularly important for IoT and mobile computing environments where demands change rapidly. Fig. 6 illustrates a layered architecture for task offloading across cloud, fog, and edge computing layers. It shows how tasks, defined by size and deadline, are processed and offloaded to optimize resource use in transportation, industry, healthcare, and agriculture.

3.2. Algorithms

3.2.1. Genetic algorithm (GA)

The GA [29], is a search heuristic inspired by the natural evolutionary process, drawing on the principles of genetics and survival dynamics observed in nature. This algorithm mimics biological processes such as selection, crossover, and mutation to generate solutions to problems. Each solution in the GA population undergoes evaluation according to a fitness function, and the best-performing solutions are chosen to form new generations through genetic operators. Over the years, GA has been refined and optimized to tackle increasingly complex problems across diverse fields, such as scheduling and computational offloading in distributed systems, demonstrating its versatility and effectiveness in finding optimal solutions.

3.2.2. Particle swarm optimization (PSO)

PSO [30], is a computational algorithm inspired by the social behavior of birds and fish moving in swarms. This method mimics how these animals optimize their movements based on the experiences of individual members and their neighbors. In PSO, each particle represents a potential solution to a problem, navigating the search space by adjusting its position towards the best personal and group findings. The particles update their velocities and positions with each iteration, aiming to discover the optimal solution through cooperation and information sharing. This technique has been successfully applied to various optimization problems, gaining recognition for its efficiency and simplicity.

3.2.3. Ant colony optimization (ACO)

ACO [31], inspired by the foraging behavior of ants. This algorithm is mainly known for solving complex optimization and path-finding problems. Ants in nature find the shortest paths from their colony to food sources by laying down pheromones, which guide other ants to follow the most efficient routes. In ACO, simulated "ants" traverse the problem space, depositing virtual pheromones that strengthen paths leading to solid solutions, while weaker paths evaporate over time. This process iteratively refines the solutions, effectively mimicking the natural decision-making process of ants. ACO has been widely applied to network routing, scheduling, and other optimization tasks, proving a robust and adaptable solution technique.

Table 5
Properties of the PSO-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[85]	GA-PSO	IoT	Enhances resource allocation, optimizes QoS, and reduces costs and energy.	Complexity in real-time decision-making, dynamic conditions.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[86]	PSO	5G- IoT	Enhances energy efficiency and optimizes 5 G data offloading.	It may not scale well with increasing network density.	Medium	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[87]	PSO	AR, AD	Reduces latency and optimizes task completion time and cost.	Challenges with complexity interdependent task structures.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[88]	PSO	IIoT	Reduces delay, balances energy and allocates resources effectively.	Complexity in multi-user and multi-server environments.	Medium	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[89]	NQPSOOM	IoT	Reduces time and energy loss and improves algorithm performance.	Complexity in managing quantum behavior particle populations.	High	High	Medium	×	✓	×	✓	✓	×	✓	×
[90]	PSO	VANET	Enhances service quality, adapts to edge server load, optimizes offloading.	Not fully address network congestion and resource scarcity.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[91]	QPSO	MDs	Reduces energy and completion time and improves task scheduling.	Limited server resources and network bandwidth.	Medium	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[92]	LMPSO	IoT	Enhances user satisfaction and improves resource and processing efficiency.	Potential local optimization traps without Lévy flights.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	✓
[93]	QPSO	VANET	Reduces system overhead and task completion delay.	Relies on a complex algorithm may affect the real-time application.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[94]	DFT, PSO	UAV	Enhances server deployment, adapts to user demand, and increases the offloading rate.	Complex server management UAV deployment challenges.	High	High	Medium	×	×	×	✓	✓	×	✓	×
[95]	PSO	IoT	Optimizes resource use, balances cost and time, reduces latency.	Resource allocation challenges, dynamic requirements.	Medium	High	Medium	×	✓	✓	✓	✓	×	✓	×

Table 6
Properties of the ACO-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[100]	ACO, PSO	IoT	Enhances QoS, balances load, reduces response time.	Communication cost challenges, complex algorithms.	High	High	Medium	×	✓	×	✓	✓	✓	×	×
[106]	SACO	IoT	Enhances latency, reduces offloading time, compares with existing algorithms.	Specific to fog environments, complexity of implementation.	Medium	High	Medium	×	✓	×	✓	✓	✓	×	×
[101]	G-ACA	IoT	Enhances efficiency, reduces iterations, improves solution quality.	Complexity in integrating two algorithms, scalability concerns.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[102]	DQN, QACO	MDs	Enhances cache efficiency, reduces delays, optimizes offloading.	Requires advanced computing capabilities, may not scale universally.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[103]	nRWS, KP01, ACO, GA	IoT	Enhances cache optimization, improves offloading efficacy.	Achieving global optimal solution consistency may be challenging.	High	Medium	Medium	×	×	×	×	×	✓	✓	✓
[104]	EACO	MDs	Reduces energy consumption, manages completion time trade-offs.	Increases completion time, limited to specific task structures.	Medium	Medium	Medium	×	✓	×	×	×	×	×	✓
[105]	ACO	6 G, LEO	Reduces system cost, optimizes resource allocation.	Limited by satellite resources, energy constraints.	Medium	Low	Medium	×	×	✓	✓	×	×	✓	×

Table 7
Properties of the SA-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[107]	GSP	SMDs	Optimizes offloading ratio, CPU speeds, bandwidth, and power for energy efficiency.	Complexity in balancing computation and communication overhead.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[108]	SAA	5 G, SMDs	Optimizes task offloading and resource allocation, enhances utility.	Require complex calibration, high computational overhead.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[109]	DQN-SA	IoMT, IoB	Maximizes utility, reduces delay, optimizes energy use.	Complexity of hybrid algorithms, dependency on precise model parameters.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[110]	SA	IoT	Adjusts time and energy preferences, enhances system performance.	Limited to single-task scenarios, might be complex in multi-task environments.	Medium	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[111]	SA, MILP	IoT	Efficient scheduling, minimizes resource usage, improves timeliness.	Not achieve the absolute optimality of MILP.	Medium	High	High	×	✓	✓	✓	✓	×	✓	×
[112]	SA	IoT	Maximizes system provider profit, assures response time.	Not fully utilize edge resources, limited by node capabilities.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	✓
[113]	ANSSA	SMDs	Balances cost, energy, and performance; enhances computation capabilities.	Costs for collaboration can be prohibitive; real-world application challenges.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[114]	FSAOS	IoT	Balances search processes, minimizes costs, optimizes resource allocation.	Risk of local optima entrapment, search balance difficulty.	Medium	High	Medium	×	✓	✓	✓	✓	×	✓	✓
[115]		MDs	Reduces system cost, optimizes task allocation and resource use.	Complexity in heterogeneous environments, real-world application concerns.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	✓

Table 8
Properties of the WOA-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[122]	MOWOA, MOWOA2	IoT	Balances offloading decisions, improves QoS and solution diversity.	long-term effectiveness untested.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[123]	WOA, FS, LRS	IoT	Reduces latency, energy, and costs in task offloading.	May not scale well in extremely diverse environments.	Medium	Medium	High	×	✓	✓	✓	✓	×	✓	✓
[124]	WOA	IoT	Reduces latency, energy, and cost; optimizes task offloading.	Not fully address security or complex task dependencies.	Medium	High	Medium	×	✓	✓	✓	✓	×	✓	✓
[125]	WOA	MDs	Minimizes energy and response time, optimizes user behavior topology.	Complexity of integration, real-world application challenges.	High	Medium	Medium	×	✓	×	✓	✓	×	×	✓
[126]	HIWDEO	IoT	Minimizes execution delay, reduces energy and cost.	Computationally expensive, single-objective focused, task dependency challenges.	High	Medium	Medium	✓	✓	✓	✓	✓	×	✓	×
[127]	WOA	IoT	Balances privacy protection with resource consumption.	Optimal balance between privacy and performance not ensured.	Medium	Medium	Medium	✓	✓	✓	✓	✓	×	✓	×
[128]	LWH	IoV	Reduces task latency, conserves energy, enhances offloading efficiency.	High mobility in proposed scheme may cause task return failures.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	×
[129]	WOA	IIoT	Enhances resource allocation, improves QoS, optimizes offloading.	Not fully address resource constraints, complex implementation.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	×

Table 9
Properties of the GWO-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[133]	GWO	IoV	Reduces system delay and energy consumption, optimizes MEC usage.	Relies heavily on local MEC capabilities.	Medium	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[134]	GWO	IoT	Optimizes latency, energy, and cost; significant performance gains.	Complexity could limit real-time applications.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[135]	GWO	IoT	Optimizes task scheduling, reduces energy, improves performance.	Complexity in implementation, may require specific conditions.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[136]	DGWO	IoT	Reduces latency, enhances computational efficiency.	Complexity in implementation, relies on precise task sequencing.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[137]	GWO-WOA	IoT	Enhances offloading, considers multiple factors, improves performance.	Complexity in application, not tested in diverse environments.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[138]	GWO	IoT	Enhances scheduling, reduces power, improves resource allocation.	Potential complexity in real-world scaling, limited SLA improvements.	High	Medium	Medium	×	✓	✓	✓	✓	✓	×	×
[139]	BCD-CONGW	VANET, IoT	Reduces latency and energy, optimizes resource allocation.	Limited Adaptability to Sudden Changes, resource Overhead.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	✓
[140]	BGWO	5 G	Reduces deployment cost, minimizes latency, enhances QoE.	Trade-off between cost and latency, NP-hard problem.	High	Medium	Medium	×	×	✓	✓	×	×	×	✓
[141]	PSO, GWO	IoT	Minimizes energy consumption, meets capacity and delay requirements.	Complex hybrid optimization may affect real-time application.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[142]	LSAG	SMDs	Reduces latency, optimizes cost, improves resource allocation.	Complexity in practical implementation, dependency on network infrastructure.	High	Medium	Medium	×	✓	✓	✓	✓	×	×	✓

Table 10
Properties of the CS, Bee, Bat-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[143]	NSGA-II, BA	MDs	Reduces power and delay and optimizes offloading trade-offs.	Limited computing resources may not meet all user needs.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	×
[144]	ODM-BCSA	MDs	Minimizes time, energy, and costs; improves resource allocation.	May increase delays due to network issues.	Medium	High	Medium	×	✓	✓	✓	✓	×	✓	×
[145]	DMOCS-CO	MDs	Reduces latency and energy achieves Pareto optimization.	Depending on simulation results, comparability may vary.	Medium	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[146]	BC	VANET	Reduces execution time and improves reliability in vehicular networks.	Might struggle with highly dynamic environments disconnection issues.	Medium	High	Medium	×	✓	×	✓	✓	×	✓	×
[147]	IBA	UAV	Reduces energy and delay enhances UAV endurance.	It may require complex setup, limited to specific UAV configurations.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[148]	GMCS	MDs	Reduces energy, improves task offloading efficiency, enhances CPU performance.	Potential complexity in practical implementation.	High	Medium	Medium	✓	✓	×	✓	✓	×	✓	×

Table 11
Properties of the HHO and MFO-based schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[149]	DBN, AHHO	MDs	Enhances load balancing, reduces energy, improves execution time and latency.	Requires complex tuning.	High	Medium	Medium	×	✓	×	✓	×	×	✓	×
[150]	PECCO-MFI	IoT	Enhances edge-cloud offloading, considers profit and cost.	Hard optimization problem, non-differentiable objective.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	✓
[151]	AHHO, GCO	MDs	Enhances privacy, minimizes resource use, reduces privacy risks.	Complexity in practical application, privacy quantification issues.	High	Medium	Medium	✓	✓	✓	×	✓	×	✓	×
[152]	HS-HHO, SMA, HHO	MDs	Reduces energy consumption, improves task clustering and system efficiency.	Scalability issues	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×
[153]	HPDO	Wireless Networks	Increases convergence rate, enhances exploitation, solves complex problems.	No optimal performance guarantee, high computational complexity.	High	Medium	Medium	×	×	×	×	✓	×	✓	×
[154]	OBLMFO	IoT	Reduces delay and energy consumption, enhances resource allocation.	Complexity in implementation, potential scalability issues.	High	Medium	Medium	✓	✓	×	✓	✓	✓	×	×
[155]	LMFO, EETOS	IoT	Reduces energy consumption, minimizes end-to-end delay.	Requires balance between fog and cloud resources, potential queuing delays.	Medium	Medium	Medium	×	✓	×	✓	✓	✓	×	✓
[156]	MFO	IoT	Enhances caching efficiency, reduces latency, improves QoS.	Complexity in global coordination, dependency on SDN controller.	High	High	Medium	×	✓	×	✓	✓	×	✓	×

Table 12
Properties of the other schemes.

Ref	Techs.	App.	Contributions	Possible limitations	System Attributes								Environment		
					Complexity	Scalability	Implementation Feasibility	Secure	Energy Efficient	Cost-Effective	Latency-Aware	Adaptable	Fog computing	Edge computing	Cloud computing
[178]	AWDA	MDs	Reduces energy, cost, delay; improves task management.	Complexity in hybrid algorithm integration, reliance on precise modeling.	High	Medium	Medium	×	✓	✓	✓	✓	×	×	✓
[179]	ODTRA, PRL	IIoT	Minimizes task execution time, optimizes offloading performance.	Reliance on high-fidelity digital twins, complex algorithms.	High	Medium	Medium	×	✓	×	✓	✓	×	×	✓
[180]	DJA	IoT	Optimizes offloading delay, enhances resource use, improves reliability.	Complex system architecture.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	×
[181]	OFDM	IoT	Enhances task offloading, optimizes delay-energy tradeoff.	Complex communication requirements.	High	High	Medium	×	✓	×	✓	✓	✓	×	×
[182]	GTA	IoT	Minimizes latency, energy, and cost; improves load balancing.	Not optimize across all MEC scenarios.	Medium	High	Medium	×	✓	✓	✓	✓	×	✓	×
[183]	MO-BFO	IoT	Reduces response time, communication cost; improves load management.	QoS challenges, scalability of the solution.	Medium	Medium	Medium	×	✓	✓	✓	✓	×	✓	×
[184]	HNIO	UAV	Reduces energy, optimizes UAV base station usage.	Not suitable without base stations, complex variable handling.	High	High	Medium	×	✓	×	✓	✓	×	✓	×
[185]	IMOAO, OBL	IoT	Reduces response time, lowers failure rates, enhances offloading efficiency.	Not consistently outperform other algorithms in diverse scenarios.	Medium	High	Medium	×	✓	×	✓	✓	✓	×	×
[186]	MMGOA-TSA	IoT	Optimizes scheduling, balances response time and energy use.	Complexity in deployment, not handle all IoT scales effectively.	High	Medium	Medium	×	✓	×	✓	✓	✓	×	✓
[187]	IBES	VANET	Reduces total offloading cost, integrates multiple computing layers.	May involve complex parameter tuning, scalability issues in dense networks.	High	Medium	Medium	×	✓	✓	✓	✓	×	✓	✓
[188]	MSSAMTO-IoV	IoV	Enhances vehicle service, optimizes task offloading, improves latency.	Limited coverage, complexity in implementation.	High	Medium	Medium	×	✓	×	✓	✓	×	✓	×

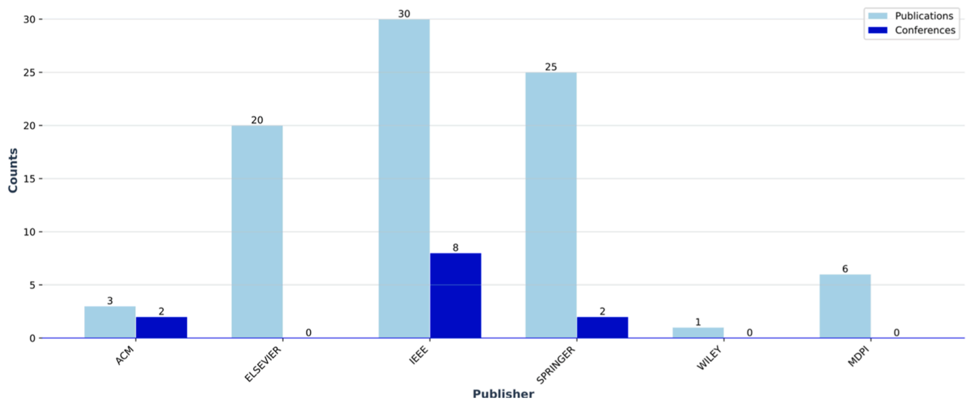

Fig. 3. Applied publishers.

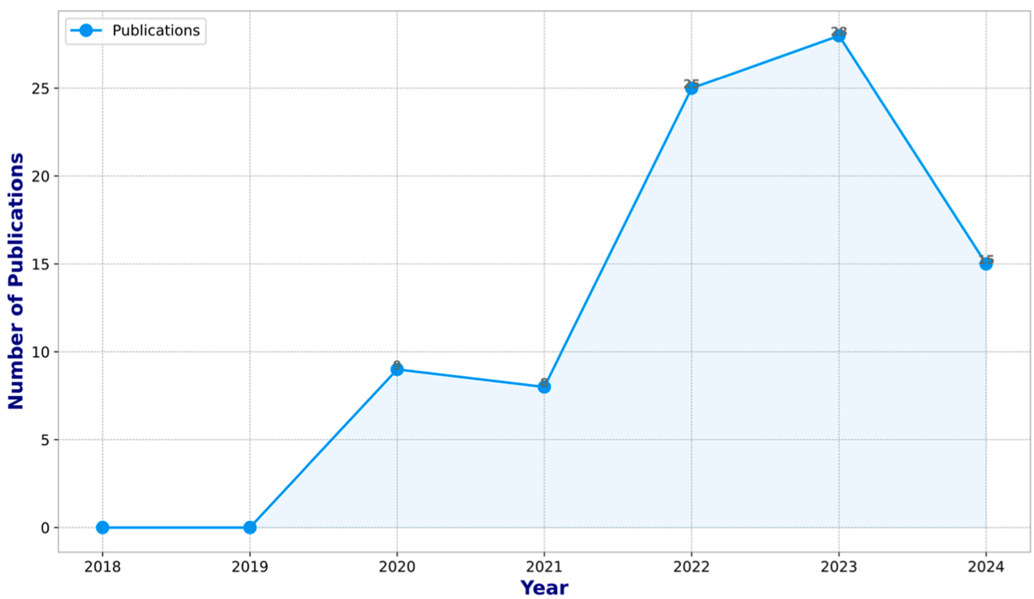

Fig. 4. Number of articles published each year.

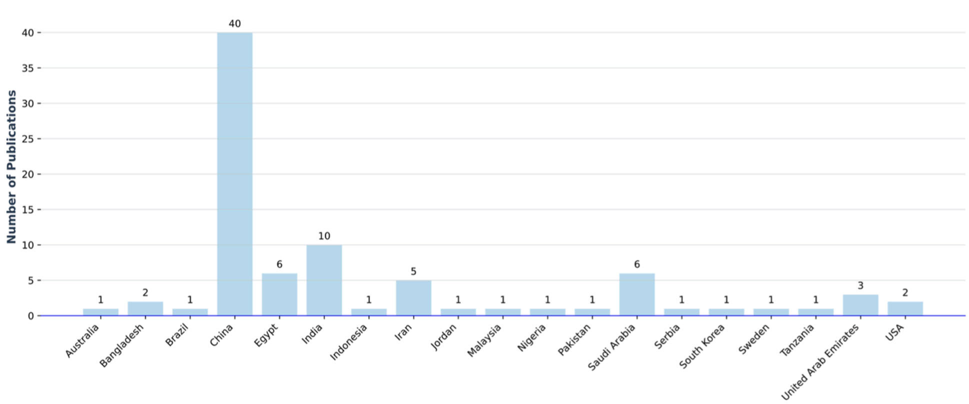

Fig. 5. Number of articles published by each country.

