



Paper Type: Original Article

Metaheuristic Algorithms for Computation Offloading: A Comprehensive Review

Amin Maghsoudi^{1,*} , Behnam Ghahramankhani²

¹ Department of Computer and Information Technology Engineering, Qa. c., Islamic Azad University, Qazvin, Iran; maghsoudi@iau.ac.ir.

² Department of Computer and Information Technology Engineering, La. c., Islamic Azad University, Lahijan, Iran; ghahramankhani@iau.ac.ir.

Citation:

Received: 24 June 2025
Revised: 08 August 2025
Accepted: 23 October 2025

Maghsoudi, A., & Ghahramankhani, B. (2025). Metaheuristic algorithms for computation offloading: A comprehensive review. *Metaheuristic algorithms with applications*, 2(4), 373-384.

Abstract

With the development of computing paradigms such as edge computing, fog computing, and cloud computing, and the need for users to process their requested services, deciding which services should be processed locally and which ones should be processed non-locally has become a challenge. This issue is addressed by the concept of computation offloading. Computation offloading is a process in which heavy computational tasks are transferred from resource-constrained devices, such as Internet of Things (IoT) devices, smartphones, and sensors, to remote computing servers with significant computational capacity. Due to the existence of numerous conditions, including environmental dynamics, user mobility, etc., and the existence of various criteria such as latency, energy consumption, cost, etc., the computation offloading problem is classified as NP-Hard. To address these problems and reach a near-optimal solution in a reasonable time, we can use metaheuristic algorithms. In this study, we review and analyze relevant research in the field of computational offloading using metaheuristic algorithms. This article can provide a perspective for researchers in this field and inform them about the challenges and open issues in this field.

Keywords: Metaheuristic algorithms, Computation offloading, Service placement, Computation paradigms.

1 | Introduction

In recent years, the use of computing, control, and data storage through cloud computing has aimed to take advantage of abundant computing resources to handle these applications. In this regard, tasks with high computational load are offloaded to remote cloud centers, which is known as computational offloading, to reduce the computational pressure on the terminal device [1], reducing the energy consumption for local processing and thus extending battery life. However, these centers are usually far from the end devices, and this relocation may not be a suitable solution for delay-constrained applications and increases bandwidth usage, cost, and energy consumption. To overcome these challenges, edge computing [2] and fog computing have been proposed as an extension of the cloud that can process data at the edge of the network and close

 Corresponding Author: maghsoudi@iau.ac.ir

 <https://doi.org/10.48313/maa.vi.25>



Licensee System Analytics. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0>).

to the end devices [3]. Fog computing introduces an edge layer by deploying fog servers between the end devices and remote cloud centers, which is mainly composed of terminals such as wireless access points, routers, base stations, etc. Since fog servers have network communication, computing, and storage functions and are close to terminal devices, they can improve the real-time processing response speed, reduce network congestion, and enhance data privacy protection capabilities.

Computational offloading is a multi-objective, NP-hard optimization problem [4] that cannot be solved in polynomial time using traditional deterministic algorithms and can only obtain an approximate solution. Previous research can be classified into the categories of mathematical optimization algorithms such as Mixed Integer Programming (MIP), heuristic, metaheuristic, and game theory; artificial intelligence-based optimization algorithms such as Machine Learning (ML); and control theory-based algorithms such as the Lyapunov optimization approach, or a combination of them.

Some of the main contributions that illustrate the innovations of the current paper are mentioned below:

- I. A classification of the works done in the field of computational offloading using metaheuristic algorithms is presented.
- II. Metaheuristic algorithms are discussed in terms of the methods used. Solving the computational offloading problem in resource-constrained environments requires the use of appropriate and efficient metaheuristic algorithms.
- III. Challenges and unsolved issues regarding computational offloading using metaheuristic algorithms are enumerated and suggested for future research areas.

The rest of the paper is as follows. Section 2 describes the basic concepts, including metaheuristic algorithms and computational offloading. Section 3 shows how the relevant research was collected. Also, Section 4 describes the metaheuristic algorithms used for computational offloading. Section 5 analyzes and reviews the relevant research. Section 6 discusses the challenges and unsolved issues, and Section 7 describes the conclusions of the current paper.

2 | Expression of Concepts

In this section, the main concepts of the research, including metaheuristic algorithms and computational offloading, are described. A detailed understanding of the aforementioned concepts can highlight the importance of the present work and facilitate a more detailed examination of related research and a better comparison of them in terms of various indicators.

2.1 | Metaheuristic Algorithms

Metaheuristic algorithms are a type of stochastic algorithms used to find the optimal solution. Optimization methods and algorithms are divided into two categories: exact algorithms and approximate algorithms. Exact algorithms are able to find the solution exactly, but in hard optimization problems, the efficiency is not sufficient, and their implementation increases exponentially compared to the problems. Approximate algorithms are able to find good solutions (close to optimal) in a short solution time for hard optimization problems. Approximate algorithms are also divided into three categories: heuristic, metaheuristic, and hyperheuristic. The two main problems of heuristic algorithms are getting stuck in local optimum points and premature convergence to these points. Metaheuristic algorithms are presented to solve these problems of primitive algorithms. In fact, metaheuristic algorithms are a type of approximate algorithm that has subsamples of points for local purposes, and one of them can be used [5].

