Computer Networks 254 (2024) 110791

Contents lists available at ScienceDirect

Computer Networks

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

Review article

# Task offloading strategies for mobile edge computing: A survey

Shi Dong [a,*], Junxiao Tang [a], Khushnood Abbas [a], Ruizhe Hou [b], Joarder Kamruzzaman [c], Leszek Rutkowski [d], Rajkumar Buyya [e]

[a] School of Computer Science and Technology, Zhoukou Normal University, Zhoukou, 466001, China
[b] the School of Automation Science and Engineering, South China University of Technology, Guangzhou, 510641, China
[c] the School of Science, Engineering and Information Technology, Federation University, Ballarat, VIC 3350, Australia
[d] the Institute of Computer Science, AGH University of Science and Technology, Warsaw, 01-447, Poland
[e] the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Parkville, VIC 3010, Australia

ARTICLE INFO

*Keywords:*
MEC
Computation offloading
Task offloading algorithm
Offloading decision

ABSTRACT

With the wide adoption of 5G technology and the rapid development of 6G technology, a variety of new applications have emerged. A multitude of compute-intensive and time-sensitive applications deployed on terminal equipment have placed increased demands on Internet delay and bandwidth. Mobile Edge Computing (MEC) can effectively mitigate the issues of long transmission times, high energy consumption, and data insecurity. Task offloading, as a key technology within MEC, has become a prominent research focus in this field. This paper presents a comprehensive review of the current research progress in MEC task offloading. Firstly, it introduces the fundamental concepts, application scenarios, and related technologies of MEC. Secondly, it categorizes offloading decisions into five aspects: reducing delay, minimizing energy consumption, balancing energy consumption and delay, enabling high-computing offloading, and addressing different application scenarios. It then critically analyzes and compares existing research efforts in these areas.

**Contents**

* Corresponding author.
*E-mail addresses:* dongshi@zknu.edu.cn (S. Dong), rutkowski@agh.edu.pl (L. Rutkowski).

https://doi.org/10.1016/j.comnet.2024.110791
Received 8 April 2024; Received in revised form 11 August 2024; Accepted 6 September 2024
Available online 12 September 2024

## 1. Introduction

Cloud computing used in many applications. However, it is not suitable for real-time IoT applications that require low-latency performance. Concurrently, the vigorous expansion of Internet of Things (IoT) technology and the widespread adoption of interconnected smart technology has engendered a relentless exponential surge in the demand for device-based computation. In response to these formidable challenges, there has been an observable paradigm shift within the realm of computation towards the burgeoning field of distributed edge computing [2].

### 1.1. Motivation

The motivation for this work arises from the transformative impact of Mobile Edge Computing (MEC) in meeting the growing computational demands of modern applications, driven by the rapid advancement of technologies such as 5G and the anticipated 6G networks. With the proliferation of energy-intensive and latency-sensitive applications-ranging from augmented reality to autonomous vehicles, there is a critical need for efficient task offloading strategies that optimize resource utilization while minimizing delays and energy consumption. As existing cloud computing paradigms struggle to meet the performance requirements of increasingly distributed and dynamic environments, MEC emerges as a pivotal solution. It enables computations to be processed closer to the data source, thereby reducing the burden on centralized data centers and enhancing responsiveness. This survey aims to present a thorough overview of current research efforts in MEC task offloading strategies, highlighting significant methodologies, challenges, and future directions. By doing so, it seeks to inform researchers and practitioners of the state of the art, fostering further innovation in this critical area of study and contributing to the realization of efficient, sustainable, and intelligent edge computing systems. The exploration of offloading algorithms not only addresses pressing issues in computational efficiency and resource management but also aligns with the broader vision of smart and connected environments that underpin the future of technology.

### 1.2. Research methodology

This section explains the search method used for task offloading in an edge computing environment. First, a systematic literature review was conducted, following specific steps to explore, collect, categorize, analyze, and assess relevant articles in this field. The review focused on identifying and selecting pertinent articles based on predefined inclusion and exclusion criteria. We have used several databases and search engines, including Google Scholar, Web of Science, DBLP, and relevant papers and their corresponding citations, to gather comprehensive and relevant articles. This approach ensured a thorough and well-rounded understanding of content caching in edge computing environments.