3.2.4. *Simulated annealing (SA)*

SA [32], takes its cue from how metals are treated in metallurgy—-heated and then cooled to fix their structure. In this optimization method, you start hot, allowing random exploration of solutions, and cool down gradually to refine to the best answer. The cooling lets the algorithm consider different solutions, accepting not just the apparent improvements but also, at times, worse ones. This avoids getting trapped in a good solution but not the best. SA is handy in fields like operations research and scheduling, where finding the optimal solution is key and getting stuck in local minima can be a real problem.

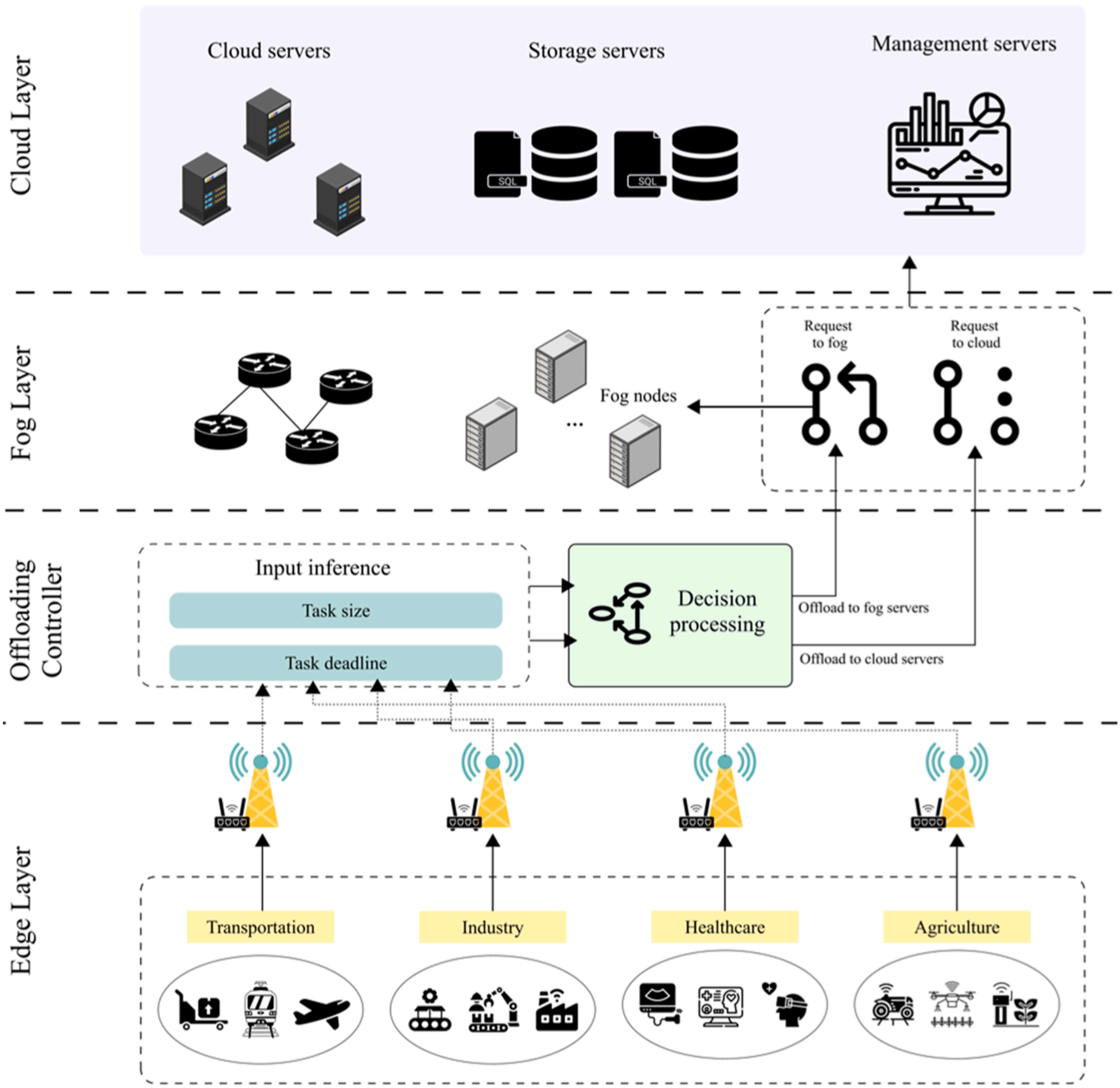

Fig. 6. Hierarchical architecture for task offloading in computing networks.

3.2.5. *Cuckoo search (CS)*

CS [33], is based on how some cuckoos lay their eggs in the nests of other birds. In the algorithm, every egg stands for a solution, and the cuckoo egg is the new solution introduced. The main aim is to swap out weaker solutions in the nest with stronger ones. The CS combines small steps (local search) and bigger, exploring steps (global search), depending on how likely it is to find new solutions. This approach works well for complex problems in areas like engineering because it finds a good balance between searching nearby and exploring more widely.

3.2.6. *Grey wolf optimizer (GWO)*

GWO [34], is inspired by the social hierarchy and hunting techniques of grey wolves. GWO mimics how wolves organize themselves into a pack and coordinate their hunting activities. In GWO, solutions are represented by wolves that assume roles like alpha, beta, and delta, which guide the rest of the pack. The wolves update their positions around the prey, representing the best solution they are pursuing. They adjust their positions to get closer to the best solutions found by the leading wolves. This method has proven effective in solving diverse, complex optimization problems, making it a helpful tool in fields requiring robust solution-searching techniques.

3.2.7. *Whale optimization algorithm (WOA)*

The WOA [34,35], is inspired by the hunting behavior of humpback whales. These whales have a unique way of hunting called the bubble-net feeding method, which WOA mimics. The algorithm simulates this behavior to solve optimization problems. It involves whales encircling prey and creating bubble nets to capture them. In the context of WOA, this translates to identifying the best solution in the problem space and then refining the search area in iterative steps, mimicking the whale's approach to encircle and tighten their path towards their target.

3.2.8. Bee algorithm (BA)

BA [36], is a nature-inspired optimization technique developed by observing honeybees' foraging behavior. It was introduced to simulate how bees search for the best food sources. In this algorithm, artificial bees explore the solution space to find the best solutions, resembling how real bees search for food. The process involves scout bees randomly searching for food sources, which are then evaluated for their nectar amount. The best sources are shared with other bees. Over iterations, the bee colony dynamically adjusts, focusing more on promising areas in the solution space.

3.2.9. Harris hawks optimization (HHO)

HHO [37,38], is inspired by the cooperative behavior and chasing style of Harris hawks in nature, particularly their strategy known as "surprise pounce." This algorithm mimics how hawks encircle, harass, and surprise their prey until they capture or the opportunity is lost. In HHO, potential solutions to optimization problems are considered as hawks that explore and exploit the search space, dynamically adjusting their positions based on the prey's location. Each iteration represents a part of the hunting process, from searching for prey to performing the surprise pounce.

3.2.10. Moth-flame optimization (MFO)

MFO [39], is inspired by the navigation method known as transverse orientation used by moths to travel in a straight path, which interestingly sometimes causes them to spiral toward artificial lights. In MFO, each moth represents a potential solution to a problem, and the flames represent the best solutions. Moths update their positions towards the flames using a mathematical model that simulates their attraction to light. This method is known for its precision and ability to effectively converge on solutions, making it useful in fields requiring complex problem-solving capabilities.

3.3. System attributes

3.3.1. Complexity

Complexity in metaheuristic algorithms refers to how much computer power there is and the detailed steps to make these algorithms work correctly. When an algorithm is more complex, it usually can produce better results, but it also needs more computer resources and a deeper understanding to manage it. This complexity is significant in areas like computational offloading, where these algorithms manage large amounts of data and make many complicated decisions quickly. Therefore, understanding how complex an algorithm is helps determine if it can be effectively used in real-world applications without too many resources.

3.3.2. Scalability

Scalability in metaheuristic algorithms indicates their ability to handle increasing amounts of work or their capability to be enlarged to accommodate that growth. This is crucial for applications that expect to scale up, such as in network systems and extensive data analysis, where the volume of data or the number of tasks can increase significantly over time. A scalable algorithm can efficiently manage growing demands without a loss in performance. This means that the algorithm continues to perform well as the system expands without the need for disproportionate resource increases.

3.3.3. Implementation feasibility

Implementation feasibility in the context of metaheuristic algorithms assesses how practical integrating these algorithms into existing systems and workflows is. This factor considers the technical and logistical aspects of deploying an algorithm, including the availability of necessary technology, compatibility with current infrastructure, and the need for specialized knowledge or training. An algorithm that is easy to implement is crucial for real-world applications, mainly when time and resources are limited. It's important because even the most potent algorithm would be useless if it cannot be practically applied.

3.4. Metrics of interests

3.4.1. Security

Secure approaches in applying metaheuristic algorithms are essential to protect data integrity and privacy, especially in sensitive fields like healthcare, finance, and personal data processing. Security in these algorithms involves designing methods that solve optimization problems and safeguard against potential threats like data breaches and unauthorized access. This involves implementing robust encryption methods, secure data handling practices, and sometimes integrating security directly into the algorithmic design. Ensuring these algorithms are safe is crucial because vulnerabilities could lead to significant risks, including losing sensitive information and exploiting system weaknesses.

3.4.2. Energy efficiency

Energy-efficient approaches in this field are increasingly important as they aim to reduce the power consumption of computing processes. This is particularly critical in environments like data centers, embedded systems, and large-scale computations, where energy costs can significantly impact overall expenses. By optimizing how these algorithms manage computational tasks, they can perform the necessary calculations with minimal energy use, thus enhancing sustainability and reducing operational costs. Energy efficiency in metaheuristics often involves refining algorithmic structures to reduce unnecessary computations, optimizing data flow, and utilizing techniques that allow for faster convergence.

3.4.3. Cost-effectiveness

Incorporating cost-effective strategies into metaheuristic algorithms is vital to maximizing efficiency while minimizing expenditure. This approach is critical when budget constraints are a significant concern, ensuring practical and economically viable solutions. Cost-effective methodologies focus on reducing the resources required—like computational power and time—thus decreasing operational costs. These strategies are essential for maintaining optimal performance in large-scale systems or long-term projects without excessive spending. Therefore, when implementing metaheuristic algorithms, it is essential to consider their cost-effectiveness to ensure that they provide the best possible solution without straining financial resources.

3.4.4. Latency awareness

Latency-aware approaches in this field are designed to minimize data processing and decision-making delays, which are crucial for real-time applications such as autonomous driving, trading, and online data services. These approaches ensure that algorithms solve optimization problems efficiently and respond swiftly to changes in the system or environment. Latency awareness is especially significant in networked and distributed computing environments, where delays can significantly impact performance and user satisfaction. By optimizing algorithms to be responsive and quick, latency-aware strategies help maintain system robustness and ensure timely outcomes, making them vital for sectors where speed is as critical as accuracy.

3.4.5. Adaptability

Adaptable approaches to using metaheuristic algorithms are crucial for ensuring that these algorithms can effectively respond to changes in problem parameters or operational environments. Adaptability allows algorithms to maintain effectiveness across various scenarios and adapt to evolving requirements without needing significant redesign. This flexibility is essential in fields where systems frequently encounter new or shifting challenges, such as dynamic market conditions or technological advancements. Implementing adaptable metaheuristic algorithms means they can be fine-tuned or modified on the fly, enhancing their utility and ensuring sustained performance.

3.5. Handling discrete optimization challenges in metaheuristic algorithms for task offloading

Metaheuristic algorithms are generally developed for continuous optimization problems. However, task offloading involves making discrete decisions, such as choosing the correct nodes for specific tasks. This shift requires substantial changes to metaheuristic methods to effectively manage the discrete nature of task offloading issues. For example, binary and discrete algorithmic changes in some metaheuristic techniques, like GAs and PSO, have been modified into binary or discrete versions to match the binary decision-making needed in task offloading. These changes help the algorithms work correctly in discrete decision environments, which is essential for picking specific servers or pathways for offloading tasks. Moreover, altering algorithmic operations and adapting metaheuristics to discrete problems involves changing their operations. For instance, in GAs, this could mean adjusting the crossover and mutation operations to handle discrete variables better. PSO focuses on tweaking the equations for velocity and position updates to keep solutions within the acceptable discrete range. These adjustments ensure the solutions remain viable and meet the discrete limits of the offloading challenge. Handling infeasible solutions that do not meet the constraints, like resource limits or task dependencies, are dealt with using specific strategies such as penalty functions or solution repair methods. These techniques help adjust unworkable solutions to a feasible range, keeping the metaheuristic algorithms functional in real-world offloading scenarios. Furthermore, hybrid metaheuristic methods blend or combine different metaheuristic strategies with custom heuristics specific to the offloading challenge. For example, integrating the discrete management of GAs with the continuous optimization strengths of PSO can create robust solutions well-suited to the demands of edge computing. These hybrids draw on each method's advantages to effectively tackle continuous and discrete problem elements. Adding domain-specific insights incorporating specific knowledge about computing environments into metaheuristic algorithms can significantly enhance the optimization process. This includes understanding network structures, server capacities, and task details, which steer the optimization towards more practical and achievable solutions. Such targeted information allows the metaheuristic methods to better address the unique needs and restrictions of task offloading.

3.6. Addressing task offloading in dynamic environments

The success of task-offloading methods depends significantly on whether the environment changes. Unlike static ones, network conditions, device movement, and the amount of computing needed can change constantly in dynamic settings. This means offloading methods must be able to adjust quickly to keep the system working well and reliably. Traditional metaheuristic strategies are usually made for static environments with the same parameters. However, in dynamic environments like MEC, offloading algorithms need to be able to adapt continuously. This may require using real-time data, feedback loops, and predictive models that respond to changes and anticipate future situations to improve offloading decisions. Using past data also helps offloading strategies become more adaptable by identifying trends in network use, device activity, and task demands. By learning from what has happened before, algorithms can better predict future conditions and adjust offloading decisions proactively. Understanding historical trends is vital in dynamic environments to prepare for changes in expected network or device conditions. Strategies designed for static settings might not work well in dynamic ones unless they are significantly altered to include adaptability and real-time decision-making. The approach we propose addresses how it adapts to dynamic environments. This includes explaining how it manages real-time network and computing resource changes, uses historical data to make predictions, and remains flexible in decision-making. Lastly, the effect of dynamic adaptation on the overall system performance needs to be closely examined. This includes exploring how well the approach keeps service quality high, reduces delays, and uses resources efficiently under varying conditions. The performance measures show the offloading strategy's capability to handle environmental changes effectively.

4. Analysis, taxonomy, and state-of-the-art analysis

This section thoroughly examines the collected literature to understand the different strategies and techniques used in offloading with metaheuristic algorithms. We aim to create a structured classification that arranges the existing studies into a clear and understandable taxonomy. This organization will enable us to see the current trends, recognize gaps, and suggest future directions for research in this area.

4.1. Taxonomy development

In the taxonomy development section of our review, we categorize various offloading schemes based on the metaheuristic algorithms they employ, facilitating a systematic exploration of each strategy to compare effectiveness and identify optimal contexts. We distinguish two main paradigms: evolutionary algorithms and swarm intelligence. Drawing from natural evolutionary processes like selection, mutation, and recombination, evolutionary algorithms allow a population of potential solutions to evolve, enhancing adaptability and robustness. They excel in complex, dynamic environments, solving diverse problems from route planning to machine learning. Inspired by collective behaviors in nature, such as ant colonies and fish schools, Swarm intelligence employs simple rules that lead to effective solutions for complex tasks requiring robust, scalable solutions, making them ideal for network optimization and resource management. Fig. 7 categorizes metaheuristic algorithms used for offloading strategies into two main groups: Evolutionary Algorithms and Swarm Intelligence, outlining specific methods under each category.

4.2. State-of-the-art studies

This section reviews the methods and provides a detailed analysis of various offloading frameworks that have used metaheuristic algorithms to optimize different metrics. This thorough evaluation aims to understand how offloading strategies can be effectively implemented to enhance the performance and reliability of applications like IoT, Mobile Devices (MDs), Vehicular Ad Hoc Network (VANET), Unmanned Aerial Vehicles (UAVs), and others in a range of scenarios. In our systematic review, the tasks are dependent. The dependencies arise primarily from the coordination required among various computing nodes and the data continuity between tasks across different servers, which is critical for completing the offloading successfully. In this field, the task offloading strategies show strong connections between tasks, mainly because they need coordinated data sharing and resource distribution among different computing nodes. These relationships are apparent in several parts of the methods used. For example, completing certain tasks depends on data already processed or stored on different edge servers. This creates a dependency, as the output from one node impacts what is needed and how tasks are carried out afterward. Moreover, methods like choosing edge servers dynamically rely on integrating data from several nodes to finish tasks, highlighting how interconnected these tasks are within these systems. This maintains data flow and improves overall system performance by using distributed resources efficiently. Furthermore, this dependency is also clear in systems designed to offload computing tasks that must pass through multiple nodes. Each node relies on the others to carry out tasks and pass on data. The efforts to optimize execution time and energy use at various edge locations also underline

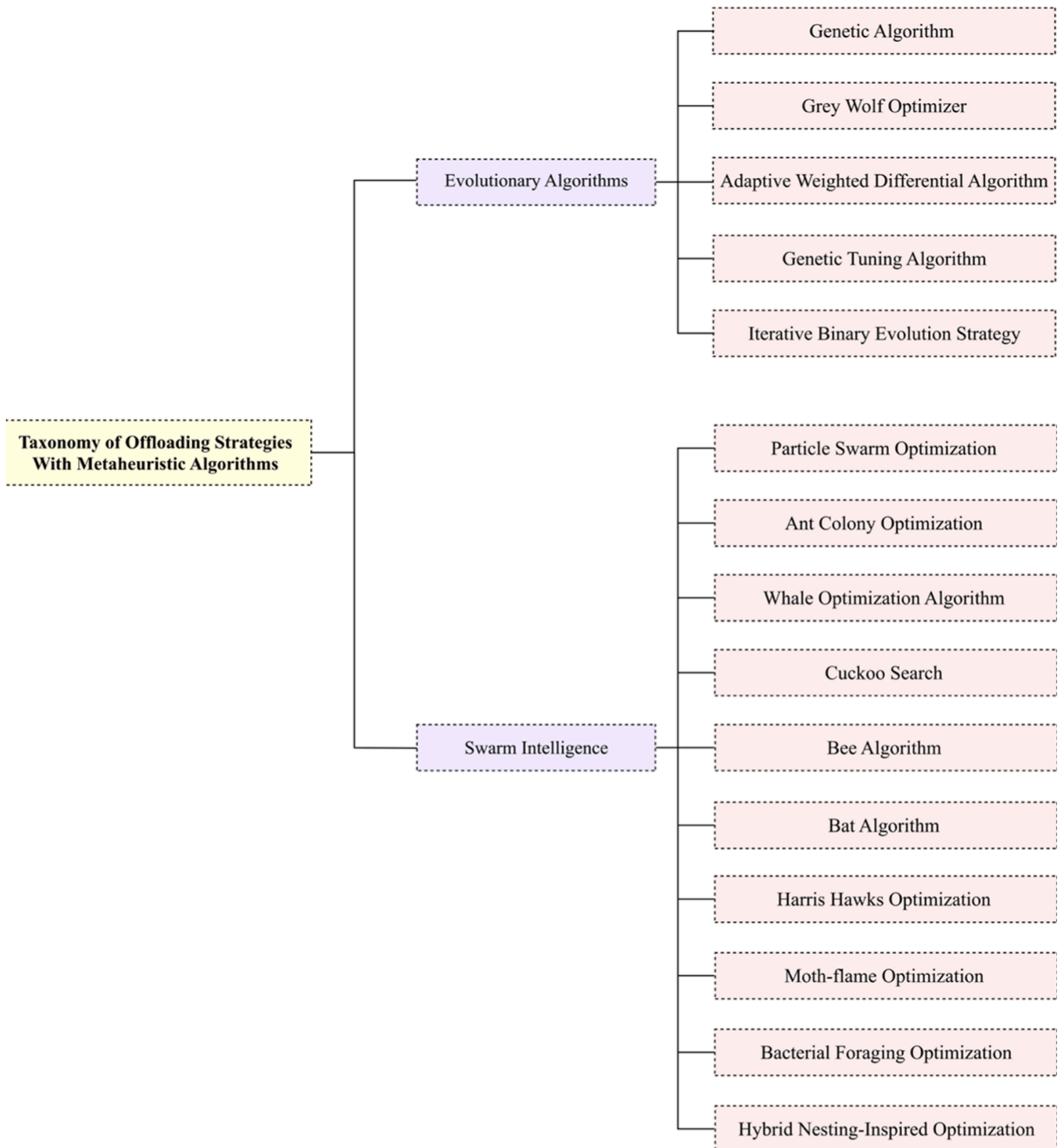

Fig. 7. Taxonomy of Offloading Strategies with metaheuristic-based algorithms.

the task dependencies, as these efforts need tasks to be distributed strategically based on what each site can handle and has available. This ensures resources are used well, and tasks are completed on time.