The process of designing and implementing metaheuristic algorithms has three consecutive stages, each of which has different steps. In each step, activities must be done to complete that step. The first stage is preparation, in which we must get a detailed understanding of the problem we want to solve, and the goals of designing a metaheuristic algorithm for it must be clearly defined according to the existing solution methods for this problem. The next step is called construction. The main goals of this stage are to choose the

solution strategy, define performance measurement criteria, and design the algorithm for the chosen strategy. The last step is the implementation, in which the algorithm designed in the previous step, including parameter setting, performance analysis, and finally compiling and preparing the results report, should be done [6].

Various criteria can be used to classify metaheuristic algorithms [7]:

- I. Based on one answer and population-based: algorithms based on one answer change one answer during the search process, while in population-based algorithms, a population of answers is considered during the search.
- II. Inspired by nature and not inspired by nature: many metaheuristic algorithms are inspired by nature, while some metaheuristic algorithms are not inspired by nature.
- III. With memory and without memory: some metaheuristic algorithms are memoryless, in the sense that these types of algorithms do not use the information obtained during the search (for example, simulated annealing). This is despite the fact that some metaheuristic algorithms, such as forbidden search, use memory. This memory stores the information obtained during the search.
- IV. Deterministic and probabilistic: a deterministic metaheuristic algorithm such as forbidden search solves the problem using deterministic decisions. But in probabilistic metaheuristic algorithms such as simulated annealing, a series of probabilistic rules are used during the search.

2.2 | Computation Offloading

Due to the limited computing, storage, and power resources of end-user devices, they can offload their computing tasks to resource-intensive servers such as fog or cloud servers. Computing offload is the sending of application components or resource-intensive computing tasks to remote servers when resources such as cloud or fog computing are used to enhance the capabilities of user devices and to accelerate delay-sensitive and resource-intensive applications. Computing offload can provide many benefits, including reducing the cost incurred by the user, saving energy consumption and battery, utilizing the abundant processing resources of remote servers, and improving performance, supporting more applications, and reducing response time. Computing offload can ensure the quality of service for users by sending delay-sensitive and intensive computing tasks to edge/fog/cloud devices [8].

Offloading decisions can be broadly divided into two types: binary (full) offloading and partial offloading [9]. A single or relatively simple task cannot be partitioned and must be processed as a whole locally on the end device or offloaded to a fog server, which is called binary offloading. In binary decision-making, the user device decides for some reason to process the entire task (component or workload) completely locally on the end device or decides to offload the entire task to the fog server for processing. Binary offloading can occur when offloading the task is not cost-effective and requires a lot of energy and time, or when these tasks must access components on the user device to execute, or when fog server resources are not available [10].

Some tasks can be executed locally on the user device and offloaded to remote servers. The decision depends on whether the difference between the cost of executing locally and the computational offloading cost outweighs the communication cost of offloading. When the transmission channel is in bad condition, the time and energy consumption during transmission will be high, and it is not economical to perform offloading, but when there is a server with a suitable transmission channel and large computing capacity, it is more appropriate to offload more computational tasks to the offloading server. In binary offloading, if the task is large, it can cause problems such as high network overhead [11].

To avoid this, a main task can be divided into smaller tasks, and a subset of these smaller tasks can be offloaded to the fog server or cloud, and the rest of them can be processed locally. Some factors, including the model and type of applications, the type of connection and wireless environments, user preferences, specifications and capabilities of the end device, and specifications and capabilities of the servers, affect the offloading decision [12]. Determining the offloading rate and also deciding to select the appropriate fog server

for offloading tasks and guaranteeing resources to determine the amount of computing, communication, and energy resources required for a task are the main issues in offloading computational tasks [13].

If a server has a high workload or the channel conditions are unsuitable, more time may be spent offloading and receiving results from edge servers. In this method, a decision must be made as to which tasks will be executed locally on the user's device and which tasks will be offloaded to fog or cloud servers, which adds a level of complexity to the problem. In case of using the partial offloading method, tasks need to be broken down. That is, a total task is broken down into several subtasks. The purpose of partitioning tasks is to select a task for offloading, distribute tasks over more servers for computation, more efficient use of resources, and less overhead. Partial offloading can cause execution time overhead [14].

3| Collecting Relevant Research

In this article, we are looking for a comprehensive review of metaheuristic algorithms for computation offloading. Collecting relevant research and subsequently reviewing them should follow a specific order, which is done by modeling articles [15], [16].