### 1.3. Mobile Edge Computing (MEC) overview

The Cisco Internet Report (2018–2023) outlined a prediction for the year 2023 that each individual is anticipated to have an average of 3.6 Internet-connected devices [3]. Moreover, within households, the number of devices and connections is expected to reach 10, with approximately 47% of these devices featuring video functionality. This signifies a proliferation of communication devices that will generate substantial data traffic. The adoption of 5G and the ongoing development of 6G communication networks are poised to offer robust capabilities for handling the escalating volume of data traffic. To address the escalating demand for computing resources, a novel computing paradigm is emerging, one that involves the integration of additional devices to furnish ample computing power support for ubiquitous computing environments. This transition is particularly noteworthy in light of the "post-Moore's era", wherein the historical trend of guaranteed performance growth in integrated circuit technology is no longer assured. Consequently, the concept of establishing expansive computing networks is gaining prominence as a critical strategy to meet the evolving demands of the digital age.

In this context, while network computing offers significant computational power for data processing, it still falls short of meeting data demands. Consider, for instance, the cloud computing paradigm in an airport monitoring application, where thousands of cameras are deployed to ensure airport security, each capable of generating 12 Mbps of data [4]. The analysis of transmitted video data alone on the central cloud server necessitates hundreds of gigabits per second (Gbps) in bandwidth resources for collection. This would significantly elevate the load on high-traffic servers within the central core network and lead to increased processing and transmission delays. Consequently, the notion of entrusting all computations and data to a centralized cloud computing center is proving both impractical and unreasonable. As an alternative paradigm, in the MEC, the edge server assumes the role of a "distributed mini cloud computing center". For instance, within the context of an airport monitoring system, technology for offloading computations directly onto the MEC server near the airport monitoring equipment can be employed. This server can analyze and filter the video stream before transmitting it to the central server, and thereby decreasing demand for network bandwidth and server processing requirement, and incurring less latency. This approach alleviates the data flow transmitted via the Wide Area Network (WAN), thus substantially reducing the burden on the central server and, subsequently, the central network server's load.

**Table 1**
Comparison of survey papers related to MEC in recent years.

| literature | Year | MEC | Energy consumption minimization | QoS | Time delay minimization | High-computing offloading | Different application scenarios |
|---|---|---|---|---|---|---|---|
| [5] | 2017 | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ |
| [8] | 2019 | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| [9] | 2020 | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| [10] | 2020 | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ |
| [11] | 2020 | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [12] | 2020 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [13] | 2021 | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [14] | 2021 | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ |
| [15] | 2022 | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ |
| [16] | 2022 | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ |
| [17] | 2023 | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Our | 2024 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

✓: Discussed, ✗: Not Discussed.

### *1.4. Significance of offloading in MEC*

MCC initially introduced the concept of computing offloading, whereby intricate computations are executed in the cloud, resulting in the simplification of mobile device functions and a reduction in configuration complexity. Traditional cloud computing encompasses use cases and methodologies for "computing migration" or offloading computations. Various factors, such as the nature of tasks, device specifications, user preferences, and the resource availability of MEC servers, play a crucial role in this process [5]. Currently, MEC leverages offloading technology to efficiently manage user task offloading and server resource allocation, significantly enhancing server task processing efficiency. Consequently, this paper focuses on one of the pivotal technologies in MEC applications-namely, the technology of offloading computational tasks. This technology involves offloading computational tasks from MDs to the edge servers, thereby alleviating the computational burden on the terminal equipment itself. By intelligently utilizing MEC server resources, this approach conserves energy and accelerates computation speeds. In MEC, the challenge of efficient task offloading is multifaceted due to the limited resources available on servers compared to cloud servers [6]. Furthermore, tasks vary in terms of priority, scheduling, computational overhead, and interdependencies. The optimal task offloading strategy necessitates the selection of the most suitable task for offloading to the most appropriate server at the optimal time, all while ensuring optimal application performance and efficient utilization of MEC resources. This issue belongs to the category of NP-hard problems, involving multi-objective optimization [7].