4.2.1. GA-based schemes

The literature features several proposed schemes based on GA, referenced in [40–55]. In this section, we have reviewed and selected these schemes. To start with, Chakraborty et al. [56] proposed a method for addressing the resource constraints of MDs in sensor MEC by offloading sensor tasks to edge servers. Their study highlights that task execution depends on data from previously utilized edge servers along MD trajectories. They introduced a dynamic edge server selection scheme that integrates data across multiple nodes to complete tasks, employing a GA for optimal results. Their method aimed to reduce device load by shifting computational work from mobile devices to edge servers. This approach lowered energy use and sped up computations while minimizing transmission delays. However, it faced issues handling large-scale operations and relied heavily on stable network connections to keep data moving between devices and edge servers. Rajapackiyam et al. [57] presented an edge-based approach where tasks are distributed strategically across various edge sites using a modified GADS to optimize execution time and energy usage, demonstrating efficiency improvements of 1–15 %. This approach focused on improving execution time and energy consumption, achieving efficiency gains from 1–15 %. Nevertheless, the study faced challenges such as potential scalability problems when expanding the network size and reliance on initial genetic diversity, which might influence the uniformity of outcomes in various implementations. Moreover, Hussain et al. [58] introduced GA-OA to efficiently manage traffic in an IoT infrastructure-cloud environment, addressing the problem of congestion and inefficiencies in data transmission and aiming to minimize delays and boost the success of IoT requests. Their methodology distributed the GA’s fitness process across gateways and infrastructure to balance optimal and sub-optimal solutions, significantly enhancing response rates and reducing complexity and delays.

In their publication, Liao et al. [59] tackled the challenges of ultra-dense cellular networks by formulating a multi-user-to-multi-servers problem, which was addressed through a binary-coded GA. This approach successfully reduced latency and energy consumption. Furthermore, Li et al. [60] utilized MEC to enhance service experiences for smart devices through a joint optimization strategy using a GA. This approach effectively minimized task completion times and maintained fairness among users by efficiently allocating tasks, channel bandwidth, and computing resources of mobile edge servers. Liu et al. [61] developed a framework for distributed multi-hop computing task offloading using GA, aiming to address the challenge of resource scarcity in complex network architectures where tasks must hop across multiple computing nodes. Their method outperformed traditional strategies and, improving resource utilization and reducing computation delays. Moreover, Zhu et al. [62] tackled the challenge of optimizing computational offloading in edge computing environments, specifically addressing how user decisions on where to offload tasks, the power used in uplink communications, and the allocation of MEC resources can be optimized to reduce operational costs and enhance system efficiency. This approach enhanced the optimization capabilities of a standard GA, significantly reducing costs and improving efficiency. Materwala et al. [63] aims to reduce energy consumption and SLA violations in VANETs by integrating edge and cloud computing platforms using an Evolutionary Genetic Algorithm (EGA) with an adaptive penalty function. Their solution also achieved notable energy savings and fewer Service Level Agreement (SLA) violations. Also, Materwala et al. [64] Also, Zalat et al. proposed a QoS-SLA-aware adaptive GA aimed at addressing the challenges associated with ensuring QoS and adherence to service-level agreements in the IoV. This method focused on multi-request overlaps and variable vehicle speeds. This method led to significant performance improvements and reduced SLA violations. Similarly, Zalat et al. [65] addressed the challenge of efficiently managing computational resources in multisite environments, where mobile applications often need to distribute their computing tasks across multiple locations to optimize performance. Their focus was on enhancing the decision-making process for task distribution, aiming to minimize energy usage, reduce execution delays, and increase data offloading efficiency among different computing sites. In other research paper, Song et al. [66] explored edge computing's ability to facilitate quicker task completion through strategic server placement and user offloading strategies, utilizing a GA-based approach that effectively reduces average service delay. Wang et al. [67] tackled IoV task scheduling limitations by employing an improved GA that differentiated communication and computation models and maximized system utility. This method significantly enhanced task completion rates. Finally, Li et al. [68] explored how to run complex, time-critical deep learning tasks on devices with limited resources by proposing a DNN model partition and offloading strategy optimized through an improved PSGA. Moreover, Jin et al. [69] enhanced IoV task scheduling models using a Distributed Deep Reinforcement Learning (DDRL)-Based and Genetic Optimization Algorithm GOA, significantly improving job completion rates and reducing system costs.

Upon reviewing recent progress in computational offloading across diverse environments, it becomes clear that while methods based on GA offer considerable improvements in efficiency and effectiveness, they also encounter a range of strengths and weaknesses. Based on the analysis of the reviewed paper in this section, the method proposed in [59] It stands out due to its ability to reduce latency and energy while improving offloading decisions, boasting high scalability and adaptability in edge computing environments. Other study [60] referenced as in IoT is notable for minimizing task completion times and ensuring fairness, with high scalability and effective performance in fog and edge computing. The proposed scheme in [63] excels in enhancing energy efficiency and supporting SLA compliance, demonstrating high scalability and good performance in fog and edge settings. The presented algorithm in [64] improves offloading speed and accommodates multiple SLA constraints, with high scalability and adaptability in fog and edge computing. Lastly, [67] GA with Halton Sequence in IoV enhances task completion rates and ensures optimal resource allocation, which is marked by medium complexity and high scalability and is suitable for edge computing environments. These algorithms demonstrate a robust balance between enhancing system performance and maintaining scalability and adaptability across various computing environments. These methods strive to enhance key performance indicators such as energy efficiency, latency, and task execution; however, they face unique challenges in different application domains. The complexity and feasibility of implementing these strategies often reflect the specific requirements of the environments in which they are deployed. For example, approaches within IoT settings focus on reducing energy consumption and latency, which are essential for the effective operation of real-time data processing and communication. Nevertheless, these strategies must also address issues like security and scalability challenges in real-world applications. Scalability is a major concern, especially in environments with complex network topologies like dense cellular networks or VANETs, where unpredictable conditions can complicate matters further. While most methods aim to improve aspects such as energy efficiency and cost-effectiveness, the compromises between these advantages and the practical challenges of security and feasibility should not be ignored. For instance, techniques that significantly cut power consumption and reduce execution times might still be inadequate for real-world applications due to their complexity or lack of sufficient security features. The suitability of GA-based solutions across different computing frameworks like fog, edge, and cloud computing highlights an important direction for future research. Each computing type presents unique challenges and requirements, indicating that a universal solution is not feasible. Customized solutions that consider the particular characteristics and needs of each setting are likely essential to fully utilize the advantages of GA approaches while overcoming their intrinsic limitations, thus improving their robustness, security, and practical applicability in future studies.

4.2.2. PSO-based schemes

Numerous schemes utilizing PSO are documented in the literature, including those listed in [70–84], and we review a selection of these in this section. To initiate the discussion, Kumar et al. [85] presented a dual-objective optimization framework addressing the allocation of computational resources in network architectures to improve the QoS while minimizing operational costs and energy consumption. Integrating a hybrid GA-PSO meta-heuristic, their model demonstrated superior cost and energy efficiency offloading, outperforming three prior methods, thereby confirming its practicality and effectiveness. Bacanin et al. [86] aimed to tackle the challenge of managing energy consumption in 5 G networks, a critical issue given the exponential increase in data traffic and the need for real-time processing. Their implementation of a multi-level offloading mechanism using PSO demonstrated that MEC could be more energy-efficient than traditional cloud processing. Subsequently, Ma et al. [87] proposed a multi-objective optimization approach to address the challenge of delivering low-latency services essential for latency-sensitive applications like augmented reality and autonomous driving, which require rapid data processing and response times to function effectively. This strategy, rooted in multi-access edge computing, significantly reduced end-to-end latency in simulations, enhancing performance by considering task time and cost. Furthermore, You et al. [88] Chen et al. designed a PSO-based offloading strategy for the IIoT to efficiently manage delay-sensitive and resource-intensive tasks. Their approach, which balances energy consumption and optimizes resource distribution, showcased superior efficiency over genetic and SA algorithms through simulations. Moreover, Chen et al. [89] addressed time consumption and energy loss in edge computing perceptual task offloading with the NQPSOOM method. This method enhanced particle population diversity and optimized performance by incorporating logistic chaos perturbation and fast elite non-dominated sorting GA operations, showing adaptability, time consumption, and energy loss reductions.

In their publication, Alqarni et al. [90] introduced a smart PSO model to enhance service quality in MEC environments supporting vehicular services. Their model, which considers vehicle movement and edge coverage, adapted to edge server workloads and utilized advances in GPU architectures, demonstrating improved service quality. Similarly, Dong et al. [91] employed particle swarm and quantum PSO-based strategies for task offloading optimization in MEC. This strategy effectively managed the constrained resources and bandwidth of MEC servers, reducing energy consumption, task completion time, and processing time more effectively than several advanced methods. Gao et al. [92] tackled the challenge of managing latency and optimizing resource allocation in a hybrid edge-cloud computing environment to improve user satisfaction by ensuring quicker response times and more efficient utilization of computing resources across both the cloud and edge layers. Their LMPSO method combines PSO with Lévy flight, improved latency responses, and execution performance in simulated tests. In a research study, Shu et al. [93] addressed the significant challenges of maintaining high-quality user experiences in intelligent transportation systems by focusing on reducing delays and conserving energy in vehicular networks through an enhanced quantum PSO algorithm that improves task execution optimization and search capabilities. Moreover, Zhang et al. [94] aimed to address the challenge of optimizing the placement and utilization of UAVs as mobile edge servers in urban areas. Employing PSO for UAV deployment, their approach confirmed an enhanced task offloading rate through trace-driven experiments, leveraging UAVs in dynamic urban centers. Yang et al. [95] proposed an intelligent two-tier task scheduling method tailored for cloud-edge collaborative computing to address the challenges of high computational demands and application performance enhancement.

Their method categorizes tasks between cloud and edge based on their types and dynamically offloads tasks to the cloud during high edge loads. They employed multi-objective optimization using a particle swarm algorithm to achieve an optimal balance between time and cost. Experimental outcomes demonstrated improved resource utilization, reduced data upload, and cost control while maintaining low latency.

While PSO techniques prove generally effective, they face significant challenges related to complexity and scalability across diverse environments like IoT, 5 G, and advanced vehicular networks. Based on the review of the paper in this section, the method suggested in [85] enhances resource allocation and QoS while reducing costs and energy, despite challenges in real-time decision-making under dynamic conditions. Other study referenced as [92] boosts user satisfaction and resource efficiency, leveraging Lévy flights to avoid optimization traps, marked by high scalability and adaptability in fog and edge computing. In [88] the presented PSO method effectively reduces delays and balances energy, although it deals with complex multi-user environments, performing well in edge computing. Moreover the algorithm which introduced in study [95] optimizes resource use, balancing cost and time, and excels in edge computing, addressing dynamic resource allocation challenges. Lastly, the scheme in [90] improves service quality and adapts to varying edge server loads, ideal for edge computing. Often, these methods encounter practical implementation issues tied to the inherent limitations of the technologies and the dynamic nature of network architectures. For example, strategies focused on improving resource allocation and service quality frequently struggle with real-time decision-making due to network environments' dynamic and complex conditions. This issue is particularly pronounced in dense networks and IoT applications where scale and density significantly impact performance. Moreover, although these approaches are designed to be adaptable and latency-aware, they face scalability challenges in rapidly expanding network infrastructures, complicating their deployment across different computing models like fog, edge, and cloud computing, each requiring tailored solutions. Security also poses a major concern; integrating these computational techniques into broader network systems must not compromise robust protection against potential breaches. Additionally, managing the trade-offs between cost-effectiveness and energy efficiency is crucial to optimizing performance without detriment.

4.2.3. ACO-based schemes

The literature proposes various schemes based on ACO, listed in references [96–99], and we have reviewed some of these here. To begin with, Hussein et al. [100] proposed two scheduling algorithms based on ACO and PSO. The authors addressed the challenge of optimizing task distribution across fog nodes in dynamic and distributed computing environments. The main problem they aimed to solve was enhancing the efficiency of resource allocation, reducing response times, and improving load balancing within fog computing networks, which are critical for maintaining high service quality and system reliability, with ACO showing notable improvements in response times and load balancing compared to PSO and traditional round robin methods. In another study, Kishor et al. [97] introduced a smart ACO scheduler. Their study highlighted SACO's capability to significantly reduce latency, outperforming conventional scheduling algorithms and other bio-inspired algorithms, such as modified PSO and the Bee Life Algorithm. Xu et al. [101] developed a hybrid method combining genetic and ant colony algorithms to overcome the limitations of standalone heuristic algorithms, which often get stuck in local optima. This approach leverages GA techniques (selection, crossover, mutation) to generate initial solutions, further refined by the ant colony algorithm, leading to fewer iterations and improved performance in edge computing scenarios. Furthermore, Li et al. [102] proposed an MEC system with a dual strategy of content caching and computing offloading to manage the demands of computation-intensive and time-sensitive tasks. They utilized DQN to derive optimal caching strategies. They introduced a quantum ant colony-based offloading strategy, which demonstrated improvements in cache hit rates and reductions in transmission delays and overall system costs.

In their publication, Zulfa et al. [103] aimed to address the challenge of efficiently managing the distribution and offloading of data across edge network devices. The fundamental problem they sought to solve was optimizing the allocation of limited edge cache resources to enhance data accessibility and reduce latency. Their approach, which utilized the Knapsack Problem 0/1 (KP01), was designed to prioritize which data should be stored or offloaded in edge caches to maximize efficiency and performance. Moreover, Danial et al. [104] focused on reducing energy consumption in mobile cloud computing applications through their energy-efficient ACO. This algorithm, while potentially increasing completion times, significantly reduces energy usage, thus extending the battery lifespan of mobile devices. Finally, Wang et al. [105] tackled the challenges of low-earth orbit (LEO) satellite edge computing, which is critical for the 6th generation mobile network. They developed a system cost model and an offloading strategy based on an enhanced ant colony algorithm to optimize local resource allocation and task queuing, reducing system costs.

ACO-based schemes have become effective solutions in various computational environments, enhancing efficiency, reducing latency, and improving QoS. Based on the above information and our evaluation, the proposed model in [100] excels in enhancing QoS, balancing loads, and reducing response times. Another study in [106] which proposed SACO in IoT stands out for improving latency and offloading times with high scalability and adaptability, optimized specifically for fog environments. Moreover the proposed algorithm in [101] (G-ACA) IoT environment significantly improves efficiency and solution quality, balancing complexity and scalability, and is adaptable in edge computing settings. The proposed combined algorithm in the study [102] (DQN, QACO) boosts cache efficiency and reduces delays in edge environments. Lastly, [103] nRWS, KP01, ACO, GA in IoT enhances cache optimization and offloading efficacy, demonstrating adaptability across fog, edge, and cloud computing. These schemes are primarily applied in IoT, mobile devices, and the burgeoning field of 6 G networks, aiming to optimize resource allocation and response times. Despite their potential, ACO-based methods face challenges related to their complexity and scalability. When combined with other algorithms like PSO and GA, ACO aims to balance loads and decrease response times effectively, particularly in settings that demand high QoS. Nevertheless, these hybrid models can incur high communication costs and complex algorithmic structures, which may restrict their broad implementation. For instance, while IoT applications benefit from improved cache efficiency and offloading efficacy, maintaining consistent global optimal solutions is challenging due to the complexity of the algorithms. Moreover, scalability and practical implementation are major concerns, as systems requiring advanced computing capabilities, such as DQN and QACO, may not be universally applicable, particularly in resource-limited environments. Even though ACO schemes are designed to be adaptable and latency-aware, their deployment across varied computing models—like fog, edge, and cloud computing—often encounters real-world limitations. Furthermore, enhancing security and energy efficiency is crucial, especially in high-demand areas like 6 G and LEO satellite networks, where resource and energy constraints are significant.

4.2.4. SA-based schemes

Several SA-based schemes have been suggested in the literature, and we discuss a selection of these in this section. In their publication, Bi et al. [107] addressed the challenge of managing high computational demands on Smart Mobile Devices (SMDs) while conserving battery life and minimizing network traffic. They used a hybrid genetic SA-based PSO, achieving faster convergence and reduced energy use compared to three other methods. Similarly, Huang et al. [108] focused on addressing task offloading and computing resource distribution problem within 5 G small cell networks. Employing a dual approach of the Lagrange multiplier and an enhanced SA algorithm, their method improved task completion times and total weighted energy consumption, showing superior results under various conditions. In their study, Yuan et al. [109] addressed the challenge of efficiently managing large-scale data flows and ensuring timely data processing in healthcare applications. They focused on optimizing how medical data from various IoT devices is accessed and used for personalized healthcare. This is a critical issue given the growing volume of patient data and the need for real-time, optimized patient treatment plans. Their work aimed to enhance healthcare systems' responsiveness and energy efficiency by utilizing the Internet of Medical Things (IoMT). Their offloading algorithm for MEC, which integrates a DQN with SA, enhanced system utilization while reducing processing delays and energy consumption.

Moreover, Li et al. [110] proposed an SA-based optimization algorithm for MEC that balances time and energy by offloading tasks from terminal devices to edge service nodes. This method demonstrated adaptability and improved system performance in real-world scenarios. Furthermore, Mahjoubi et al. [111] developed an SA-based method within a three-layer MEC architecture to address IoT challenges in digitalizing industries. Their solution, formulated as a Mixed-Integer Linear Programming (MILP) problem, aimed to enhance task scheduling across IoT devices, edge servers, and cloud environments. Yuan et al. [112] Xu et al. created a framework to enhance IoT through edge computing, focusing on maximizing provider profit while ensuring response times. Their offloading algorithm for cloud-edge systems utilized a simulated-annealing-based optimization, demonstrating profitability and effective resource balancing in real-world applications. Researchers in [113] introduced a cost-based model that integrates mobile devices and edge servers to enhance computational power within edge networks. Their adaptive neighborhood search with SA-optimized task offloading and resource allocation showed significant enhancements in energy and computational cost reductions during extensive testing. In their publication, Gabi et al. [114] proposed a fruit fly-based SA optimization scheme (FSAOS) for mobile edge-cloud continuum resource allocation, aiming to address the challenge of efficiently allocating computational resources across the mobile edge and cloud continuum. This scheme effectively balanced local and global searches, achieving optimal service quality and reducing execution costs, as confirmed by tests on the EdgeCloudSim Simulator. Finally, Yuan et al. [115] focused on optimizing energy consumption in computation-intensive environments, particularly in data centers where managing operational costs and computational efficiency are critical. They developed an energy-minimized partial computation offloading method using a hybrid meta-heuristic algorithm that combines SA's Metropolis criterion with genetic operations. Their method, tested using Google data center tasks, significantly reduced system costs and improved convergence speed, showcasing its efficacy compared to standard benchmarks.