3.1| How to Search for Relevant Research

We counted a number of articles published from 2021 to 2025 that demonstrate computation offloading using metaheuristic algorithms. In this regard, articles from various magazines and conferences were considered. The most common scientific databases used globally in this research are ieeexplore.ieee.org, sciencedirect.com, and Springerlink.com. Articles were searched using the keywords "metaheuristic algorithm", "heuristic algorithm", and "computation offloading". The diagram of how to search for relevant research is shown in Fig. 1.

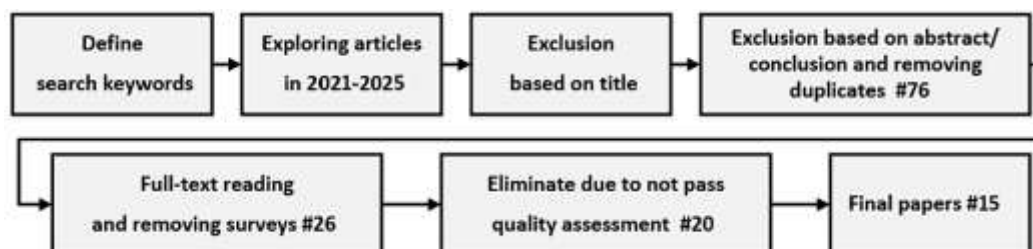
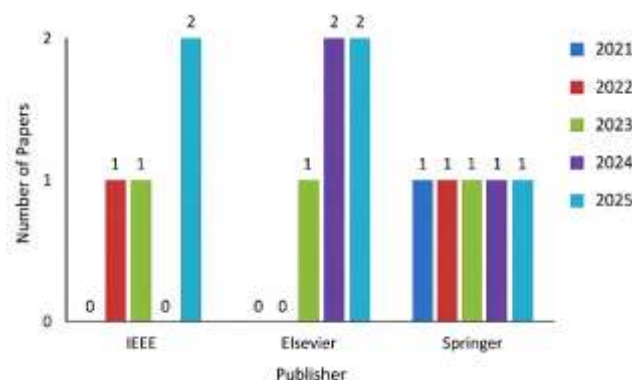


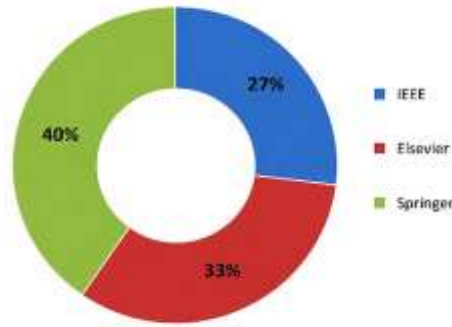
Fig. 1. Diagram of how to search for relevant research.

3.2| Diversity of Relevant Research

Fig. 2 shows the distribution of articles based on the year of publication and the name of the publisher. As we can see, Springer has the largest share compared to other publishers with 40%, followed by Elsevier with 33%. Fig. 3 shows the number of publications by year. Fig. 2 shows an upward trend in the number of studies conducted from 2021 to 2025.



a.



b.

Fig. 2. Distribution chart of papers: a. by publisher over the years, b. by publisher name.

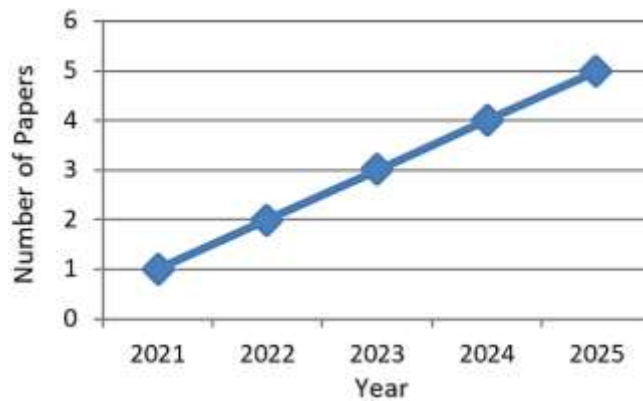


Fig. 3. Number of publications per year.

4 | Metaheuristic Algorithms for Computation Offloading

In [17], task offloading is presented as a multi-objective optimization problem focusing on reducing the total system power consumption and task execution delay. Then, according to the NP-hard nature of the problem, it has been solved using two metaheuristic methods, i.e., the Non-Dominated Sorting Genetic Algorithm (NSGA-II) and the Bees Algorithm. The simulation results support the robustness of both metaheuristic algorithms in terms of energy consumption and delay reduction. The proposed method achieves a better trade-off regarding the probability of discharge and the power required for data transmission.

In [18], a metaheuristic scheduler, Smart Ant Colony Optimization task offloading algorithm inspired by nature, is proposed to offload the Internet of Things (IoT)-sensor applications tasks in a fog environment. Numerical results show a significant improvement in delay in task offloading of IoT-sensor applications. In [19], an improved metaheuristic optimization algorithm called particle swarm optimization based on genetic learning is designed, which integrates the strong local search capacity of a particle swarm optimizer and the genetic operation of a genetic algorithm. The proposed method jointly optimizes the offloading of tasks, transmission power, and bandwidth allocation of existing channels. Simulations with real-world data show that the proposed method significantly outperforms other existing methods in energy consumption of a whole system.