### *1.5. Objectives and scope*

This work presents the recently proposed task-offloading methods in MEC, highlights the research progress of task-offloading algorithms, and outlines the current challenges in MEC.

The main contributions of this paper are as follows:

(1) The development background, basic concepts, and differences between different computing paradigms of MEC are briefly summarized.

(2) This paper summarizes the research progress of computing offloading algorithms in edge computing, studies the architecture design of the MEC system, and provides a typical MEC application scenario to facilitate readers to compare the design of such applications.

(3) The offloading algorithms are qualitatively compared from the aspects of offloading methods, application scenarios, and main ideas, and the algorithms are analyzed and classified.

The reminder of this paper is organized as follows: The Section 2 surveys the recent related works. The Section 3 provide fundamental knowledge of MEC technology. The Section 4 discusses the architectural framework of MEC technology at the system level. Section 4 introduces system model for offloading. Section 5 gives offloading approaches in MEC. Section 6 introduces the classification of mobile edge computing offloading decisions. Section 7 discusses the key findings and future directions. Finally, Section 8 summarizes the survey.

## 2. Related surveys

We have studied and summarized the overview of task offloading of edge computing from six aspects, including MEC, Energy Consumption Minimization, QoS, Time Delay Minimization, High-computing Offloading and different application scenarios. Table 1 shows the comparison of survey papers related to MEC in recent years.

Among them, the main review object of paper [8–10,12] is the task offloading for edge computing. However, paper [5,11,13–17] are the task offloading for mobile edge computing. Paper [5,8–11,14,15] conducted research on minimizing energy consumption. Among them, paper [8,11,14] are for MEC. In terms of QoS, for the review articles on MEC, only paper [8] has discussed it. For the time delay minimization, most review papers on MEC have analyzed and discussed this issue. This indicates the importance of time delay for MEC in task offloading. Relatively speaking, there is less discussion on the remaining two topics, especially regarding the discussion on MEC. There is no relevant review paper to summarize and discuss the different application scenarios.

Mach et al. [5] categorize the research on computation offloading into three main areas: (1) decisions on computation offloading, (2) allocation of computing resources within the MEC, and (3) mobility management. Due to the relatively simple research scenario at that time and the lack of consideration for more complex QoS and different application scenarios, the issues in this part of the scenario were not discussed. Jiang et al. [8] focus on the various task computing offloading strategies. However, the paper mainly discusses issues related to minimizing energy consumption and QoS, without addressing some aspects such as time latency and different application scenarios. Wang et al. [9] present taxonomy of task offloading in edge-cloud environments and discuss the resource task for edge-cloud and multi-cloud scenes. Lin et al. [10] focus on the different offloading modeling methods. However, the paper did not take into account and discuss the issue of time dalay for edge computing. Zheng et al. [12] discuss computation offloading in edge computing including offloading scenarios, influence factors and offloading strategies. However, the paper did not follow the energy consumption, QoS, time delay et al. Mustafa et al. [13] present the survey of MEC offloading techniques with WPT. This paper, like the previous paper, did not discuss the above issues. Qadir et al. [14] analyze and discuss task offloading based on the performance parameters. However, the paper ignores the summary and discussion of QoS and different application scenarios. Shakarami et al. [11] focus on game theory (GT)-based computation offloading mechanisms in the MEC. However, the paper did not analyze or discuss further offloading strategies. Feng et al. [15] summarize various offloading objectives including energy consumption minimization and time delay.

However, there is also a lack of discussion on QoS and task offloading strategies. Huda et al. [16] review computation offloading approaches in UAV-Enabled MEC. For offloading decisions, the paper focuses on the energy/power consumption and time delay. Akhlaqi et al. [17] add algorithm classification, bibliometric analysis in the paper. However, there was no analysis and discussion on task offloading strategies and performance requirements. In our work, we introduce the mobile edge computing fundamentals and system model for offloading. According to the MEC offloading decisions, we summarize and classify the task offloading for MEC from six aspects including MEC, Energy Consumption Minimization, QoS, Time Delay Minimization, High-computing Offloading and different application scenarios. Moreover, we identify research gaps with respect to each aspect and propose future research directions.