SA-based schemes are gaining recognition for optimizing computational tasks in diverse settings, particularly IoT, SMDs, and emerging 5 G technologies. Based on the above analysis, the study presented in [111] excels in efficient scheduling, minimizing resource usage, and improving timeliness, distinguished by its high scalability and adaptability in fog and edge computing environments. Moreover, the authors proposed a model in [114] that stands out for its ability to balance search processes, minimize costs, and optimize resource allocation with high scalability and effectiveness in fog and edge settings. In other research paper cited as [108] the proposed SAA in 5 G, SMDs optimizes task offloading and resource allocation, enhancing utility with good scalability and performance in edge environments. Finally, [109] DQN-SA in IoMT, IoB maximizes utility, reduces delays, and optimizes energy usage, compared to other proposed schemes with notable energy efficiency and latency awareness suitable for edge computing. The proposed approaches in this section focus on optimizing offloading ratios, CPU speeds, bandwidth, and power to enhance energy efficiency and manage resource allocation effectively. However, the implementation of SA is often complicated by its inherent complexity and the significant computational overhead required. SA-based methods are valued for their flexibility to adapt to different operational demands, improving system performance by fine-tuning task execution and resource distribution. For example, SA effectively balances energy consumption against performance needs in IoT and SMD scenarios, helping systems operate within optimal parameters. Nonetheless, this balance is frequently challenged by the complexities of managing both computation and communication overhead, which can pose difficulties in real-world applications. Additionally, the scalability of SA schemes often becomes problematic in environments like 5 G or integrated IoT systems, where variable network density and demand may necessitate complex calibrations and depend heavily on precise model parameters, restricting their broad applicability. SA is designed to be adaptable and latency-aware. Yet, it occasionally struggles to fully utilize resources, particularly in edge computing contexts where resource limitations and node capabilities can limit practical deployment. This issue is even more pronounced in multi-task settings or heterogeneous networks where balancing cost, energy efficiency, and performance is crucial.

4.2.5. WOA-based schemes

The literature contains a range of suggested schemes based on WOA found in [116–121], which we have reviewed some of them in this section. For example, Huang et al. [122] introduced the Multi-Objective WOA to optimize time and energy in MEC under constrained network conditions. The authors aimed to address the challenge of optimizing computational task offloading in terms of both time and energy efficiency within MEC environments, particularly under network conditions characterized by limited bandwidth and high latency. Hosny et al. [123] addressed the challenge of managing dependent tasks in multi-edge cloud environments, where tasks have interdependencies that complicate offloading. They proposed a WOA-based method to manage the offloading of dependent tasks in multi-edge cloud environments. By incorporating frameshifting and a load redistribution strategy, EWA reduced latency, energy consumption, and costs, outperforming the standard WOA. In a subsequent study, Hosny et al. [124] tackled the inherent limitations of IoT and mobile devices, such as limited computing power and battery life. They proposed the refined WOA to enhance QoS by reducing application latency and resource costs, improving over other optimization methods. In another study, Anoop et al. [125] introduced a topology optimized for mobile user behavior to reduce response times and energy consumption amid growing mobile demands. Their exploitation WOA, which integrates differential evaluation techniques, showed enhanced accuracy and speed in applications like puzzle solving and face detection when tested on VMware. Li et al. [126] proposed the Immune Whale Differential Evolution Optimization algorithm (HIWDEO) for managing task dependencies in MEC, effectively minimizing execution delays and reducing energy and costs. Moreover, Liu et al. [127] focused on privacy problems and concerns in MEC by developing a privacy-preserving computing offloading scheme using the WOA. The scheme, which uses differential privacy to protect user location data, balances privacy with resource consumption and has proven effective in managing cost and privacy. In other research, Bi et al. [128] addressed the unique challenges of the IoV using an enhanced adaptive Lévy flight-based WOA algorithm with hierarchical learning. This method facilitates task offloading to fog servers or idle vehicles and shows superior global search capabilities compared to five other prevalent algorithms. Lastly, Kumar et al. [129] proposed an AI-enabled, sustainable framework for an optimized, multi-layered, integrated cloud-fog environment. The primary problem addressed by the authors was the inefficiency in managing QoS parameters tailored for IIoT applications, featuring a fuzzy-based offloading controller and incorporating the WOA to significantly improve QoS parameters like makespan time, energy consumption, and execution costs.

WOA-based schemes are increasingly used to enhance computational tasks across domains such as IoT, MDs, and IIoT. These methods are valued for balancing offloading decisions, improving QoS, enhancing solution diversity, and reducing latency, energy consumption, and costs. However, WOA-based strategies face significant complexity, scalability, and practical application challenges. For example, while WOA combined with feature selection and learning rate strategies has shown promise in reducing task latency and energy usage these benefits may be limited in highly diverse or dynamic environments where scalability is a concern. Moreover, integrating WOA into existing frameworks can be complicated, often requiring substantial computational resources and facing difficulties addressing security or intricate task dependencies. Issues like balancing privacy with resource consumption underscore the challenges in achieving optimal outcomes without compromising other vital system attributes. The reviewed studies in this section reveal that the proposed method in [124] is better in reducing latency, energy, and costs while optimizing task offloading in fog and edge environments than other studies. Furthermore, in [123], the authors proposed that WOA, FS, and LRS in IoT significantly reduce latency, energy, and costs, with high implementation feasibility and effective performance in edge computing. The proposed scheme in [122] stands out for balancing offloading decisions and enhancing QoS with high scalability, primarily effective in edge computing. In [126], the presented HIWDEO in IoT minimizes execution delays and optimizes energy and cost efficiency, which is noted for its security and adaptability in edge computing environments. Lastly, [127] WOA in IoT effectively balances privacy protection with resource consumption, supported by its comprehensive positive attributes in security, energy efficiency, and adaptability, suitable for edge computing applications. Additionally, some implementations, such as HIWDEO in IoT, focus on single-objective optimizations like minimizing execution delays, which might not sufficiently address scenarios involving interdependent tasks. In high-mobility contexts like the IoV, WOA-based strategies may encounter problems such as task return failures due to the scheme's mobility.

4.2.6. GWO-based schemes

Recently, the literature has highlighted various GWO-based schemes documented in references [130–132]. This section explores some of these recent proposals. Zhang et al. [133] developed an adaptive offloading strategy using GWO for vehicular edge computing with an enhanced gray-wolf algorithm. This strategy, which does not rely on cloud computing, aims to optimize computing delays, energy usage, and MEC server capacity, effectively reducing system delays and energy consumption. Nujhat et al. [134] introduced a multi-objective optimization framework for offloading tasks from mobile devices to edge servers. The primary problem their framework, WOLVERINE, addressed was the challenge of balancing the often-conflicting objectives of minimizing latency, reducing energy consumption, and cutting operational costs in MEC environments. Their framework optimizes latency, energy, and cost, achieving notable performance improvements in experimental settings. Furthermore, Shang et al. [135] proposed a refined OPGWO incorporating Latin hypercube sampling for population generation and an orthogonal inverse strategy during optimization. They also introduced a resource allocation method using a V-function mapping policy, significantly enhancing performance and energy efficiency. Tang et al. [136] focused on reducing latency in MEC environments with battery less devices. They introduced a discrete GWO that uses a task permutation sequence for efficient offloading, enhancing system performance beyond conventional metaheuristics. In their publication, Feng et al. [137] enhanced an existing offloading model by normalizing variables to reduce dimensional disparities and introduced a hybrid algorithm combining GWO and WOA. This hybrid, GWO-WOA, outperformed traditional methods in tests. Hashemi et al. [138] developed a multi-objective GWO approach for service scheduling and task offloading in fog computing. Their method, tested in iFogSim on up to 5000 nodes, showed significant improvements in execution times and resource reallocation. Similarly, Cong et al. [139] aimed to address the critical challenges of computational offloading in scenarios where minimizing both latency and energy consumption is paramount for maintaining efficient operation and extending battery life of mobile devices in edge computing environments. They introduced an offloading model that evaluates tasks via a loss function affected by latency and energy. They developed a method using Block Coordinate Descent with convex optimization and GWO, demonstrating superior latency and energy metrics performance. Moreover, Shahjalal et al. [140] used an AI-based Binary GWO for deploying Virtual Network Functions within the hybrid cloud of the 5 G Internet. The primary problem addressed by their research was optimizing the deployment of these functions to achieve a dual objective: reducing operational costs and minimizing latency, which is critical for maintaining high QoE in 5 G networks. This scheme reduced costs and improved QoE, targeting cost reduction and minimizing latency. Mahenge et al. [141] addressed balancing energy consumption with system capacity and response times in computational offloading scenarios.

They proposed a hybrid method combining PSO with GWO for this aim. This method significantly outperformed conventional methods in enhancing energy efficiency through optimal resource distribution. Finaly, Bi et al. [142] proposed the Lévy flight and SA-based GWO for computation offloading using small base stations. This two-stage optimization algorithm optimally selects edge devices and coordinates task scheduling, reducing latency and costs effectively.

GWO is utilized across diverse computational areas such as the IoV, IoT, 5 G networks, and SMDs, primarily for its effectiveness in reducing system delays, optimizing energy consumption, and improving task scheduling and resource allocation. For example, the proposed algorithm in [134] excels in optimizing latency, energy, and cost, marked by significant performance improvements and adaptability in edge computing compared to other schemes. Moreover, the method in [138] is notable for its enhancements in scheduling, power reduction, and resource allocation. The proposed model in [141] minimize energy consumption and meet capacity and delay requirements efficiently, making it suitable for edge computing compared to other proposed schemes. Distinguished by its simulation of the social hierarchy and hunting techniques of grey wolves, GWO excels in tackling complex multi-dimensional optimization problems. It stands out from other optimization methods like the WOA and SA due to its unique nature-inspired approach. GWO is celebrated for enhancing system performance by efficiently managing latency, energy, and cost, especially in IoT environments that have driven significant performance improvements. Like WOA and SA, GWO faces challenges in scalability and complexity, particularly in dynamic or diverse settings where specific conditions or precise task sequencing are necessary. Moreover, security remains a consistent limitation in all these optimization algorithms, impacting their broader applicability and effectiveness in sensitive environments. Furthermore, while GWO shares a similar level of complexity and adaptability to WOA, making both suitable for various computing paradigms such as fog, edge, and cloud computing, it often exhibits a lower computational overhead than SA, enhancing its desirability in scenarios where this is a critical factor. However, GWO and WOA sometimes struggle with real-world application complexities, much like SA's challenges with computational overhead and integration into existing systems. Despite these issues, GWO's unique mechanism, based on natural wolf behavior, makes it an effective tool for optimizing complex systems. It is a favorable option in environments demanding efficient and effective resource and task optimization.

4.2.7. CS-based, Bee, and Bat-based schemes

In this section, we reviewed studies that have utilized CS, Bee, and Bat algorithms to optimize offloading in various applications. For instance, Keshavarznejad et al. [143] tackled the processing limitations of mobile devices in cloud environments by utilizing fog computing to manage delay-sensitive tasks. They balanced processing between local devices and fog nodes by framing task offloading as a multi-objective optimization problem. This was effectively solved using the NSGA-II and Bees algorithms, achieving notable improvements in energy consumption and delay reduction. Moreover, Alqarni et al. [144] introduced an offloading decision-making framework utilizing the Binary CS Algorithm to optimize offloading strategies to reduce completion times, energy, and costs in MEC settings. Song et al. [145] addressed the challenge of optimizing computation offloading in MEC, where the dual objectives of reducing average latency and minimizing energy consumption are critical. This issue is particularly significant in scenarios where rapid and efficient data processing is required close to user devices while also conserving energy to prolong device battery life and reduce operational costs. They introduced a discrete multi-objective CS algorithm for MEC computation offloading (DMOCS-CO). Their algorithm reached an approximate Pareto optimal set and surpassed other established algorithms in optimizing these objectives during simulation studies. In other study, Souza et al. [146] focused on the challenge of efficiently managing high computational loads and minimizing latency in dynamic vehicular networks, where vehicles rapidly move in and out of network ranges, causing frequent changes in network topology and resource availability. Their bee colony-based task offloading algorithm aimed to optimize resource utilization and reduce delays in vehicular edge computing systems, crucial for latency-sensitive applications like autonomous driving and augmented reality. Furthermore, Xu et al. [147] proposed an MEC architecture that includes a single unmanned helicopter acting as an MEC server and multiple reconnaissance UAVs. They addressed the challenges of excessive delay and high energy consumption in multi-UAV operations with an improved bat algorithm (IBA). This algorithm avoids early convergence and provides greater accuracy and stability than traditional heuristic approaches, optimizing the trade-off between energy use and operational delay. Hong et al. [148] proposed a group mapping-based CS metaheuristic for an MEC system enhanced with data processing units. Their approach focused on minimizing the computational load on CPUs by efficiently offloading decryption tasks through a combinatorial optimization model. The GMCS method improved solution diversity and search effectiveness, demonstrating significant energy savings in various test cases compared to standard algorithms.

These algorithms are good in optimizing crucial tasks, including offloading, resource allocation, and enhancing network performance by reducing latency, energy usage, and operational costs. For instance, integrating NSGA-II with BA specifically aims to reduce power consumption and delays in mobile devices. At the same time, the ODM-BCSA enhances resource allocation efficiency, though it may inadvertently increase delays due to network inconsistencies. Among the other reviewed studies in this section, the proposed algorithm in [144] (ODM-BCSA) excels by minimizing time, energy, and costs while improving resource allocation, with high scalability and marked as energy efficient, cost-effective, latency-aware, and adaptable, particularly effective in edge computing environments. The other efficient model has been proposed in [143] that is notable for reducing power and delay, optimizing offloading trade-offs, and standing out for its energy efficiency, latency awareness, and adaptability in fog and edge environments despite its high complexity. Finally, [146] BC in VANET reduces execution time and enhances reliability in vehicular networks, benefiting from high scalability, energy efficiency, latency awareness, and adaptability, making it especially suitable for edge computing settings. These techniques, however, face unique challenges; for example, Bee and Bat Algorithms can struggle in dynamic environments like VANET and UAV systems due to rapid changes and potential disconnections that may compromise their effectiveness. Additionally, the complexity of these setups, especially in specialized UAV configurations with the IBA, poses substantial deployment challenges. Despite being designed for adaptability and energy efficiency, these schemes often contend with scalability and complexity, particularly in dynamic or resource-limited settings. For instance, the GMCS aims to improve task offloading and CPU performance in mobile devices but is hindered by the complexity of its implementation.

4.2.8. HHO-based and MFO-based schemes

We will explore various studies in this section that have implemented HHO, and MFO to optimize offloading in different applications. In their research study, Priya et al. [149] Wu et al. tackled the challenge of offloading diverse mobile app tasks to MEC hosts by introducing a deep belief network optimized by an Adaptive HHO algorithm. This approach, incorporating Gaussian mutation and CS, effectively balanced exploitation and exploration. Similarly, Wu et al. [150] introduced PECCO, an optimization model using the advanced MFO, PECCO-MFI, to handle the increasing data demands in IoT. This model considered task heterogeneity and load balancing. In another study, Zhang et al. [151] tackled the issue of safeguarding user privacy in MEC environments, where sensitive data is processed closer to users, heightening the risk of data breaches. They developed a privacy-aware computing offloading strategy using privacy entropy as a metric, enhanced by a HHO algorithm with a Gaussian-Cauchy operator. This method effectively obscured user data and minimized resource use while addressing privacy risks in MEC. Li et al. [152] aimed to tackle the problem of high central cloud loads and significant transmission delays in MEC environments. By developing a hybrid HS-HHO algorithm for MEC deployment near User Equipment (UE), they sought to optimize resource allocation and enhance the overall efficiency of edge-cloud collaborative scenarios, ultimately improving system responsiveness and reducing energy consumption. Hijjawi et al. [153] focused on improving resource allocation in wireless networks, which often struggle with the efficient distribution of limited network resources across many users and devices. Their hybrid Prairie Dog Optimization (PDO) algorithm, enhanced with Harris Hawks Optimization (HHO) techniques, was designed to address the challenge of achieving faster convergence and more effective optimization in these complex network environments.

Moreover, Nematollahi et al. [154] introduced an architecture that combined MFO with Opposition-based Learning (OBLMFO) for efficient resource distribution and offloading in a system incorporating sensors, controllers, and FC servers. Their approach resulted in significant reductions in delay and energy consumption. Singh et al. [155] advocated for fog computing to address the high costs and latency of cloud computing. They developed an Energy-efficient Task Offloading Strategy (EETOS) using the Levy-flight MFO algorithm, which significantly reduced energy consumption.

Finaly, Jazaeri et al. [156] focused on IoT content caching in edge networks with SDN processing capabilities. They implemented MFO algorithms for intelligent clustering and caching, resulting in substantial improvements in energy usage, response times, and cache-hit rates in SDN-enabled IoT environments.

Both HHO and MFO provide significant benefits in optimizing resource allocation and improving QoS. However, they face challenges related to complexity, scalability, and practical implementation. HHO schemes, for instance, are effective in tasks like enhancing load balancing and reducing energy use, but they require complex tuning and sophisticated algorithmic handling, which complicates their practical application. These methods can face issues like privacy quantification and scalability challenges when integrated with other advanced techniques. Similarly, MFO schemes focus on boosting caching efficiency and minimizing latency within IoT environments. Despite their adaptability in reducing energy consumption and end-to-end delays, MFO struggles with high computational complexity and the need for precise balance between fog and cloud resources, which can lead to queuing delays. Both strategies also need to overcome the absence of guaranteed optimal performance and challenges in achieving global coordination, especially when dependent on SDN controllers within IoT settings. Additionally, their scalability issues and the complexity of system integration necessitate ongoing enhancements. To conclude, based on the studied papers, the proposed model in [150] excels in enhancing edge-cloud offloading while considering both profit and cost, supported by high energy efficiency and adaptability in fog and edge computing environments. The other optimized method proposed in [151] enhances privacy, minimizes resource use, and reduces privacy risks, making it effective in edge computing with attributes like security and energy efficiency. Moreover, in [154] the authors work on an algorithm called OBLMFO in IoT significantly reduces delay and energy consumption while enhancing resource allocation, performing effectively in fog and edge settings with high energy efficiency and latency awareness. Lastly, [155] LMFO, EETOS in IoT is notable for reducing energy consumption and minimizing end-to-end delay, excelling in fog and edge computing due to its cost-effectiveness, latency awareness, and adaptability.