In [20], an attempt has been made to investigate the profitability of computational offloading from the service provider's perspective. Initially, near-optimal evacuation decisions are determined using evolutionary algorithms such as genetic algorithms and binary particle swarm optimization algorithms. After several iterations of the evolutionary process for the evacuation decision, the optimal solution is finally obtained, which allocates the resources based on the exact calculation of the profit value. According to the performed

simulations, the proposed evolutionary algorithms perform better than the spectral efficiency-based evacuation algorithm, with shorter times in terms of profitability.

In [21], an offload decision-making framework based on the binary cuckoo search algorithm is proposed for mobile edge computing. Simulation results show that offload decisions depend on several parameters, including the number of mobile devices managed by the edge server, bandwidth, and number of tasks. The decisions made also depend on the priority assigned to each target. The proposed method significantly reduces the execution time when a large number of mobile devices are involved. In [22], a layered fitting algorithm is proposed that distributes tasks equally between the fog and the cloud, based on priority levels. Also, a metaheuristic approach based on modified Harris-Hawks optimization is proposed to allocate the best available resource to a task in a layer. The main goal is to reduce task completion time, task execution cost, and power consumption and increase resource utilization in both the fog and cloud layers. Based on the results, the system performance is improved in terms of task completion time, execution cost, and power consumption.

In [23], a new metaheuristic approach based on a modified genetic algorithm with directed search operators is presented to optimize overall execution time and energy consumption. A comparison of the efficiency of the proposed approach with alternatives shows improvements in solution quality and convergence speed to the solution. In [24], a partial computational offloading technique with minimum energy is proposed. A constrained optimization problem is formulated to minimize the cost for the system, and then an improved hybrid metaheuristic algorithm is developed that synergistically combines the Metropolis acceptance criterion from annealing simulation and genetic operations. Experiments with real tasks have shown that the proposed approach has significantly lower system cost and faster convergence speed than benchmarks.

In [25], an improved Aquila multi-objective optimizer equipped with a Pareto front is proposed to offload tasks from IoT devices to fog nodes with the aim of reducing response time. To improve the algorithm, contrast-based learning is used to diversify the population and discover optimal solutions. The proposed algorithm is evaluated with the number of tasks and fog nodes to reduce the response time. The results show that the average response time and failure rate obtained by the proposed algorithm are improved compared to other algorithms. In [26], a hybrid genetic algorithm is proposed to solve the task offloading problem. The speed of simulated annealing and the accuracy of the genetic algorithm are used to develop a fast and accurate Hybrid algorithm. Experimental results show that the proposed approach achieves significantly lower scheduling time compared to other schemes.

In [27], a hybrid metaheuristic algorithm for task offloading in edge computing is proposed, which is based on the simulated annealing algorithm and the red deer algorithm to generate an optimal task offloading schedule that minimizes system completion time and transfer time. Simulation results prove that the proposed approach can significantly reduce system completion time and transfer time. In [28], a hybrid metaheuristic algorithm combined with a greedy rule, hybrid mutation, and the whale optimization algorithm is proposed to realize partial computational offloading. Also, a partial offloading architecture of heterogeneous edge intelligence is proposed to validate the model and algorithm with real-world hardware and software. Experimental results show that the proposed approach outperforms several classical optimization algorithms in minimizing energy consumption.

In [29], the joint optimization of task offloading, communication, and computational resource allocation in edge computing is investigated, aiming to minimize energy consumption under deadline constraints and capacity constraints. The decision space is divided into several decision regions based on the characteristics of the problem under study, and a task offloading and resource allocation algorithm based on particle swarm optimization is designed to manage each decision region separately. The simulation and testbed-based experimental results show that the proposed algorithm can save energy consumption and improve efficiency compared to existing algorithms.

In [30], binary particle swarm optimization is used to make workload offloading decisions that consider both system cost and quality of service. Experimental results show that the proposed approach significantly outperforms existing benchmark algorithms in balancing system cost and quality of service in shared cloud-edge-end scenarios. In [31], a hybrid computational offload framework is presented that integrates edge and cloud resources to facilitate efficient and scalable data processing. The proposed system integrates a genetic

algorithm and particle swarm optimization to enhance task offload, reduce delay, and optimize resource utilization. Experimental findings show that this approach significantly enhances system performance, minimizes energy consumption, and ensures stable connectivity between different devices.

Table 1. Metaheuristic algorithms used for computation offloading.

Publisher	Future Work	Simulator	Paradigm	Main Goals	Metaheuristic Algorithm	Publication Year	Reference
Springer	Using ML methods	iFogSim	Fog	Reducing the total system power consumption and task execution delay	Genetic and bees	2021	[17]
Springer	—	iFogSim	Fog	Improvement in delay in task offloading of IoT-sensor applications	Ant Colony	2022	[18]
IEEE	—	MATLAB	Edge	Optimizes the offloading of tasks, transmission power, and bandwidth allocation of existing channels	Particle swarm	2022	[19]
IEEE	Considering new optimization modeling, including objective function and constraints	MATLAB	Mobile edge	Improvement of profitability	Genetic and particle swarm	2023	[20]
Elsevier	—	Cloudsim	Mobile edge	Reduction of execution time	Cuckoo	2023	[21]
Springer	—	iFogSim	Fog	Improvement of completion time, execution cost, and power consumption	Harris-Hawks	2023	[22]
Elsevier	—	MATLAB	Edge	Optimize the overall execution time and energy consumption	Genetic	2024	[23]
Elsevier	—	MATLAB	Edge	Lower system cost and faster convergence speed	Simulated annealing and genetic	2024	[24]
Springer	—	iFogSim	Fog	Improve response time and failure rate	Aquila	2024	[25]
Springer	—	iFogSim	Edge	Improvement of scheduling time	Genetic	2024	[26]
Springer	—	MATLAB	Edge	Minimizing system completion time and transfer time	Simulated annealing and red deer	2025	[27]
IEEE	—	MATLAB	Edge	Minimizing energy consumption	Whale	2025	[28]
IEEE	—	MATLAB	Edge	Minimizing energy consumption	Particle swarm	2025	[29]
Elsevier	—	MATLAB	Edge	Balancing system cost and quality of service	Particle swarm	2025	[30]
Elsevier	—	iFogSim	Edge	Increase task offload, reduce delay, and optimize resource utilization	Genetic and particle swarm	2025	[31]

5 | Analysis and Scrutiny of Related Research

In this section, we analyze and review related research to provide an overview for those interested in computational offloading using metaheuristic algorithms. This helps researchers make the right choice of improvement criteria, the paradigm under discussion, the simulator used, and other important research parameters.

Figure 4 shows the number of uses of each metaheuristic algorithm in the relevant papers. As we can see, the popular metaheuristic algorithms such as the genetic algorithm and particle swarm optimizer have been the most used, each of which has been used by 6 and 5 papers, respectively.

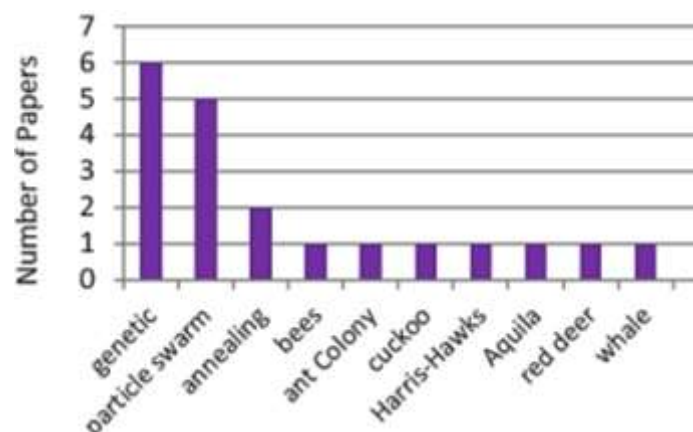


Fig. 4. Distribution of metaheuristic algorithms for computation offloading.

As we can see in Fig. 5, today, reducing execution time delay, reducing energy consumption, and reducing execution cost are the criteria that have received more attention from researchers than other criteria. A review of the relevant articles shows that 38%, 21%, and 13% of previous articles focused on reducing execution time delay, reducing energy consumption, and reducing execution cost, respectively. Delay-sensitive and real-time applications have made the delay criterion very important, and the limitation of users' battery life has also made energy consumption one of the important criteria. On the other hand, system developers want to implement at the lowest cost, so cost will also be one of the criteria of interest.

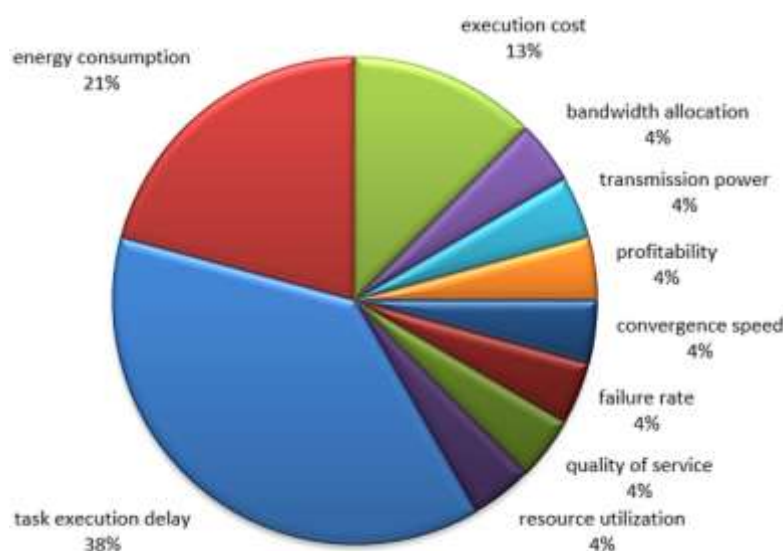


Fig. 5. Main metrics used in relevant research.