4.2.9. *Other metaheuristic-based schemes*

This section evaluates a series of Metaheuristic-based schemes documented in the literature, including those found in references [157–175]. Moreover, there are some new metaheuristic methods introduced to address complex optimization problems in cloud computing environments. The Server Residual Efficiency-aware Particle Swarm Optimization (SR-PSO) algorithm enhances virtual machine scheduling by tuning classical PSO operators to manage dynamic scheduling intervals, focusing on optimal energy efficiency and minimal virtual machine migrations. Another innovative approach is the Artificial Immune System based Virtual Machine Scheduling using Modified Clonal Selection Algorithm (VMS-MCSA), which adapts the classical Clonal Selection Algorithm to the dynamic virtual machine scheduling context. This method incorporates a randomized mutation operator to adapt to workload variability with efficient virtual machine consolidation. Both methods aim to improve energy efficiency and operational effectiveness in cloud data centers. In this section we have studied some of the metaheuristic-based schemes [176,177]. To start with, Almadhor et al. [178] introduced a green offloading strategy utilizing a hybrid meta-heuristic algorithm that combines the African Wild Dog Algorithm with cellular learning automata. They tackled the problem of excessive energy consumption, high operational costs, and delays in data processing across networked systems. This method significantly outperformed existing algorithms in simulations, reducing energy consumption, delay time, and cost. Swaminathan et al. [179] focused on the problem of optimizing server resources such as capacity, bandwidth, and power consumption in scenarios where maintaining real-time monitoring and data synchronization is critical. The authors created a model that incorporates Digital Twins and real-time monitoring, optimized by metaheuristic algorithms including the Offloading with Digital Twins and Raindrop Algorithm (ODTRA) and the Probabilistic Recursive Local (PRL) search algorithm. These algorithms optimized server capacity, bandwidth, and power consumption, demonstrating enhanced offloading efficiency in various simulations. In their publication, Kumari et al. [180] developed a metaheuristic scheme using a modified Discrete Jaya Algorithm (DJA) to enhance offloading speed and resource efficiency, modeling it as a multi-objective optimization problem. Their simulations, validated by the Friedman test, showed this method outperforming existing algorithms. Moreover, AlShathri et al. [181] explored integrating IoT with fog computing to address latency issues. They enhanced scalability using population-based meta-heuristics within a novel parallel multi-threading framework, which improved real-time task processing in balancing delay and energy consumption. Hosny et al. [182] addressed the complexity of managing interdependent tasks in multi-access edge computing environments, where coordinating task execution can impact system performance due to dependencies among tasks. They proposed an enhanced Gorilla Troops Algorithm (IGTA) for offloading tasks in multi-access edge computing environments. By introducing task dependencies and adapting a customized crossover operation, IGTA significantly reduced latency, energy consumption, and cost, demonstrating superior efficiency. Furthermore, Babar et al. [183] introduced a nature-inspired multi-objective bacterial foraging optimization (MO-BFO) algorithm to improve edge computing for low-power IoT sensors. Compared to ACO, PSO, and round-robin scheduler, MO-BFO effectively reduced response time and communication costs while managing load on edge servers. In other study, Chen et al. [184] tackled reducing energy consumption in UAVs equipped with MEC servers by developing a hybrid nature-inspired optimization algorithm (HNIO). Enhanced by mutation and diversity detection mechanisms, HNIO outperformed existing algorithms in precision and stability. Moreover, Nematollahi et al. [185] introduced an enhanced multi-objective Aquila optimizer (IMOAO) with a Pareto front to minimize response times and failure rates in fog computing. By incorporating opposition-based learning, the IMOAO improved solution diversity and demonstrated significant reductions in response times and failure rates. Also, Maashi et al. [186] tackled the challenge of optimizing task scheduling to handle the fluctuating demands and energy constraints typical in Next-Generation IoT Fog-Cloud Networks. Their approach, using the Metaheuristic Mountain Gazelle Optimization Algorithm (MMGOA-TSA), was designed to efficiently manage the high variability in workload and network conditions, aiming to balance the trade-off between response time and energy consumption for enhanced system performance. Shen et al. [187] introduced a computational offloading decision method (IBES) for vehicular networks, focusing on reducing expenses related to delay and energy consumption. The model integrates computing across local vehicles, MEC servers, and cloud computing, enhancing the bald eagle search algorithm with various techniques to address task priorities and system costs effectively. Finaly, Alseid et al. [188] addressed the issue of optimizing task offloading in dense and highly mobile vehicular environments, where traditional single-hop offloading methods struggle with efficiency due to the fast-moving and constantly changing network conditions typical in IoV systems. Their dual-phase MSSAMTO-IoV model, utilizing a multi-hop task offloading strategy enhanced by a sparrow search algorithm, aimed to improve the management of network resources and reduce offloading latency, ensuring faster and more reliable service delivery in MEC frameworks integrated with IoV.

Exploring a variety of computational offloading schemes across different technological platforms reveals a sophisticated array of strategies, each designed to enhance certain aspects of system performance. For example, AWDA focuses on reducing energy, cost, and delays in mobile devices but faces challenges like complex algorithm integration and heavy reliance on precise modeling, which can limit scalability and practical deployment. Similarly, ODTRA and PRL in IIoT aim to minimize task execution times and enhance performance, yet they depend on complex digital twin technologies that may not be feasible in all scenarios. In IoT applications, DJA, OFDM, and GTA optimize offloading delays and resource utilization but may struggle with complex system architectures and communication requirements, potentially limiting effectiveness in all MEC scenarios. Moreover, schemes like MO-BFO face QoS challenges and scalability issues, despite their ability to reduce response times and communication costs. In more dynamic environments such as UAVs and IoV, algorithms like HNIO optimize UAV base station usage but are not suitable without these stations, while MSSAMTO-IoV enhances vehicle services but faces limitations due to coverage and implementation complexities. Among the other proposed schemes, [187] IBES and [182] GTA stand out due to their holistic improvements in cost, energy, and latency, alongside their adaptability and environmental fit, making them particularly valuable for dynamic and resource-sensitive applications like VANET and IoT. [181] OFDM, [185] IMOAO, OBL, and [184] HNIO also provide significant contributions to IoT and UAV systems, focusing on optimizing system operations while ensuring efficiency and responsiveness, crucial for real-time data processing and communication.

5. Discussion

In this section, we discuss and provide answers to the research questions based on the analysis of the articles we have reviewed.

Answering RQ1: Among the metaheuristic algorithms reviewed, several stand out for their scalability and adaptability in offloading tasks across fog, edge, and cloud computing environments. The GA is notable for its effectiveness in optimizing server deployment and task completion, particularly in complex IoT environments. PSO excels in reducing latency and enhancing resource usage, making it suitable for IoT and IIoT applications. ACO effectively balances loads and reduces response times, optimizing resource allocation in fog and cloud layers. SA demonstrates robust performance in system optimization and energy efficiency, applicable across various computing frameworks.

Answering RQ2: The use of metaheuristic algorithms in computational offloading introduces various complexities and security risks, as highlighted by numerous studies. Complexity arises primarily from the need to balance optimality and performance in real-time environments, which can challenge the scalability and feasibility of these algorithms. Specifically, algorithms like GA, GWO, WOA, and PSO often require fine-tuning of parameters and may struggle in dynamic network conditions due to their inherent reliance on iterative improvement processes. Security risks are another critical concern, as metaheuristic algorithms can be susceptible to attacks that exploit their operational mechanics. For instance, iterative methods used in these algorithms can expose data to interception or manipulation if not adequately protected. Furthermore, the decentralized nature of the environments where these algorithms typically operate, such as fog and edge computing, increases the vulnerability to unauthorized access and data breaches. This necessitates the integration of robust security protocols to safeguard the data during the offloading process, ensuring the confidentiality and integrity of information across networks.

Fig. 8 illustrates the distributions of complexity, scalability, and implementation feasibility for the systems studied. Most notably, complexity in the analyzed schemes predominantly falls within the high category, reflecting the significant challenges involved in computational offloading. Scalability is mainly categorized as medium, suggesting that while these systems are somewhat flexible, there are still substantial improvements needed to fully scale these solutions. Implementation feasibility primarily ranks medium, indicating that while feasible, these offloading systems often face practical constraints that could limit broader application and deployment.

Answering RQ3: Most of the studied schemes do not focus on cost-effectiveness, with approximately 85 % not sufficiently addressing this aspect. However, among the few that do, WOA and GWO have proven to be more cost-effective. These algorithms stand out for their ability to optimize resource allocation and operational efficiency, which in turn helps reduce overall costs.

Answering RQ4: Nearly all the studied metaheuristic algorithms contribute to improved latency and quality of service in offloading tasks across different systems. Specifically, algorithms like GA, PSO, GWO, and WOA are particularly effective in enhancing these attributes. These algorithms are designed to optimize decision-making processes and resource allocation, which are critical for minimizing latency and maximizing the efficiency of service delivery in computational offloading scenarios.

As shown in Fig. 9, security remains a significant concern in metaheuristic algorithms for computational offloading, with only three studies directly addressing this critical issue. The bar graph highlights the distribution of various attributes, such as energy efficiency, cost-effectiveness, latency awareness, and adaptability across the reviewed schemes. Despite the high attention to latency awareness and adaptability, the minimal focus on security underscores a critical gap in the current research. This gap suggests the need for more dedicated efforts to enhance the security measures within these systems, ensuring the integrity and privacy of data in increasingly complex computing environments. The figure further illustrates how these attributes are prioritized differently across studies, pointing to a potential misalignment between the perceived importance of security and its actual coverage in research.

Answering RQ5: Based on the analysis in Fig. 11 and the details from the studied schemes, the feasibility of implementing metaheuristic algorithms for computational offloading in diverse technological environments shows varied levels. Implementation feasibility is concentrated mainly in the medium category, which accounts for 84 % of the schemes evaluated. This suggests that while these algorithms can be adapted across different settings, the integration process requires careful

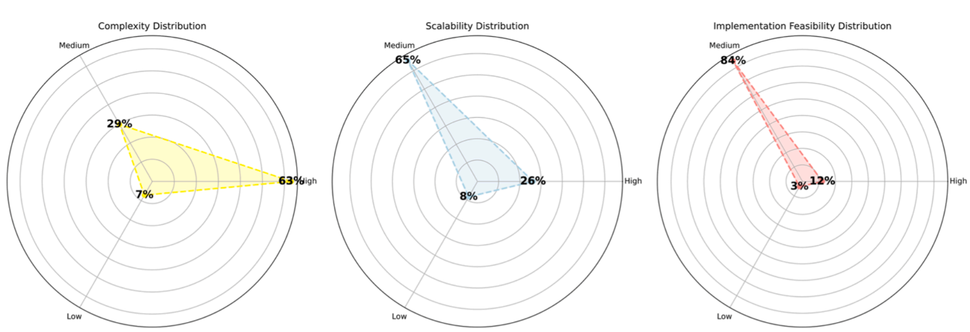

Fig. 8. Distributions of complexity in studied schemes.

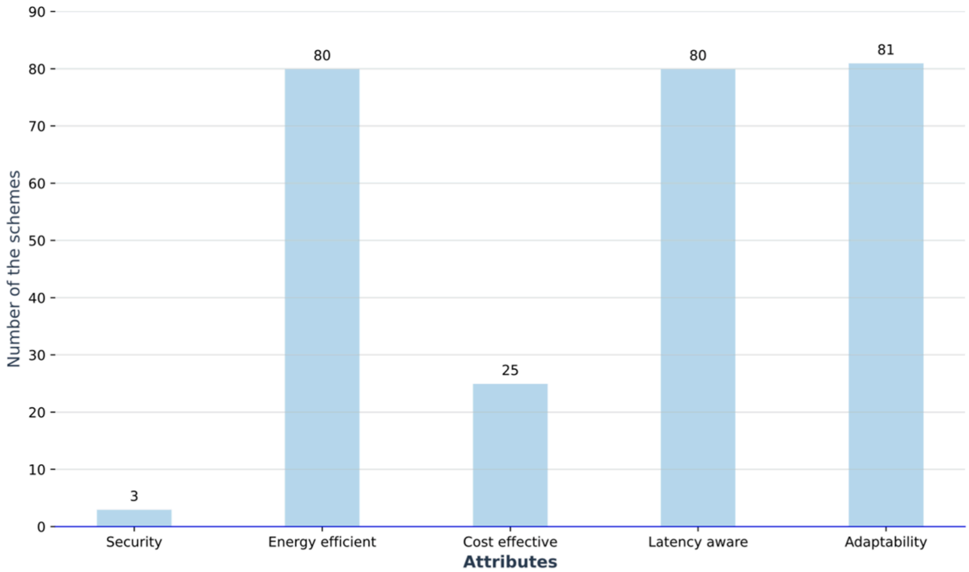

Fig. 9. System attributes of studied schemes.

planning and optimization to address specific environment needs and constraints. Therefore, while feasible, the successful deployment of these algorithms depends heavily on the particular requirements and characteristics of the technological environment in question.

Fig. 10 displays the distribution of computing technologies in metaheuristic algorithm applications across fog, edge, and cloud environments, as quantified by the number of instances each is used: fog computing, edge computing, and cloud computing. This graph highlights the substantial reliance on edge computing for its ability to reduce latency, which is crucial for real-time processing tasks. In contrast, though less frequent, cloud computing's role is significant due to its vast storage and powerful computational capabilities, which make it ideal for handling extensive data-intensive applications. At the same time, fog computing is the least utilized intermediary layer that enhances data accessibility and reduces latency by processing data closer to the end users. It shows the emerging importance in bridging the gap between edge and cloud computing in network architectures.

As shown in Fig. 11, the distribution of metaheuristic algorithm applications across different technologies is presented. The IoT dominates with 43 applications, underscoring its importance and the intense focus on improving connectivity and automation. The IoV and MDs follow, indicating significant interest in enhancing mobile connectivity and performance. Applications in UAVs and vehicular technology are also notable, though less prominent. The graph highlights minimal application in emerging areas like 6 G, AR, and LEO satellites, suggesting potential growth areas for future research in metaheuristics within these technologies.

6. Current challenges in the field and future directions

Metaheuristic algorithms have been widely recognized for their effectiveness in addressing computational offloading challenges across diverse technological environments. However, despite their potential, several significant challenges limit their broader application and efficacy. These challenges are:

Complexity and Scalability: The complexity of metaheuristic

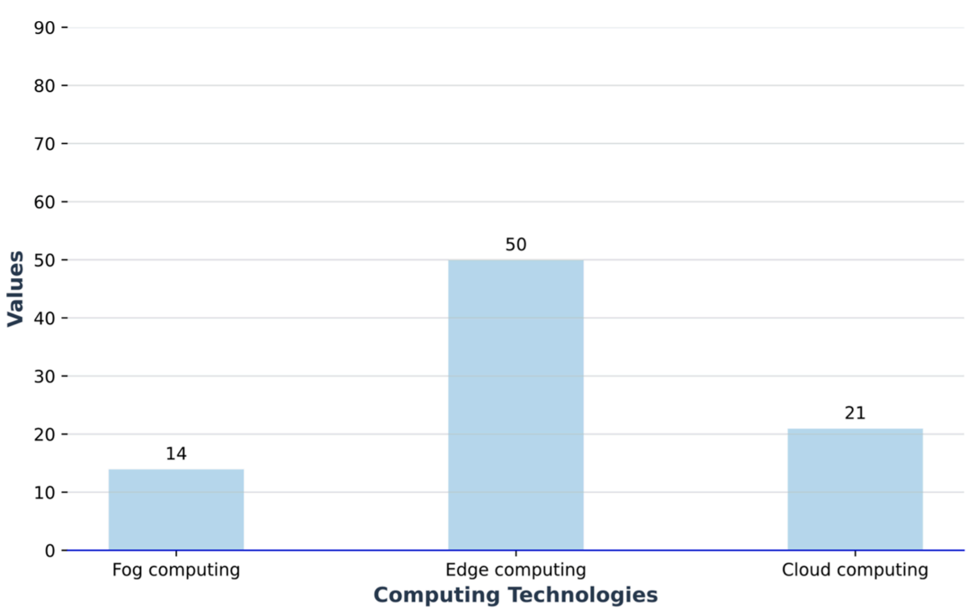

Fig. 10. Distribution of computing technologies in studied schemes.

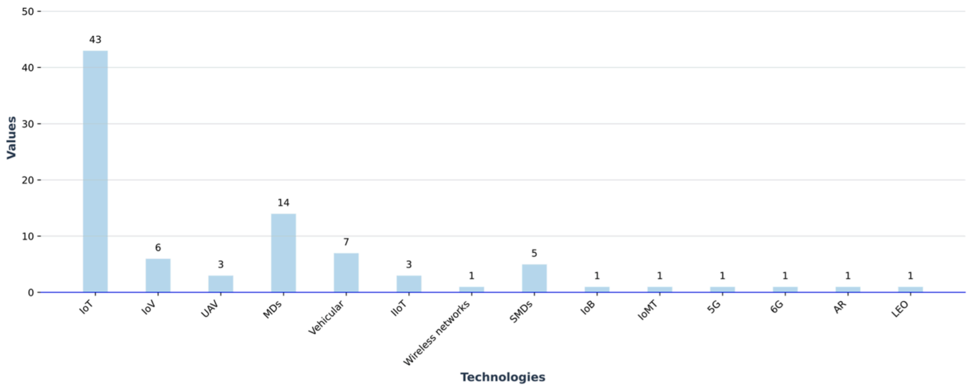

Fig. 11. Number of the applied applications in studied schemes.

algorithms can be a significant barrier, especially in environments with dynamic and unpredictable behaviors. Scalability remains a pressing issue as these algorithms often struggle to perform efficiently as network size and complexity increase. The balance between depth of search (exploration) and the speed of convergence (exploitation) requires careful tuning, which can be challenging to achieve universally across different contexts.

Security and Privacy Concerns: Security is a major challenge in deploying these algorithms for computational offloading, particularly in decentralized environments like fog and edge computing. These algorithms often share data across nodes, exposing sensitive information to potential security breaches. Implementing robust security measures that do not overly degrade performance is crucial and complex.

Real-Time Implementation: Many metaheuristic algorithms are not designed to handle real-time data processing efficiently, which is essential for applications requiring immediate computational offloading decisions, such as those in autonomous driving or real-time health monitoring.

Energy Efficiency and Cost-Effectiveness: Although some metaheuristic algorithms are designed to be energy-efficient, there is often a trade-off between optimization performance and energy consumption. Additionally, cost-effectiveness is not always a primary focus of algorithm design, which can limit their practical application in cost-sensitive environments.

Integration with Existing Systems: Integrating metaheuristic algorithms into existing systems can be challenging due to compatibility issues with legacy systems and the need for substantial customization to address specific operational requirements.

Algorithm Convergence and Stability: Ensuring stable and consistent performance across different deployments and conditions can be challenging. Metaheuristic algorithms may converge prematurely or get stuck in local optima, leading to suboptimal offloading decisions.

Adaptation to New Technologies: As new technologies emerge, such as 5 G/6 G communications, IoT, and smart devices, metaheuristic algorithms need continuous updates to handle new computational loads and network architectures effectively.

Here are the future directions that are important to explore more:

Hybrid metaheuristic models for real-time offloading: Future research should focus on hybridizing different algorithms to create models that can adapt to changing network conditions in real time, thereby reducing latency and improving the responsiveness of mobile and IoT applications.

Metaheuristics for offloading in heterogeneous networks: Metaheuristic-based offloading strategies tailored explicitly for heterogeneous networks, where different types of networks (like 5 G, Wi-Fi, and satellite) coexist are needed. This would address compatibility and performance optimization across varied network standards.

QoS-aware offloading using metaheuristics: Future studies should explore the use of metaheuristics to optimize the energy consumption of both the edge devices and network infrastructure during data offloading, helping to extend battery life and reduce environmental impact.

Scalable metaheuristic approaches for ultra-dense environments: Research should aim to develop scalable algorithms that efficiently manage resource allocation and task offloading in ultra-dense urban environments, where the number of devices and data demand are exceptionally high.

Secure metaheuristic offloading mechanisms: There is a gap in secure offloading strategies that incorporate metaheuristics to safeguard against data breaches and ensure privacy, especially in sensitive applications like health monitoring and personal data management.

Blockchain-enhanced metaheuristic offloading: Integrating blockchain technology with metaheuristic algorithms to enhance security and privacy in offloading processes. This approach can leverage blockchain's decentralized and tamper-resistant ledger to manage and secure offloading decisions, ensuring transparency and trust among multiple stakeholders, such as mobile users and edge service providers.

Privacy-preserving metaheuristic offloading: Developing privacy-preserving offloading frameworks that utilize metaheuristic algorithms to optimize computational tasks without compromising user privacy. Techniques like differential privacy could be embedded within metaheuristic strategies to anonymize sensitive data before offloading, ensuring that user data remains confidential while benefiting from edge computing environments' efficiency.

Metaheuristics for offloading decisions under uncertainty: Developing offloading algorithms that effectively handle user demand and network conditions uncertainties, ensuring reliable and efficient service delivery despite fluctuating environments.

Context-aware offloading using metaheuristics: Future research should investigate context-aware offloading strategies in which metaheuristic algorithms consider the user's context (such as location, device capability, and current network conditions) to optimize offloading decisions.

Quantum metaheuristics for offloading: Investigating the potential of quantum computing to enhance metaheuristic algorithms for offloading. This could lead to breakthroughs in processing speeds and capabilities, solving complex offloading problems faster than traditional methods.

Multi-objective metaheuristic offloading algorithms: Developing multi-objective metaheuristic algorithms that simultaneously optimize several performance metrics, such as cost, latency, energy, and accuracy, to provide a balanced approach to offloading.

Metaheuristics for offloading in autonomous vehicles: Tailoring metaheuristic algorithms to manage data offloading in autonomous vehicles, focusing on minimizing latency and maximizing reliability in vehicle-to-vehicle and vehicle-to-infrastructure communication.

Metaheuristic algorithms for edge AI offloading: Creating metaheuristic algorithms that optimize where AI tasks should be processed—on the device, on the edge, or in the cloud—to effectively balance speed and resource use.