As we can see in Fig. 6, edge computing is the most popular paradigm in research related to computational offloading using metaheuristic algorithms. The importance of accelerating the processing of user-requested services and the need for processing resources to be close to users will further increase the use of edge computing. Also, user mobility and addressing this challenge will cause the use of the mobile edge computing paradigm to increase further.

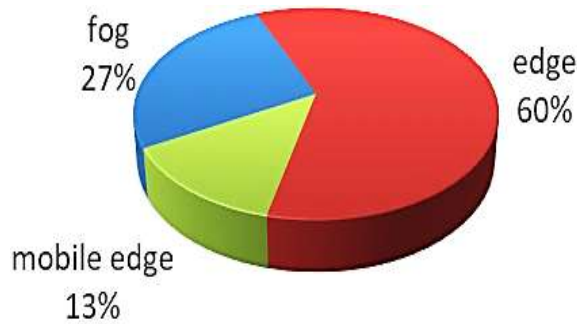


Fig. 6. Distribution of paradigms in relevant research.

As shown in Fig. 7, 53% of the research on computational offloading using metaheuristic algorithms used the MATLAB environment to evaluate the results. Also, about 40% of the simulation studies were conducted using the iFogSim tool. iFogSim brings ease in evaluating fog and edge networks due to its ready-made modules and classes.

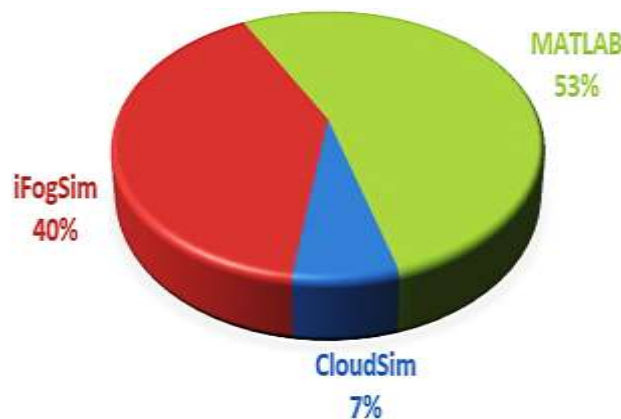


Fig. 7. Simulators used in relevant research.

6 | Challenges and Issues

With the expansion of delay- and energy-sensitive programs, optimization of computational load unloading to increase service quality criteria, including reducing delay and energy consumption, has been a hot topic of interest in recent years, and many studies have been presented in this regard with different approaches and in different computing environments. However, it has always faced challenges due to the dynamism and uncertainty of computing environments in real-world scenarios. In this section, we point out some of the main challenges of the previous approaches in computation offloading:

I. Getting stuck in local optima and setting too many hyperparameters:

Some of the unloading research, despite their achievements, have their own shortcomings. The biggest problem of metaheuristic methods is getting stuck in local optima. To obtain a global optimal solution, they must make a trade-off between exploration and exploitation. In addition, metaheuristic algorithms have many meta-parameters that are often difficult to tune. These are parameters that are manually set by the researcher. For example, in a genetic algorithm, crossover rate and mutation rate must be carefully adjusted. In contrast, the development of newer algorithms can require providing fewer parameters and better global search capabilities to be used to model more complex high-dimensional real-world scenarios such as offloading in fog computing.

II. Failure to consider the dynamics of the environment and lack of interaction with the environment:

Some of the unloading research, despite their achievements, often considers the environment static for convenience. This unrealistic assumption cannot include real-world requirements with dynamic environment conditions such as time-varying transmission channel conditions, bandwidth limitations, computing

capabilities of servers, energy available in user devices, and random arrival rates of tasks. Considering that system dynamics play an important role in long-term performance, these methods may fail to achieve long-term computing discharge performance in dynamic, variable environments. The variety of tasks and the dynamics of the fog computing environment make the problem of offloading complex and difficult. In such a dynamic and complex system, traditional optimization methods may not be able to converge to an optimal solution in a reasonable time.

III. Exponential increase in search space and high computational load:

Existing algorithms usually suffer from an exponential increase in search space and heavy computational load, especially in large-scale scenarios. In most real-world scenarios, system dynamics are difficult to measure or model. This difficulty is especially true when the dimensions of the search spaces are large. The use of ML methods, especially deep reinforcement learning, can lead to the improvement of the situation during evacuation operations by gathering new experiences. As a result, the needs of delay-sensitive applications can be met by increasing adaptability to environmental changes.