Adaptive learning in metaheuristic offloading: Research should incorporate machine learning techniques with metaheuristics to continually improve offloading strategies based on ongoing system performance data, leading to self-optimizing networks.

7. Conclusion and future directions

This systematic review analyzed computational offloading strategies based on metaheuristics across different computing frameworks. We noted significant gains in efficiency and reliability for IoT, mobile, and vehicular network applications. However, the review has several limitations. It does not detail the types of datasets or simulation tools used in the studies, which limits our ability to judge how reproducible or scalable the findings are. Also, it lacks a thorough examination of the challenges these algorithms might encounter in real-world settings. The review does not compare these metaheuristic methods to traditional algorithms, investigate how changing network conditions affect algorithm performance, or evaluate energy use comprehensively across studies. Future research should work to fill these gaps by using a wider variety of datasets, explaining simulation techniques more clearly, and including assessments of newer network technologies and more complex computing environments.

CRediT authorship contribution statement

Shakiba Rajabi: Writing – original draft, Methodology, Conceptualization. **Komeil Moghaddasi:** Writing – original draft, Methodology, Conceptualization. **Farhad Soleimanian Gharehchopogh:** Writing – original draft, Methodology, Conceptualization. **Mehdi Hosseinzadeh:** Writing – review & editing, Project administration. **Parisa Khoshvaght:** Writing – review & editing, Data curation. **Amir Haider:** Writing – review & editing, Validation, Resources. **Amir Masoud Rahmani:** Writing – review & editing, Resources.

Declaration of Competing 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.

Data availability

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

References

[1] M. Soori, B. Arezoo, R. Dastres, Internet of things for smart factories in industry 4.0, a review, Internet Things Cyber-Phys. Syst. (2023).
[2] S. Nižetić, P. Šolić, D. López-de-Ipiña González-de-Artaza, L. Patrono, Internet of Things (IoT): opportunities, issues and challenges towards a smart and sustainable future, /11/20/ 2020, J. Clean. Prod. 274 (2020) 122877, https://doi.org/10.1016/j.jclepro.2020.122877.
[3] J. Li, J. Dai, A. Issakhov, S.F. Almojil, A. Souri, Towards decision support systems for energy management in the smart industry and Internet of Things, /11/01/ 2021, Comput. Ind. Eng. 161 (2021) 107671, https://doi.org/10.1016/j.cie.2021.107671.
[4] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, D.O. Wu, Edge computing in industrial internet of things: architecture, advances and challenges, IEEE Commun. Surv. Tutor. 22 (4) (2020) 2462–2488, https://doi.org/10.1109/COMST.2020.3009103.
[5] M.R. Rezaee, N.A.W.A. Hamid, M. Hussin, Z.A. Zukarnain, Fog offloading and task management in IoT-fog-cloud environment: review of algorithms, networks, and SDN application, IEEE Access 12 (2024) 39058–39080, https://doi.org/10.1109/ACCESS.2024.3375368.
[6] T. Allaoui, K. Gasmi, T. Ezzedine, Reinforcement learning based task offloading of IoT applications in fog computing: algorithms and optimization techniques, /05/17 2024, Clust. Comput. (2024), https://doi.org/10.1007/s10586-024-04518-z.
[7] N.M. Quy, L.A. Ngoc, N.T. Ban, N.V. Hau, V.K. Quy, Edge computing for real-time internet of things applications: future internet revolution, /09/01 2023, Wirel. Pers. Commun. 132 (2) (2023) 1423–1452, https://doi.org/10.1007/s11277-023-10669-w.
[8] C. Kai, H. Zhou, Y. Yi, W. Huang, Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability, IEEE Trans. Cogn. Commun. Netw. 7 (2) (2021) 624–634, https://doi.org/10.1109/TCCN.2020.3018159.
[9] A. Rajagopalan, et al., Empowering power distribution: unleashing the synergy of IoT and cloud computing for sustainable and efficient energy systems, /03/01/ 2024, Results Eng. 21 (2024) 101949, https://doi.org/10.1016/j.rineng.2024.101949.
[10] B. Kar, W. Yahya, Y.D. Lin, A. Ali, Offloading using traditional optimization and machine learning in federated cloud–edge–fog systems: a survey, IEEE Commun. Surv. Tutor. 25 (2) (2023) 1199–1226, https://doi.org/10.1109/COMST.2023.3239579.
[11] L. Sun, X. Jiang, H. Ren, Y. Guo, Edge-cloud computing and artificial intelligence in internet of medical things: architecture, technology and application, IEEE Access 8 (2020) 101079–101092, https://doi.org/10.1109/ACCESS.2020.2997831.
[12] J. Singh, P. Singh, S.S. Gill, Fog computing: a taxonomy, systematic review, current trends and research challenges, /11/01/ 2021, J. Parallel Distrib. Comput. 157 (2021) 56–85, https://doi.org/10.1016/j.jpdc.2021.06.005.
[13] Y. Mansouri, M.A. Babar, A review of edge computing: features and resource virtualization, /04/01/ 2021, J. Parallel Distrib. Comput. 150 (2021) 155–183, https://doi.org/10.1016/j.jpdc.2020.12.015.
[14] Z. Zabihi, A.M.E. Moghadam, M.H. Rezvani, Reinforcement learning methods for computation offloading: a systematic review, Article 17, ACM Comput. Surv. 56 (1) (2023), https://doi.org/10.1145/3603703.
[15] S. Taheri-abed, A.M. Eftekhari Moghadam, M.H. Rezvani, Machine learning-based computation offloading in edge and fog: a systematic review, /10/01 2023, Clust. Comput. 26 (5) (2023) 3113–3144, https://doi.org/10.1007/s10586-023-04100-z.
[16] W. Li, J. Ou, Machine scheduling with restricted rejection: an application to task offloading in cloud–edge collaborative computing, /05/01/ 2024, Eur. J. Oper. Res. 314 (3) (2024) 912–919, https://doi.org/10.1016/j.ejor.2023.11.002.
[17] J. Zhang, Y. Zhang, H. Wu, W. Li, An ordered submodularity-based budget-feasible mechanism for opportunistic mobile crowdsensing task allocation and pricing, IEEE Trans. Mob. Comput. 23 (2) (2024) 1278–1294, https://doi.org/10.1109/TMC.2022.3232513.
[18] J. Zhang, X. Zhang, Multi-task allocation in mobile crowd sensing with mobility prediction, IEEE Trans. Mob. Comput. 22 (2) (2023) 1081–1094, https://doi.org/10.1109/TMC.2021.3088291.
[19] J. Zhang, Z. Wang, H. Wu, W. Li, Ordered submodularity-based value maximization of UAV data collection in earthquake areas, IEEE Trans. Netw. Sci. Eng. 11 (5) (2024) 4886–4897, https://doi.org/10.1109/TNSE.2024.3400288.
[20] H. Zamani, M.H. Nadimi-Shahraki, S. Mirjalili, F. Soleimanian Gharehchopogh, D. Oliva, A critical review of moth-flame optimization algorithm and its variants: structural reviewing, performance evaluation, and statistical analysis, Arch. Comput. Methods Eng. (2024) 1–49.
[21] K. Zheng, G. Jiang, X. Liu, K. Chi, X. Yao, J. Liu, DRL-based offloading for computation delay minimization in wireless-powered multi-access edge computing, IEEE Trans. Commun. 71 (3) (2023) 1755–1770, https://doi.org/10.1109/TCOMM.2023.3237854.
[22] S. Zhang, S. Bao, K. Chi, K. Yu, S. Mumtaz, DRL-based computation rate maximization for wireless powered multi-AP edge computing, IEEE Trans. Commun. 72 (2) (2024) 1105–1118, https://doi.org/10.1109/TCOMM.2023.3325905.
[23] H. Min, A.M. Rahmani, P. Ghaderkourehpaz, K. Moghaddasi, M. Hosseinzadeh, A joint optimization of resource allocation management and multi-task offloading in high-mobility vehicular multi-access edge computing networks, /01/01/ 2025, Ad Hoc Netw. 166 (2025) 103656, https://doi.org/10.1016/j.adhoc.2024.103656.
[24] K. Moghaddasi, S. Rajabi, F.S. Gharehchopogh, A. Ghaffari, An advanced deep reinforcement learning algorithm for three-layer D2D-edge-cloud computing architecture for efficient task offloading in the Internet of Things, /09/01/ 2024, Sustain. Comput.: Inform. Syst. 43 (2024) 100992, https://doi.org/10.1016/j.suscom.2024.100992.
[25] K. Moghaddasi and S. Rajabi, Double Deep Q-Learning Networks for Energy-Efficient IoT Task Offloading in D2D MEC Environments, in 2023 7th International Conference on Internet of Things and Applications (IoT), 25-26 Oct. 2023 2023, pp. 1-6, doi: 10.1109/IoT60973.2023.10365356.
[26] M. Nssibi, G. Manita, O. Korbaa, Advances in nature-inspired metaheuristic optimization for feature selection problem: a comprehensive survey, /08/01/ 2023, Comput. Sci. Rev. 49 (2023) 100559, https://doi.org/10.1016/j.cosrev.2023.100559.
[27] N. Khaledian, M. Voelp, S. Azizi, M.H. Shirvani, AI-based & heuristic workflow scheduling in cloud and fog computing: a systematic review, /05/08 2024, Clust. Comput. (2024), https://doi.org/10.1007/s10586-024-04442-2.
[28] B. Kitchenham and S. Charters, Guidelines for performing systematic literature reviews in software engineering, ed: UK, 2007.
[29] S. Katoch, S.S. Chauhan, V. Kumar, A review on genetic algorithm: past, present, and future, /02/01 2021, Multimed. Tools Appl. 80 (5) (2021) 8091–8126, https://doi.org/10.1007/s11042-020-10139-6.
[30] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, 6-9 July 1999, *Proc. 1999 Congr. Evolut. Comput. -CEC99 (Cat. No. 99TH8406)* 3 (1999) 1945–1950, https://doi.org/10.1109/CEC.1999.785511.
[31] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE Comput. Intell. Mag. 1 (4) (2006) 28–39, https://doi.org/10.1109/MCI.2006.329691.
[32] P.J.M. van Laarhoven, E.H.L. Aarts, Simulated annealing, in: P.J.M. van Laarhoven, E.H.L. Aarts (Eds.), Simulated Annealing: Theory and Applications, Springer Netherlands, Dordrecht, 1987, pp. 7–15.

[33] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, /01/01 2013, Eng. Comput. 29 (1) (2013) 17–35, https://doi.org/10.1007/s00366-011-0241-y.
[34] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, /03/01/ 2014, Adv. Eng. Softw. 69 (2014) 46–61, https://doi.org/10.1016/j.advengsoft.2013.12.007.
[35] F.S. Gharehchopogh, H. Gholizadeh, A comprehensive survey: whale optimization algorithm and its applications, /08/01/ 2019, Swarm Evolut. Comput. 48 (2019) 1–24, https://doi.org/10.1016/j.swevo.2019.03.004.
[36] D. Karaboga, B. Akay, A comparative study of Artificial Bee Colony algorithm, /08/01/ 2009, Appl. Math. Comput. 214 (1) (2009) 108–132, https://doi.org/10.1016/j.amc.2009.03.090.
[37] H.M. Alabool, D. Alarabiat, L. Abualigah, A.A. Heidari, Harris hawks optimization: a comprehensive review of recent variants and applications, /08/01 2021, Neural Comput. Appl. 33 (15) (2021) 8939–8980, https://doi.org/10.1007/s00521-021-05720-5.
[38] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: algorithm and applications, /08/01/ 2019, Future Gener. Comput. Syst. 97 (2019) 849–872, https://doi.org/10.1016/j.future.2019.02.028.
[39] S. Mirjalili, Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm, /11/01/ 2015, Knowl. -Based Syst. 89 (2015) 228–249, https://doi.org/10.1016/j.knosys.2015.07.006.
[40] M. Indana Zulfa, R. Hartanto, A. Erna Permanasari, W. Ali, GenACO a multi-objective cached data offloading optimization based on genetic algorithm and ant colony optimization, /09/28 2021, PeerJ Comput. Sci. 7 (2021) e729, https://doi.org/10.7717/peerj-cs.729.
[41] H. Wang, J. Han, S. Cao, and X. Zhang, Computation Offloading Strategy of Multi-satellite Cooperative Tasks Based on Genetic Algorithm in Satellite Edge Computing, in 2021 International Conference on Space-Air-Ground Computing (SAGC), 23-25 Oct. 2021 2021, pp. 22-28, doi: 10.1109/SAGC52752.2021.00011.
[42] A. LAMMARI and M. BESSEDIK, Contribution of Three-Neighborhood Local Search on the Design of a Genetic Algorithm Applied to the Optimization of Computational Offloading in Mec."
[43] S. Aleteemat and M. Shurman, Task Offloading Scheduling in Fog Computing Using Hybrid Genetic Algorithm, in 2023 14th International Conference on Information and Communication Systems (ICICS), 21-23 Nov. 2023 2023, pp. 1-6, doi: 10.1109/ICICS60529.2023.10330532.
[44] S. Balasubramanian, T. Meyyappan, Offloading decision making for workflow applications using an enhanced genetic algorithm in the edge environment, /04/01 2022, J. Phys.: Conf. Ser. 2224 (1) (2022) 012011, https://doi.org/10.1088/1742-6596/2224/1/012011.
[45] S.C. Sinthiya, N.I. Shuvo, R.R. Mahmud, J. Ahmed, Low-cost task offloading scheme for mobile edge cloud and internet cloud using genetic algorithm, Dec. 2022, 2022 4th Int. Conf. Sustain. Technol. Ind. 4. 0 (STI) 17-18 (2022) 1–6, https://doi.org/10.1109/STI56238.2022.10103247.
[46] Z. Shang, C. Zhao, The mobile edge computing task offloading in wireless networks based on improved genetic algorithm, Web Intell. 20 (2022) 269–277, https://doi.org/10.3233/WEB-220019.
[47] H. Li, W. Li, X. Zhang, A Genetic Algorithm for Task Offloading problem in Vehicular Edge Computing, in 2022 China Automation Congress (CAC), 25-27 Nov. 2022 2022, pp. 6242-6247, doi: 10.1109/CAC57257.2022.10054675.
[48] D. Wang, et al., Improved Genetic Algorithm for Computation Offloading in Cloud-Edge-Terminal Collaboration Networks, in: Q. Liu, X. Liu, T. Shen, X. Qiu (Eds.), *The 10th International Conference on Computer Engineering and Networks*, 2021, Springer Singapore, Singapore, 2021, pp. 1493–1499.
[49] T. Li, L. Zhao, C. Mao, N. Lin, Q. He, and X. Wang, A Genetic Algorithm Based Adaptive Offloading Scheme for Domain Generalization in VEC, in 2023 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), 21-24 Dec. 2023 2023, pp. 161-168, doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom59178.2023.00054.
[50] Q. Li, B. Peng, Q. Li, M. Lin, C. Chen, and S. Peng, A Latency-Optimal Task Offloading Scheme Using Genetic Algorithm for DAG Applications in Edge Computing, in 2023 8th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 26-28 April 2023 2023, pp. 344-348, doi: 10.1109/ICCCBDA56900.2023.10154698.
[51] J. Fang, W. Li, Y. Wang, and X. Sun, A Novel Multi-Edge Cooperative Offloading Genetic Algorithm towards Environmental Monitoring System, presented at the Proceedings of the 2022 5th International Conference on Telecommunications and Communication Engineering, <conf-loc> , <city>Chengdu< /city> , <country>China< /country> , < /conf-loc> , 2023. [Online]. Available: https://doi.org/10.1145/3577065.3577088.
[52] L. Zhao, H. Fang, J. Wu, Y. Wan, Y. Zhang, and J. Chen, Computing Task Offloading Based on An Enhanced Genetic Algorithm in Mobile Edge Computing, presented at the Proceedings of the 2023 4th International Conference on Computer Science and Management Technology, <conf-loc> , <city>Xi'an< /city> , <country>China< /country> , < /conf-loc> , 2024. [Online]. Available: https://doi.org/10.1145/3644523.3644545.
[53] W. Huang, Y. Zhou, B. Zhang, L. Yu, and Z. Guo, Improved Genetic Offloading Algorithm Based on Edge Terminal Server of Power Distribution Station Area, in 2022 5th International Conference on Energy, Electrical and Power Engineering (CEEPE), 22-24 April 2022 2022, pp. 590-595, doi: 10.1109/CEEPE55110.2022.9783386.
[54] N. El Menbawy, H.A. Ali, M.S. Saraya, A.M.T. Ali-Eldin, M.M. Abdelsalam, Energy-efficient computation offloading using hybrid GA with PSO in internet of robotic things environment, /11/01 2023, J. Supercomput. 79 (17) (2023) 20076–20115, https://doi.org/10.1007/s11227-023-05387-w.
[55] Z. Wei, J. Pan, Z. Lyu, J. Xu, L. Shi, J. Xu, An offloading strategy with soft time windows in mobile edge computing, /12/01/ 2020, Comput. Commun. 164 (2020) 42–49, https://doi.org/10.1016/j.comcom.2020.09.011.
[56] S. Chakraborty, K. Mazumdar, Sustainable task offloading decision using genetic algorithm in sensor mobile edge computing, /04/01/ 2022, J. King Saud. Univ. - Comput. Inf. Sci. 34 (4) (2022) 1552–1568, https://doi.org/10.1016/j.jksuci.2022.02.014.
[57] E. Rajapackiyam, et al., An efficient computation offloading in edge environment using genetic algorithm with directed search techniques for IoT applications, /09/01/ 2024, Future Gener. Comput. Syst. 158 (2024) 378–390, https://doi.org/10.1016/j.future.2024.04.021.
[58] A. Hussain, S.V. Manikanthan, T. Padmapriya, M. Nagalingam, Genetic algorithm based adaptive offloading for improving IoT device communication efficiency, /05/01 2020, Wirel. Netw. 26 (4) (2020) 2329–2338, https://doi.org/10.1007/s11276-019-02121-4.
[59] Z. Liao, J. Peng, B. Xiong, J. Huang, Adaptive offloading in mobile-edge computing for ultra-dense cellular networks based on genetic algorithm, /02/17 2021, J. Cloud Comput. 10 (1) (2021) 15, https://doi.org/10.1186/s13677-021-00232-y.
[60] Z. Li, Q. Zhu, Genetic Algorithm-Based Optimization of Offloading and Resource Allocation in Mobile-Edge Computing, Information 11 (2) (2020) 83. ⟨https://www.mdpi.com/2078-2489/11/2/83⟩ ([Online]. Available).
[61] H. Liu, Z. Niu, J. Du, X. Lin, Genetic algorithm for delay efficient computation offloading in dispersed computing, /04/01/ 2023, Ad Hoc Netw. 142 (2023) 103109, https://doi.org/10.1016/j.adhoc.2023.103109.
[62] A. Zhu, Y. Wen, Computing Offloading Strategy Using Improved Genetic Algorithm in Mobile Edge Computing System, /08/10 2021, J. Grid Comput. 19 (3) (2021) 38, https://doi.org/10.1007/s10723-021-09578-8.
[63] H. Materwala, L. Ismail, R.M. Shubair, R. Buyya, Energy-SLA-aware genetic algorithm for edge–cloud integrated computation offloading in vehicular networks, /10/01/ 2022, Future Gener. Comput. Syst. 135 (2022) 205–222, https://doi.org/10.1016/j.future.2022.04.009.
[64] H. Materwala, L. Ismail, H.S. Hassanein, QoS-SLA-aware adaptive genetic algorithm for multi-request offloading in integrated edge-cloud computing in Internet of vehicles, /10/01/ 2023, Veh. Commun. 43 (2023) 100654, https://doi.org/10.1016/j.vehcom.2023.100654.
[65] M.S. Zalat, S.M. Darwish, M.M. Madbouly, An Adaptive Offloading Mechanism for Mobile Cloud Computing: A Niching Genetic Algorithm Perspective, IEEE Access 10 (2022) 76752–76765, https://doi.org/10.1109/ACCESS.2022.3192391.
[66] H. Song, B. Gu, K. Son, W. Choi, Joint Optimization of Edge Computing Server Deployment and User Offloading Associations in Wireless Edge Network via a Genetic Algorithm, IEEE Trans. Netw. Sci. Eng. 9 (4) (2022) 2535–2548, https://doi.org/10.1109/TNSE.2022.3165372.
[67] K. Wang, X. Wang, X. Liu, Sustainable internet of vehicles system: a task offloading strategy based on improved genetic algorithm, Sustainability 15 (9) (2023) 7506. ⟨https://www.mdpi.com/2071-1050/15/9/7506⟩ ([Online]. Available).
[68] C. Li, L. Chai, K. Jiang, Y. Zhang, J. Liu, S. Wan, DNN Partition and Offloading Strategy with Improved Particle Swarm Genetic Algorithm in VEC, IEEE Trans. Intell. Veh. (2023) 1–11, https://doi.org/10.1109/TIV.2023.3346506.
[69] H. Jin, Y.-G. Kim, Z. Jin, C. Fan, Y. Xu, Joint Task Offloading Based on Distributed Deep Reinforcement Learning-Based Genetic Optimization Algorithm for Internet of Vehicles, /02/26 2024, J. Grid Comput. 22 (1) (2024) 34, https://doi.org/10.1007/s10723-024-09741-x.
[70] N. Keshari, T.S. Gupta, D. Singh, Particle Swarm Optimization based Task Offloading in Vehicular Edge Computing, 19-21 Dec. 2021, 2021 IEEE 18th India Counc. Int. Conf. (INDICON) (2021) 1–8, https://doi.org/10.1109/INDICON52576.2021.9691758.
[71] S. Li, H. Ge, X. Chen, L. Liu, H. Gong, and R. Tang, Computation Offloading Strategy for Improved Particle Swarm Optimization in Mobile Edge Computing, in 2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 24-26 April 2021 2021, pp. 375-381, doi: 10.1109/ICCCBDA51879.2021.9442609.
[72] J. Bi, K. Zhang, H. Yuan, Q. Hu, Energy-Aware Task Offloading with Genetic Particle Swarm Optimization in Hybrid Edge Computing, 17-20 Oct. 2021, 2021 IEEE Int. Conf. Syst., Man, Cybern. (SMC) (2021) 3194–3199, https://doi.org/10.1109/SMC52423.2021.9658678.
[73] B. Wang, J. Wei, Particle Swarm Optimization with Genetic Evolution for Task Offloading in Device-Edge-Cloud Collaborative Computing, // 2023, in: D.-S. Huang, P. Premaratne, B. Jin, B. Qu, K.-H. Jo, A. Hussain (Eds.), Advanced Intelligent Computing Technology and Applications, Springer Nature Singapore, Singapore, 2023, pp. 340–350. // 2023.
[74] C. Ju, Y. Ma, Z. Yin, F. Zhang, An Request Offloading and Scheduling Approach Base on Particle Swarm Optimization Algorithm in IoT-Fog Networks, June 2021, 2021 13th Int. Conf. Commun. Softw. Netw. (ICCSN) 4-7 (2021) 185–188, https://doi.org/10.1109/ICCSN52437.2021.9463602.
[75] W. Li, W. Deng, R. She, N. Zhang, Y. Wang, and W. Ma, Edge Computing Offloading Strategy Based on Particle Swarm Algorithm for Power Internet of Things, in 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), 26-28 March 2021 2021, pp. 145-150, doi: 10.1109/ICBAIE52039.2021.9389919.
[76] Y. Deng, P. Wang, and L. Li, Task Offloading for Mixed Cloud/Fog Computing in Vehicular Network Using Genetic Particle Swarm Optimization, in 2021 4th