7 | Conclusion

With the development of computing paradigms such as edge computing, fog computing, and cloud computing, and the need for users to process their requested services, deciding which services should be processed locally and which ones should be processed non-locally has become a challenge. This issue is addressed by the concept of computation offloading. Computation offloading is a process in which heavy computational tasks are transferred from resource-constrained devices, such as IoT devices, smartphones, and sensors, to remote computing servers with significant computational capacity. Due to the existence of numerous conditions, including environmental dynamics, user mobility, etc., and the existence of various criteria such as latency, energy consumption, cost, etc., the computation offloading problem is classified as NP-Hard. To address these problems and reach a near-optimal solution in a reasonable time, we can use metaheuristic algorithms. In this study, we reviewed and analyzed related research in the field of computational offloading using metaheuristic algorithms. This article can provide a perspective for researchers interested in this field and inform them about the challenges and open issues in this field.

Author Contribution

All authors contributed to the ideation and design of the study. A. M. wrote the initial draft and collected and analyzed the data. Project supervision and validation of the results were performed by B. Gh. All authors read and approved the final version.

Funding

This research did not receive any specific grant from funding agencies.

Data Availability

All the data are available in this paper.

Conflicts of Interest

The authors declare no conflict of interest.

Consent for Publication

The author confirms consent for the publication of this work.

Ethics Approval and Consent to Participate

This article does not contain any studies with human participants performed by the author.

References

- [1] Chen, X., & Liu, G. (2021). Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks. *IEEE internet of things journal*, 8(13), 10843-10856. <https://doi.org/10.1109/JIOT.2021.3050804>
- [2] Shi, W., & Dustdar, S. (2016). The promise of edge computing. *Computer*, 49(5), 78-81. <https://doi.org/10.1109/MC.2016.145>
- [3] Ren, J., Zhang, D., He, S., Zhang, Y., & Li, T. (2019). A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. *ACM computing surveys (CSUR)*, 52(6), 1-36. <https://doi.org/10.1145/3362031>
- [4] Islam, A., Debnath, A., Ghose, M., & Chakraborty, S. (2021). A survey on task offloading in multi-access edge computing. *Journal of systems architecture*, 118, 102225. <https://doi.org/10.1016/j.sysarc.2021.102225>
- [5] Tomar, V., Bansal, M., & Singh, P. (2024). Metaheuristic algorithms for optimization: A brief review. *Engineering proceedings*, 59(1), 238. <https://doi.org/10.3390/engproc2023059238>
- [6] Yaghini, M., & Akhavan, R. (2010). DIMMA: A design and implementation methodology for metaheuristic algorithms-A perspective from software development. *International journal of applied metaheuristic computing (IJAMC)*, 1(4), 57-74. <https://doi.org/10.4018/jamc.2010100104>
- [7] Talbi, E. G. (2009). *From design to implementation*. Hoboken, NY: John Wiley. https://zeus.inf.ucv.cl/~bcrawford/DiplomadoIA_2024/Cap1_Metaheuristics_Talbi.pdf
- [8] Zabihi, Z., Eftekhari Moghadam, A. M., & Rezvani, M. H. (2023). Reinforcement learning methods for computation offloading: A systematic review. *ACM computing surveys*, 56(1), 1-41. <https://doi.org/10.1145/3603703>
- [9] Shakarami, A., Ghobaei-Arani, M., & Shahidinejad, A. (2020). A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective. *Computer networks*, 182, 107496. <https://doi.org/10.1016/j.comnet.2020.107496>
- [10] Feng, C., Han, P., Zhang, X., Yang, B., Liu, Y., & Guo, L. (2022). Computation offloading in mobile edge computing networks: A survey. *Journal of network and computer applications*, 202, 103366. <https://doi.org/10.1016/j.jnca.2022.103366>
- [11] Choudhury, A., Ghose, M., & Islam, A. (2024). Machine learning-based computation offloading in multi-access edge computing: A survey. *Journal of systems architecture*, 148, 103090. <https://doi.org/10.1016/j.sysarc.2024.103090>
- [12] Mach, P., & Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE communications surveys & tutorials*, 19(3), 1628-1656. <https://doi.org/10.1109/COMST.2017.2682318>
- [13] Lin, H., Zeadally, S., Chen, Z., Labiod, H., & Wang, L. (2020). A survey on computation offloading modeling for edge computing. *Journal of network and computer applications*, 169, 102781. <https://doi.org/10.1016/j.jnca.2020.102781>
- [14] Peng, P., Lin, W., Wu, W., Zhang, H., Peng, S., Wu, Q., & Li, K. (2024). A survey on computation offloading in edge systems: From the perspective of deep reinforcement learning approaches. *Computer science review*, 53, 100656. <https://doi.org/10.1016/j.cosrev.2024.100656>
- [15] Keshavarz Haddadha, P., Rezvani, M. H., MollaMotalebi, M., & Shankar, A. (2024). Machine learning methods for service placement: a systematic review: PK Haddadha et al. *Artificial intelligence review*, 57(3), 61. <https://doi.org/10.1007/s10462-023-10684-0>
- [16] Motlagh, S. M. H., Rezvani, M. H., & Khounsivash, M. (2025). AI methods for personality traits recognition: A systematic review. *Neurocomputing*, 640, 130301. <https://doi.org/10.1016/j.neucom.2025.130301>
- [17] Keshavarznejad, M., Rezvani, M. H., & Adabi, S. (2021). Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms. *Cluster computing*, 24(3), 1825-1853. <https://doi.org/10.1007/s10586-020-03230-y>