International Conference on Information Communication and Signal Processing (ICICSP), 24-26 Sept. 2021 2021, pp. 489-494, doi: 10.1109/ICICSP54369.2021.9611841.
[77] J. Bi, H. Yuan, S. Duanmu, Energy-efficient Task Offloading Using Hybrid Particle Swarm Optimization with Genetic Operations in Smart Edge, /01/01/ 2020, IFAC-Pap. 53 (5) (2020) 19–24, https://doi.org/10.1016/j.ifacol.2021.04.122.
[78] R. Verma, S. Benedict, AdaptPSOFL: Adaptive Particle Swarm Optimization-Based Layer Offloading Framework for Federated Learning, // 2023, in: S. Shakya, J.M. R.S. Tavares, A. Fernández-Caballero, G. Papakostas (Eds.), Fourth International Conference on Image Processing and Capsule Networks, Springer Nature Singapore, Singapore, 2023, pp. 597–610. // 2023.
[79] X. Yuan, H. Tian, W. Zhang, H. Zhao, Z. Zhao, and N. Zhang, CA-PSO: A Combinatorial Auction and Improved Particle Swarm Optimization based Computation Offloading Approach for E-Healthcare, in ICC 2022 - IEEE International Conference on Communications, 16-20 May 2022 2022, pp. 3850-3855, doi: 10.1109/ICC45855.2022.9838733.
[80] X. Chen and S. Zheng, Resource Allocation and Task Offloading Strategy Base on Hybrid Simulated Annealing-Binary Particle Swarm Optimization in Cloud-Edge Collaborative System, in 2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 16-18 Dec. 2022 2022, vol. 5, pp. 379-383, doi: 10.1109/IMCEC55388.2022.10019879.
[81] J. He and W. Bai, Computation Offloading and Task Scheduling Based on Improved Integer Particle Swarm Optimization in Fog Computing, in 2023 3rd International Conference on Neural Networks, Information and Communication Engineering (NNICE), 24-26 Feb. 2023 2023, pp. 633-638, doi: 10.1109/NNICE58320.2023.10105675.
[82] D. Ye, X. Wang, and J. Hou, An Edge Computing Offloading Algorithm Based on Second-Order Oscillatory Particle Swarm Optimization, in 2022 3rd Information Communication Technologies Conference (ICTC), 6-8 May 2022 2022, pp. 221-226, doi: 10.1109/ICTC55111.2022.9778592.
[83] K. Yuan and S. Chen, Improved Particle Swarm Optimization-Based Computation Offloading and Caching Decision for Internet of Things, in 2022 IEEE/CIC International Conference on Communications in China (ICCC), 11-13 Aug. 2022 2022, pp. 1068-1073, doi: 10.1109/ICCC55456.2022.9880663.
[84] S. Ghosh, P. Kuila, T. Biswas, An Energy Efficient Offloading Technique for UAV-Assisted MEC Using Nature Inspired Algorithm, // 2023, in: V. Bhateja, X.-S. Yang, J. Chun-Wei Lin, R. Das (Eds.), Intelligent Data Engineering and Analytics, Springer Nature Singapore, Singapore, 2023, pp. 309–318. // 2023.
[85] M. Kumar, A. Kishor, P.K. Singh, K. Dubey, Deadline-aware Cost and Energy Efficient Offloading in Mobile Edge Computing, IEEE Trans. Sustain. Comput. (2024) 1–12, https://doi.org/10.1109/TSUSC.2024.3381841.
[86] N. Bacanin, M. Antonijevic, T. Bezdan, M. Zivkovic, K. Venkatachalam, S. Malebary, Energy efficient offloading mechanism using particle swarm optimization in 5G enabled edge nodes, /02/01 2023, Clust. Comput. 26 (1) (2023) 587–598, https://doi.org/10.1007/s10586-022-03609-z.
[87] S. Ma, S. Song, L. Yang, J. Zhao, F. Yang, L. Zhai, Dependent tasks offloading based on particle swarm optimization algorithm in multi-access edge computing, /11/01/ 2021, Appl. Soft Comput. 112 (2021) 107790, https://doi.org/10.1016/j.asoc.2021.107790.
[88] Q. You, B. Tang, Efficient task offloading using particle swarm optimization algorithm in edge computing for industrial internet of things, /07/28 2021, J. Cloud Comput. 10 (1) (2021) 41, https://doi.org/10.1186/s13677-021-00256-4.
[89] L. Chen, D.-G. Zhang, J. Zhang, T. Zhang, W.-J. Wang, Y.-H. Cao, A novel offloading approach of IoT user perception task based on quantum behavior particle swarm optimization, /04/01/ 2023, Future Gener. Comput. Syst. 141 (2023) 577–594, https://doi.org/10.1016/j.future.2022.12.016.
[90] M.A. Alqarni, M.H. Mousa, M.K. Hussein, Task offloading using GPU-based particle swarm optimization for high-performance vehicular edge computing, /11/01/ 2022, J. King Saud. Univ. - Comput. Inf. Sci. 34 (10) (2022) 10356–10364, https://doi.org/10.1016/j.jksuci.2022.10.026.
[91] S. Dong, Y. Xia, J. Kamruzzaman, Quantum Particle Swarm Optimization for Task Offloading in Mobile Edge Computing, IEEE Trans. Ind. Inform. 19 (8) (2023) 9113–9122, https://doi.org/10.1109/TII.2022.3225313.
[92] T. Gao, Q. Tang, J. Li, Y. Zhang, Y. Li, J. Zhang, A Particle Swarm Optimization With Lévy Flight for Service Caching and Task Offloading in Edge-Cloud Computing, IEEE Access 10 (2022) 76636–76647, https://doi.org/10.1109/ACCESS.2022.3192846.
[93] W. Shu, Y. Li, Joint offloading strategy based on quantum particle swarm optimization for MEC-enabled vehicular networks, /02/01/ 2023, Digit. Commun. Netw. 9 (1) (2023) 56–66, https://doi.org/10.1016/j.dcan.2022.03.009.
[94] H. Zhang, S. Jin, Z. Tang, and L. Chang, Joint Offloading with Fixed-Site and UAV-Mounted Edge Servers Based on Particle Swarm Optimization, in 2023 9th International Conference on Control Science and Systems Engineering (ICCSSE), 16-18 June 2023 2023, pp. 190-195, doi: 10.1109/ICCSSE59359.2023.10245119.
[95] Z. Yang, A Two-Stage Intelligent Task Offloading Approach for Cloud-Edge Collaborative Computing Based on Particle Swarm Algorithm, in 2023 5th International Conference on Frontiers Technology of Information and Computer (ICFTIC), 17-19 Nov. 2023 2023, pp. 113-118, doi: 10.1109/ICFTIC59930.2023.10455821.
[96] J. Wang and H. Wang, A Secure Data Offloading Strategy for UAV Wireless Networks Based on Improved Ant Colony Algorithms, in 2022 3rd International Conference on Electronics, Communications and Information Technology (CECIT), 23-25 Dec. 2022 2022, pp. 57-61, doi: 10.1109/CECIT58139.2022.00017.
[97] Y. Wang, J. Fu, and Y. Zhang, Multi-edge Server Load Balancing Offloading Method Based On Ant Colony Algorithm, in 2021 40th Chinese Control Conference (CCC), 26-28 July 2021 2021, pp. 1826-1831, doi: 10.23919/CCC52363.2021.9549527.
[98] J. Chang, C. Liu, M. Yang, Y. Gong, L. Huang, and K. Liu, Cloud Offloading Technique based on an Optimized Ant Colony Algorithm, in 2022 3rd International Conference on Computer Science and Management Technology (ICCSMT), 18-20 Nov. 2022 2022, pp. 317-321, doi: 10.1109/ICCSMT58129.2022.00074.
[99] X. Huang, Y. Yang, and X. Wu, A Meta-Heuristic Computation Offloading Strategy for IoT Applications in an Edge-Cloud Framework, presented at the Proceedings of the 2019 3rd International Symposium on Computer Science and Intelligent Control, Amsterdam, Netherlands, 2020. [Online]. Available: https://doi.org/10.1145/3386164.3390513.
[100] M.K. Hussein, M.H. Mousa, Efficient Task Offloading for IoT-Based Applications in Fog Computing Using Ant Colony Optimization, IEEE Access 8 (2020) 37191–37201, https://doi.org/10.1109/ACCESS.2020.2975741.
[101] F. Xu, Z. Qin, L. Ning, Z. Zhang, Research on computing offloading strategy based on Genetic Ant Colony fusion algorithm, /07/01/ 2022, Simul. Model. Pract. Theory 118 (2022) 102523, https://doi.org/10.1016/j.simpat.2022.102523.
[102] C. Li, Y. Zhang, Y. Luo, DQN-enabled content caching and quantum ant colony-based computation offloading in MEC, /01/01/ 2023, Appl. Soft Comput. 133 (2023) 109900, https://doi.org/10.1016/j.asoc.2022.109900.
[103] M.I. Zulfa, R. Hartanto, A.E. Permanasari, W. Ali, Improving Cached Data Offloading Optimization Based on Enhanced Hybrid Ant Colony Genetic Algorithm, IEEE Access 10 (2022) 84558–84568, https://doi.org/10.1109/ACCESS.2022.3197205.
[104] C.W. Danial, I.A. Saroit, and S.M. Mohamed, Energy Efficient Ant Colony Cloud Offloading Algorithm (EACO), presented at the Proceedings of the 9th International Conference on Software and Information Engineering, Cairo, Egypt, 2021. [Online]. Available: https://doi.org/10.1145/3436829.3436846.
[105] B. Wang, T. Feng, D. Huang, and X. Li, Mobile Edge Computing for LEO Satellite: A Computation Offloading Strategy Based Improved Ant Colony Algorithm, in Proceedings of the 11th International Conference on Computer Engineering and Networks, Singapore, Q. Liu, X. Liu, B. Chen, Y. Zhang, and J. Peng, Eds., 2022// 2022: Springer Nature Singapore, pp. 1664-1676.
[106] A. Kishor, C. Chakarbarty, Task Offloading in Fog Computing for Using Smart Ant Colony Optimization, /11/01 2022, Wirel. Pers. Commun. 127 (2) (2022) 1683–1704, https://doi.org/10.1007/s11277-021-08714-7.
[107] J. Bi, H. Yuan, S. Duanmu, M. Zhou, A. Abusorrah, Energy-Optimized Partial Computation Offloading in Mobile-Edge Computing With Genetic Simulated-Annealing-Based Particle Swarm Optimization, IEEE Internet Things J. 8 (5) (2021) 3774–3785, https://doi.org/10.1109/JIOT.2020.3024223.
[108] T. Huang, S. Li, and X. Gao, Computing Resource Allocation and Offloading Method based on Simulated Annealing Algorithm, in 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 29-30 Oct. 2020 2020, pp. 276-282, doi: 10.1109/CyberC49757.2020.00052.
[109] X. Yuan, et al., A MEC Offloading Strategy Based on Improved DQN and Simulated Annealing for Internet of Behavior, p. Article 28, ACM Trans. Sen. Netw. 19 (2) (2022), https://doi.org/10.1145/3532093.
[110] Y. Li, Optimization of Task Offloading Problem Based on Simulated Annealing Algorithm in MEC, 4-7 June 2021, 2021 9th Int. Conf. Intell. Comput. Wirel. Opt. Commun. (ICWOC) (2021) 47–52, https://doi.org/10.1109/ICWOC52624.2021.9530216.
[111] A. Mahjoubi, K.J. Grinnemo, and J. Taheri, An Efficient Simulated Annealing-based Task Scheduling Technique for Task Offloading in a Mobile Edge Architecture, in 2022 IEEE 11th International Conference on Cloud Networking (CloudNet), 7-10 Nov. 2022 2022, pp. 159-167, doi: 10.1109/CloudNet55617.2022.9978900.
[112] H. Yuan, J. Bi, M. Zhou, J. Zhang, and W. Zhang, Profit-Maximized Task Offloading with Simulated-annealing-based Migrating Birds Optimization in Hybrid Cloud-Edge Systems, in 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 11-14 Oct. 2020 2020, pp. 1218-1223, doi: 10.1109/SMC42975.2020.9283467.
[113] B. Xu, et al., Optimization of cooperative offloading model with cost consideration in mobile edge computing, /06/01 2023, Soft Comput. 27 (12) (2023) 8233–8243, https://doi.org/10.1007/s00500-022-07733-1.
[114] D. Gabi, et al., Dynamic scheduling of heterogeneous resources across mobile edge-cloud continuum using fruit fly-based simulated annealing optimization scheme, /08/01 2022, Neural Comput. Appl. 34 (16) (2022) 14085–14105, https://doi.org/10.1007/s00521-022-07260-y.
[115] H. Yuan, J. Bi, Z. Wang, J. Yang, J. Zhang, Partial and cost-minimized computation offloading in hybrid edge and cloud systems, /09/15/ 2024, Expert Syst. Appl. 250 (2024) 123896, https://doi.org/10.1016/j.eswa.2024.123896.
[116] L.N.T. Huynh, Q.V. Pham, T.D.T. Nguyen, M.D. Hossain, J.H. Park, and E.N. Huh, A Study on Computation Offloading in MEC Systems using Whale Optimization Algorithm, in 2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM), 3-5 Jan. 2020 2020, pp. 1-4, doi: 10.1109/IMCOM48794.2020.9001756.
[117] J. Li, W. Zhang, P. Cheng, Y. Wang, and X. Du, Adaptive Binary Whale Optimization Algorithm for Computation Offloading Optimization in Mobile Edge Computing, in 2022 12th International Conference on Information Science and

Technology (ICIST), 14-16 Oct. 2022 2022, pp. 52-57, doi: 10.1109/ICIST55546.2022.9926934.
[118] H. Shingare, M. Kumar, Whale Optimization-Based Task Offloading Technique in Integrated Cloud-Fog Environment, // 2023, in: M. Thakur, S. Agnihotri, B. S. Rajpurohit, M. Pant, K. Deep, A.K. Nagar (Eds.), Soft Computing for Problem Solving, Springer Nature Singapore, Singapore, 2023, pp. 459–469. // 2023.
[119] J. Bi, W. Gu, H. Yuan, and Z. Yu, Latency-minimized Computation Offloading in Fog Computing with Hybrid Whale optimization, in 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 9-12 Oct. 2022 2022, pp. 504-509, doi: 10.1109/SMC53654.2022.9945284.
[120] S. Lin et al., An Offloading Strategy for Ocean Edge Computing Based on Whale Optimization Algorithm, in 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 19-21 July 2023 2023, pp. 1-6, doi: 10.1109/ICECCME57830.2023.10252368.
[121] X. Zhu, M. Zhou, Multiobjective Optimized Cloudlet Deployment and Task Offloading for Mobile-Edge Computing, IEEE Internet Things J. 8 (20) (2021) 15582–15595, https://doi.org/10.1109/JIOT.2021.3073113.
[122] M. Huang, Q. Zhai, Y. Chen, S. Feng, F. Shu, Multi-Objective Whale Optimization Algorithm for Computation Offloading Optimization in Mobile Edge Computing, Sensors 21 (8) (2021) 2628. ⟨https://www.mdpi.com/1424-8220/21/8/2628⟩ ([Online]. Available).
[123] K.M. Hosny, A.I. Awad, W. Said, M. Elmezain, E.R. Mohamed, M.M. Khashaba, Enhanced whale optimization algorithm for dependent tasks offloading problem in multi-edge cloud computing, /06/01/ 2024, Alex. Eng. J. 97 (2024) 302–318, https://doi.org/10.1016/j.aej.2024.04.038.
[124] K.M. Hosny, A.I. Awad, M.M. Khashaba, M.M. Fouda, M. Guizani, E.R. Mohamed, Optimized Multi-User Dependent Tasks Offloading in Edge-Cloud Computing Using Refined Whale Optimization Algorithm, IEEE Trans. Sustain. Comput. 9 (1) (2024) 14–30, https://doi.org/10.1109/TSUSC.2023.3294447.
[125] V. Anoop, P.R. Bipin, Exploitation whale optimization based optimal offloading approach and topology optimization in a mobile ad hoc cloud environment, /02/01 2022, J. Ambient Intell. Humaniz. Comput. 13 (2) (2022) 1053–1072, https://doi.org/10.1007/s12652-021-02945-z.
[126] J. Li, Q. Wang, S. Hu, L. Li, Hybrid Immune Whale Differential Evolution Optimization (HIWDEO) Based Computation Offloading in MEC for IoT, /11/21 2023, J. Grid Comput. 21 (4) (2023) 70, https://doi.org/10.1007/s10723-023-09705-7.
[127] Z. Liu, J. Wang, Z. Gao, J. Wei, Privacy-preserving edge computing offloading scheme based on whale optimization algorithm, /02/01 2023, J. Supercomput. 79 (3) (2023) 3005–3023, https://doi.org/10.1007/s11227-022-04756-1.
[128] J. Bi, X. Xue, H. Yuan, and J. Zhang, Latency-Minimized Computation Offloading in Vehicle Fog Computing with Improved Whale Optimization Algorithm, in 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 1-4 Oct. 2023 2023, pp. 5003-5008, doi: 10.1109/SMC53992.2023.10394272.
[129] M. Kumar, G.K. Walia, H. Shingare, S. Singh, S.S. Gill, AI-Based Sustainable and Intelligent Offloading Framework for IIoT in Collaborative Cloud-Fog Environments, IEEE Trans. Consum. Electron. 70 (1) (2024) 1414–1422, https://doi.org/10.1109/TCE.2023.3320673.
[130] M. Zhang, A Binary Hybrid Grey Wolf Optimizer for MEC Offloading, in 2022 12th International Conference on Information Technology in Medicine and Education (ITME), 18-20 Nov. 2022 2022, pp. 525-529, doi: 10.1109/ITME56794.2022.00115.
[131] L. Luo, Priority Task Offloading Decision Optimization Based on Multi-Objective Grey Wolf Algorithm in Internet of Things Edge Computing Scenarios, in 2023 IEEE 3rd International Conference on Data Science and Computer Application (ICDSCA), 27-29 Oct. 2023 2023, pp. 1574-1577, doi: 10.1109/ICDSCA59871.2023.10393696.
[132] X. Chen, X. Li, An Energy-Efficient Task Offloading Decision in Electric Power IoT Based on Edge Computing, 23-26 Sept. 2021, 2021 Int. Conf. Electron. Inf. Eng. Comput. Sci. (EIECS) (2021) 597–600, https://doi.org/10.1109/EIECS53707.2021.9588103.
[133] W. Zhang, K. Tuo, Research on Offloading Strategy for Mobile Edge Computing Based on Improved Grey Wolf Optimization Algorithm, Electronics 12 (11) (2023) 2533. ⟨https://www.mdpi.com/2079-9292/12/11/2533⟩ ([Online]. Available).
[134] N. Nujhat, et al., Task offloading exploiting grey wolf optimization in collaborative edge computing, /01/23 2024, J. Cloud Comput. 13 (1) (2024) 23, https://doi.org/10.1186/s13677-023-00570-z.
[135] W. Shang, P. Ke, T. Zhou, Mobile Edge Computing Offloading Problem Based on Improved Grey Wolf Optimizer, // 2023, in: D.-S. Huang, P. Premaratne, B. Jin, B. Qu, K.-H. Jo, A. Hussain (Eds.), Advanced Intelligent Computing Technology and Applications, Springer Nature Singapore, Singapore, 2023, pp. 343–355. // 2023.
[136] Y. Tang, G. Yin, P. Cong, J. Sun, and J. Zhou, A Discrete Grey Wolf Optimizer Metaheuristic for Task Offloading in Multi-Server MEC with Batteryless Devices, in 2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS), 17-21 Dec. 2023 2023, pp. 2235-2242, doi: 10.1109/ICPADS60453.2023.00301.
[137] S. Feng, Y. Chen, Q. Zhai, M. Huang, F. Shu, Optimizing computation offloading strategy in mobile edge computing based on swarm intelligence algorithms, /07/08 2021, EURASIP J. Adv. Signal Process. 2021 (1) (2021) 36, https://doi.org/10.1186/s13634-021-00751-5.
[138] S.M. Hashemi, A. Sahafi, A.M. Rahmani, M. Bohlouli, GWO-SA: Gray Wolf Optimization Algorithm for Service Activation Management in Fog Computing, IEEE Access 10 (2022) 107846–107863, https://doi.org/10.1109/ACCESS.2022.3212439.
[139] Y. Cong, K. Xue, C. Wang, W. Sun, S. Sun, F. Hu, Latency-Energy Joint Optimization for Task Offloading and Resource Allocation in MEC-Assisted Vehicular Networks, IEEE Trans. Veh. Technol. 72 (12) (2023) 16369–16381, https://doi.org/10.1109/TVT.2023.3289236.
[140] M. Shahjalal, N. Farhana, P. Roy, M.A. Razzaque, K. Kaur, M.M. Hassan, A Binary Gray Wolf Optimization algorithm for deployment of Virtual Network Functions in 5G hybrid cloud, /09/01/ 2022, Comput. Commun. 193 (2022) 63–74, https://doi.org/10.1016/j.comcom.2022.06.041.
[141] M.P.J. Mahenge, C. Li, C.A. Sanga, Energy-efficient task offloading strategy in mobile edge computing for resource-intensive mobile applications, /12/01/ 2022, Digit. Commun. Netw. 8 (6) (2022) 1048–1058, https://doi.org/10.1016/j.dcan.2022.04.001.
[142] J. Bi, Z. Wang, H. Yuan, J. Zhang, M. Zhou, Cost-Minimized Computation Offloading and User Association in Hybrid Cloud and Edge Computing, IEEE Internet Things J. 11 (9) (2024) 16672–16683, https://doi.org/10.1109/JIOT.2024.3354348.
[143] M. Keshavarznejad, M.H. Rezvani, S. Adabi, Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms, /09/01 2021, Clust. Comput. 24 (3) (2021) 1825–1853, https://doi.org/10.1007/s10586-020-03230-y.
[144] M. Alqarni, A. Cherif, E. Alkayyal, ODM-BCSA: An Offloading Decision-Making Framework based on Binary Cuckoo Search Algorithm for Mobile Edge Computing, /05/01/ 2023, Comput. Netw. 226 (2023) 109647, https://doi.org/10.1016/j.comnet.2023.109647.
[145] C. Song and H. Zhou, Computation Offloading Optimization in Mobile Edge Computing Based on Multi-Objective Cuckoo Search Algorithm, presented at the Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence, Xiamen, China, 2020. [Online]. Available: https://doi.org/10.1145/3390557.3394318.
[146] A.B. d Souza, P.A.L. Rego, V. Chamola, T. Carneiro, P.H.G. Rocha, J.N. d Souza, A Bee Colony-Based Algorithm for Task Offloading in Vehicular Edge Computing, IEEE Syst. J. 17 (3) (2023) 4165–4176, https://doi.org/10.1109/JSYST.2023.3237363.
[147] F. Xu, S. Zi, J. Wang, J. Ma, A computing offloading strategy for UAV based on improved bat algorithm, /01/01/ 2023, Cogn. Robot. 3 (2023) 265–283, https://doi.org/10.1016/j.cogr.2023.07.005.
[148] S. Hong, J. Zhu, Y. Yuan, and J. Sun, An Energy-Efficient Metaheuristic for Offloading Security-Critical Tasks in Mobile Edge Computing with Integrated DPUs, in 2023 IEEE 3rd International Conference on Intelligent Technology and Embedded Systems (ICITES), 27-30 Oct. 2023 2023, pp. 157-164, doi: 10.1109/ICITES59818.2023.10356875.
[149] J.S. Priya, A. Bhagyalakshmi, K. Muthulakshmi, N. Deepa, DBAHHO: Deep belief network-based adaptive Harris Hawks optimization for adaptive offloading strategy in mobile edge computing, /10/01 2022, J. Supercomput. 78 (15) (2022) 16745–16769, https://doi.org/10.1007/s11227-022-04501-8.
[150] J. Wu, H. Dai, Y. Wang, S. Shen, C. Xu, PECCO: A profit and cost-oriented computation offloading scheme in edge-cloud environment with improved Moth-flame optimization, Concurr. Comput.: Pract. Exp. 34 (22) (2022) e7163.
[151] D.G. Zhang, H.Z. An, J. Zhang, T. Zhang, W.M. Dong, X.R. Jiang, Novel Privacy Awareness Task Offloading Approach Based On Privacy Entropy, 1, IEEE Trans. Netw. Serv. Manag. (2024) 1, https://doi.org/10.1109/TNSM.2024.3355967.
[152] H. Li, L. Liu, X. Duan, H. Li, P. Zheng, L. Tang, Energy-efficient offloading based on hybrid bio-inspired algorithm for edge–cloud integrated computation, /04/01/ 2024, Sustain. Comput.: Inform. Syst. 42 (2024) 100972, https://doi.org/10.1016/j.suscom.2024.100972.
[153] M. Hijjawi, et al., A Novel Hybrid Prairie Dog Algorithm and Harris Hawks Algorithm for Resource Allocation of Wireless Networks, IEEE Access 11 (2023) 145146–145166, https://doi.org/10.1109/ACCESS.2023.3335247.
[154] M. Nematollahi, A. Ghaffari, A. Mirzaei, Task and resource allocation in the internet of things based on an improved version of the moth-flame optimization algorithm, /04/01 2024, Clust. Comput. 27 (2) (2024) 1775–1797, https://doi.org/10.1007/s10586-023-04041-7.
[155] P. Singh, R. Singh, Energy-Efficient Delay-Aware Task Offloading in Fog-Cloud Computing System for IoT Sensor Applications, /10/17 2021, J. Netw. Syst. Manag. 30 (1) (2021) 14, https://doi.org/10.1007/s10922-021-09622-8.
[156] S.S. Jazaeri, S. Jabbehdari, P. Asghari, H.H.S. Javadi, An efficient edge caching approach for SDN-based IoT environments utilizing the moth flame clustering algorithm, /04/01 2024, Clust. Comput. 27 (2) (2024) 1503–1525, https://doi.org/10.1007/s10586-023-04023-9.
[157] L.X. Nguyen, Y.K. Tun, T.N. Dang, Y.M. Park, Z. Han, C.S. Hong, Dependency Tasks Offloading and Communication Resource Allocation in Collaborative UAV Networks: A Metaheuristic Approach, IEEE Internet Things J. 10 (10) (2023) 9062–9076, https://doi.org/10.1109/JIOT.2022.3233667.
[158] A. Lakhan, T.M. Groenli, G. Muhammad, P. Tiwari, Evolutionary Meta-Heuristic Offloading and Scheduling Schemes Enabled Industrial Cyber-Physical System, IEEE Syst. J. (2024) 1–10, https://doi.org/10.1109/JSYST.2023.3347523.
[159] K.M. Hosny, A.I. Awad, M.M. Khashaba, M.M. Fouda, M. Guizani, E.R. Mohamed, Enhanced multi-objective gorilla troops optimizer for real-time multi-user dependent tasks offloading in edge-cloud computing, /09/01/ 2023, J. Netw. Comput. Appl. 218 (2023) 103702, https://doi.org/10.1016/j.jnca.2023.103702.
[160] B. Kar, Y.-D. Lin, Y.-C. Lai, Cost optimization of omnidirectional offloading in two-tier cloud–edge federated systems, /06/01/ 2023, J. Netw. Comput. Appl. 215 (2023) 103630, https://doi.org/10.1016/j.jnca.2023.103630.
[161] C.R. Prasad, V. S. K, P.R. Rao, S. Kollem, S. Yalabaka, and S. Samala, Optimization of Task Offloading for smart cities using IoT with Fog Computing- A Survey, in

2022 International Conference on Signal and Information Processing (IConSIP), 26-27 Aug. 2022 2022, pp. 1-5, doi: 10.1109/ICoNSIP49665.2022.10007447.
[162] P. Kaur, S. Mehta, Improvement of Task Offloading for Latency Sensitive Tasks in Fog Environment, in: R. Tiwari, M. Mittal, L.M. Goyal (Eds.), Energy Conservation Solutions for Fog-Edge Computing Paradigms, Springer Singapore, Singapore, 2022, pp. 49–63.
[163] B.B. Dash, S.S. Patra, R. Satpathy, and B. Dash, Improvement of SDN-based Task Offloading using Golden Jackal Optimization in Fog Center, in 2023 World Conference on Communication & Computing (WCONF), 14-16 July 2023 2023, pp. 1-6, doi: 10.1109/WCONF58270.2023.10235040.
[164] V.S. Rao, K. Srinivas, Reduce Overall Execution Cost (OEC) of Computation Offloading in Mobile Cloud Computing for 5G Network Using Nature Inspired Computing (NIC) Algorithms, // 2022, in: N. Marriwala, C.C. Tripathi, S. Jain, D. Kumar (Eds.), Mobile Radio Communications and 5G Networks, Springer Nature Singapore, Singapore, 2022, pp. 679–690. // 2022.
[165] M.T. Khan, L. Barik, A. Adholiya, S.S. Patra, A.N. Brahma, and R.K. Barik, Task Offloading Scheme for Latency Sensitive Tasks In 5G IOHT on Fog Assisted Cloud Computing Environment, in 2022 3rd International Conference for Emerging Technology (INCET), 27-29 May 2022 2022, pp. 1-5, doi: 10.1109/INCET54531.2022.9824699.
[166] W. He, S. Wu, and J. Sun, An Effective Metaheuristic for Partial Offloading and Resource Allocation in Multi-Device Mobile Edge Computing, in 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), 20-22 Dec. 2021 2021, pp. 1419-1426, doi: 10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00213.
[167] C. Huang, Y. Yan, Y. Zhang, and J. Sun, A Metaheuristic Algorithm for Mobility-Aware Task Offloading for Edge Computing Using Device-to-Device Cooperation, in 2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta), 15-18 Dec. 2022 2022, pp. 656-663, doi: 10.1109/SmartWorld-UIC-ATC-ScalCom-DigitalTwin-PriComp-Metaverse56740.2022.00106.
[168] Y. Xu, Y. Wang, and J. Yang, Meta-Heuristic Search Based Model for Task Offloading and Time Allocation in Mobile Edge Computing, presented at the Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence, Tianjin, China, 2020. [Online]. Available: https://doi.org/10.1145/3404555.3404566.
[169] C. Zhao and X. Ding, A Levy Flight-Based Offloading Path Decision Scheme in VANET, in 2022 IEEE 5th International Conference on Electronics Technology (ICET), 13-16 May 2022 2022, pp. 985-993, doi: 10.1109/ICET55676.2022.9824880.
[170] A. Abbas, A. Raza, F. Aadil, M. Maqsood, Meta-heuristic-based offloading task optimization in mobile edge computing, 15501477211023021, Int. J. Distrib. Sens. Netw. 17 (6) (2021), https://doi.org/10.1177/15501477211023021.
[171] Y. Zeng and D. Liu, Computation Offloading Based on Improved Sparrow Search Algorithm in Edge Computing Scenario, in 2022 International Conference on Computing, Communication, Perception and Quantum Technology (CCPQT), 5-7 Aug. 2022 2022, pp. 231-236, doi: 10.1109/CCPQT56151.2022.00047.
[172] X. Wang, F. Qi, Z. Tang, M. Su, and X. Xing, An Energy-Efficient Computing Offloading Strategy Based on Improved Sparrow Search Algorithm in Mobile Edge Computing, in Proceedings of the International Conference on Internet of Things, Communication and Intelligent Technology, Singapore, J. Dong and L. Zhang, Eds., 2023// 2023: Springer Nature Singapore, pp. 59-73.
[173] G. Wei, A. Wang, C. Chen, and K. Huang, Multi-objective Sparrow Search Algorithm Based on Knee Points for Task Offloading in Multi-Cloudlet, in 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), 15-17 April 2022 2022, pp. 360-368, doi: 10.1109/ICSP54964.2022.9778362.
[174] M. Jehad, A. ELMoursy, and A. Khedr, Multi Objective Task Offloading in Fog Computing using Sparrow Search algorithm, in 2022 19th International Multi-Conference on Systems, Signals & Devices (SSD), 6-10 May 2022 2022, pp. 1292-1299, doi: 10.1109/SSD54932.2022.9955791.
[175] M. Su, G. Wang, J. Chen, Efficient task offloading with swarm intelligence evolution for edge-cloud collaboration in vehicular edge computing, Softw.: Pract. Exp. (2022).
[176] K. Ajmera, T.K. Tewari, SR-PSO: server residual efficiency-aware particle swarm optimization for dynamic virtual machine scheduling, /09/01 2023, J. Supercomput. 79 (14) (2023) 15459–15495, https://doi.org/10.1007/s11227-023-05270-8.
[177] K. Ajmera, T.K. Tewari, VMS-MCSA: virtual machine scheduling using modified clonal selection algorithm, /12/01 2021, Clust. Comput. 24 (4) (2021) 3531–3549, https://doi.org/10.1007/s10586-021-03320-5.
[178] A. Almadhor, A. Alharbi, A.M. Alshamrani, W. Alosaimi, H. Alyami, A new offloading method in the green mobile cloud computing based on a hybrid meta-heuristic algorithm, /12/01/ 2022, Sustain. Comput.: Inform. Syst. 36 (2022) 100812, https://doi.org/10.1016/j.suscom.2022.100812.
[179] D. Swaminathan, et al., ODTRA-Based Task Offload Optimization for Industrial Internet of Things: Improving Efficiency and Performance With Digital Twins and Metaheuristic Optimization, IEEE Access 12 (2024) 51796–51817, https://doi.org/10.1109/ACCESS.2024.3385636.
[180] N. Kumari, P.K. Jana, A metaheuristic-based task offloading scheme with a trade-off between delay and resource utilization in IoT platform, /11/29 2023, Clust. Comput. (2023), https://doi.org/10.1007/s10586-023-04193-6.
[181] S.I. AlShathri, S.A. Chelloug, D.S.M. Hassan, Parallel Meta-Heuristics for Solving Dynamic Offloading in Fog Computing, Mathematics 10 (8) (2022) 1258. ⟨https://www.mdpi.com/2227-7390/10/8/1258⟩ ([Online]. Available).
[182] K.M. Hosny, A.I. Awad, M.M. Khashaba, E.R. Mohamed, New improved multi-objective gorilla troops algorithm for dependent tasks offloading problem in multi-access edge computing, /04/01 2023, J. Grid Comput. 21 (2) (2023) 21, https://doi.org/10.1007/s10723-023-09656-z.
[183] M. Babar, A. Din, O. Alzamzami, H. Karamti, A. Khan, M. Nawaz, A bacterial foraging based smart offloading for IoT sensors in edge computing, /09/01/ 2022, Comput. Electr. Eng. 102 (2022) 108123, https://doi.org/10.1016/j.compeleceng.2022.108123.
[184] Y. Chen, D. Pi, S. Yang, Y. Xu, J. Chen, A.W. Mohamed, HNIO: a hybrid nature-inspired optimization algorithm for energy minimization in UAV-assisted mobile edge computing, IEEE Trans. Netw. Serv. Manag. 19 (3) (2022) 3264–3275, https://doi.org/10.1109/TNSM.2022.3176829.
[185] M. Nematollahi, A. Ghaffari, A. Mirzaei, Task offloading in Internet of Things based on the improved multi-objective aquila optimizer, /02/01 2024, Signal, Image Video Process. 18 (1) (2024) 545–552, https://doi.org/10.1007/s11760-023-02761-2.
[186] M. Maashi, et al., Elevating survivability in next-gen IoT-fog-cloud networks: scheduling optimization with the metaheuristic mountain gazelle algorithm, IEEE Trans. Consum. Electron. 70 (1) (2024) 3802–3809, https://doi.org/10.1109/TCE.2024.3371774.
[187] X. Shen, Z. Chang, X. Xie, S. Niu, Task offloading strategy of vehicular networks based on improved bald eagle search optimization algorithm, Appl. Sci. 12 (18) (2022) 9308. ⟨https://www.mdpi.com/2076-3417/12/18/9308⟩ ([Online]. Available).
[188] M. Alseid, A.A. El-Moursy, O. Alfawaz, A.M. Khedr, MSSAMTO-IoV: modified sparrow search algorithm for multi-hop task offloading for IoV, /12/01 2023, J. Supercomput. 79 (18) (2023) 20769–20789, https://doi.org/10.1007/s11227-023-05446-2.