- [18] Kishor, A., & Chakarbarty, C. (2022). Task offloading in fog computing for using smart ant colony optimization. *Wireless personal communications*, 127(2), 1683-1704. <https://doi.org/10.1007/s11277-021-08714-7>
- [19] Bi, J., Yuan, H., Zhang, K., & Zhou, M. (2022). Energy-minimized partial computation offloading for delay-sensitive applications in heterogeneous edge networks. *IEEE transactions on emerging topics in computing*, 10(4), 1941-1954. <https://doi.org/10.1109/TETC.2021.3137980>
- [20] Singh, S., & Kim, D. H. (2023). Joint optimization of computation offloading and resource allocation in C-RAN with mobile edge computing using evolutionary algorithms. *IEEE access*, 11, 112693-112705. <https://doi.org/10.1109/ACCESS.2023.3322650>
- [21] Alqarni, M., Cherif, A., & Alkayyal, E. (2023). ODM-BCSA: An offloading decision-making framework based on binary cuckoo search algorithm for mobile edge computing. *Computer networks*, 226, 109647. <https://doi.org/10.1016/j.comnet.2023.109647>
- [22] Yakubu, I. Z., & Murali, M. (2023). An efficient meta-heuristic resource allocation with load balancing in IoT-Fog-cloud computing environment. *Journal of ambient intelligence and humanized computing*, 14(3), 2981-2992. <https://doi.org/10.1007/s12652-023-04544-6>
- [23] Rajapackiyam, E., Devi, A., Reddy, M. S., Arumugam, U., Vairavasundaram, S., Vairavasundaram, I., & Suresh, V. (2024). An efficient computation offloading in edge environment using genetic algorithm with directed search techniques for IoT applications. *Future generation computer systems*, 158, 378-390. <https://doi.org/10.1016/j.future.2024.04.021>
- [24] Yuan, H., Bi, J., Wang, Z., Yang, J., & Zhang, J. (2024). Partial and cost-minimized computation offloading in hybrid edge and cloud systems. *Expert systems with applications*, 250, 123896. <https://doi.org/10.1016/j.eswa.2024.123896>
- [25] Nematollahi, M., Ghaffari, A., & Mirzaei, A. (2024). Task offloading in Internet of Things based on the improved multi-objective aquila optimizer. *Signal, image and video processing*, 18(1), 545-552. <https://doi.org/10.1007/s11760-023-02761-2>
- [26] Ben Sada, A., Khelloufi, A., Naouri, A., Ning, H., & Dhelim, S. (2024). Hybrid metaheuristics for selective inference task offloading under time and energy constraints for real-time IoT sensing systems. *Cluster computing*, 27(9), 12965-12981. <https://doi.org/10.1007/s10586-024-04578-1>
- [27] Abuthahir, S. S., & Peter, J. S. P. (2025). A hybrid meta-heuristic algorithm for task offloading in vehicular edge computing network. *Wireless personal communications*, 141(1), 51-74. <https://doi.org/10.1007/s11277-025-11769-5>
- [28] Xu, B., Ruan, Y., Qiu, C., He, S., Shu, F., Kang, X., & Zhang, L. (2025). Multilevel and energy-efficient partial computation offloading in heterogeneous edge intelligence. *IEEE internet of things journal*, 12(11), 14993-15007. <https://doi.org/10.1109/JIOT.2025.3529185>
- [29] Zeng, Y., Zeng, Y., Chen, J., Shen, Y., Li, L., Cong, P., ... & Li, K. (2025). AoI-oriented computation offloading and resource allocation for end-edge-cloud computing systems. *IEEE internet of things journal*, 12(19), 41118-41134. <https://doi.org/10.1109/JIOT.2025.3591682>
- [30] Liu, L., & Zhang, Y. (2025). Task offloading optimization for multi-objective based on cloud-edge-end collaboration in maritime networks. *Future generation computer systems*, 164, 107588. <https://doi.org/10.1016/j.future.2024.107588>
- [31] Khan, S., Liu, S., Pan, L., & Mei, G. (2025). Optimization-based hybrid offloading framework for IoMT in edge-cloud healthcare systems. *Future generation computer systems*, 176, 108163. <https://doi.org/10.1016/j.future.2025.108163>