## 3. Mobile edge computing fundamentals

What distinguishes MEC from MCC is that it takes the computing power to the near the edge of the network [18]. Applications running on MDs transfer computationally intensive tasks to nearby MEC servers to reduce overhead and further alleviate network congestion [19–21].

### *3.1. Key concepts of mobile edge computing*

In 2014, the concept of the mobile computing paradigm emerged through collaboration between the European Telecommunications Standards Institute (ETSI) and industry leaders like Intel, Nokia, Huawei, IBM, and others. This collaboration set the stage for what we now know as MEC [22,23]. ETSI formally defined MEC as “deployed in the carrier network or its subnet to provide management and host functions for mobile edge applications”, solidifying its essential role in the emerging 5G/6G landscape. Initially, MEC was envisioned as a way to integrate cloud computing technology into cellular networks. However, by 2017, ETSI expanded the concept to encompass various wireless network types, thereby extending its reach beyond 3GPP networks [24]. Today, MEC is recognized as a cloud server designed for deployment at the edge of mobile communication networks, capable of executing specialized tasks. Despite evolving terminology, the industry predominantly still refers to it as ’MEC,’ underscoring its significant presence and importance within the domain of mobile communication networks [25].

### *3.2. Role of 5G in MEC*

5G networks possess three distinct characteristics: Enhanced Mobile Broadband Technology (eMBB), Large-Scale Machine-Type Communication Technology (m-MTC), and Ultra-High Reliability and Low-Latency Communication (uRLLC). The advent of 5G has given rise to a surge in the demand for connecting numerous devices and performing high-speed computations. In fact, it is estimated that by 2023, 5G will support over 10% of all global mobile connections, as per a report [26]. The primary connectivity goals within the EU include ensuring seamless 5G wireless broadband access in urban areas and transportation routes by 2025, and extending 5G coverage to all populated regions by 2030. Official statistics indicate that 5G coverage has presently reached 72% of the EU population [27,28]. Followings are the core features of 5G:

**Ultra-High Bandwidth:** 5G boasts incredibly high data transfer rates, allowing for seamless streaming and downloading of large files.

**Ultra-High Density:** This refers to the ability of 5G networks to accommodate a massive number of devices and connections within a limited area.

**Ultra-Low Delay:** 5G minimizes communication delays, making it ideal for applications where real-time responsiveness is critical. However, amid the exciting possibilities of 5G, there are technical challenges to address. As the number of devices connected to mobile networks increases significantly, ensuring timely data transmission becomes crucial. Two primary approaches can tackle this challenge:

**Strengthening Network Capabilities:** While 5G inherently enhances network capabilities, it is important to acknowledge that network performance can be unpredictable and subject to interference [29]. Therefore, reinforcing network capacity alone may not suffice.

**Front-End Enhancement with MEC:** MEC emerges as a valuable solution. Even though small processors have limited computing power, MEC empowers the front end by bringing computation closer to the edge of the network. This approach mitigates the reliance solely on the network.[1] Cisco’s annual Internet report predicts that, by 2023, Machine-to-Machine (M2M) connections supporting networked applications will make up approximately 50% (14.7 billion) of all global equipment and connections [30]. This underscores the significant role of interconnected devices in the future of communication networks. In this context, the demands of 5G and the IoT form a joint force: on the one hand, the faster communication technologies, such as 5G and WIFI6, can significantly reduce the delay of data transmission, so that the single-hop delay of front-end equipment can be reduced to a single millisecond level. On the other hand, it can also relieve the pressure of cloud computing centers by decentralized processing of massive computing demand of edge devices. On the premise of meeting the existing computing demand, it is bound to spawn all kinds of real-time computing services [31]. 5G/6G and beyond, represents a significant advancement in network capabilities. It is characterized by its ability to efficiently handle substantial volumes of data, offering attributes such as low latency, high reliability, and exceptional speed or bandwidth. However, the efficacy of 5G/6G is contingent upon its integration with edge computing. This integration becomes imperative when a multitude of devices are interconnected within the network, and the data generated by each device inundates the network infrastructure. In such circumstances, the conventional approach of transporting all this data to centralized cloud servers proves to be increasingly impractical and resource-intensive. Moreover, the quintessential promise of 5G/6G, namely the attainment of revolutionary latency levels, necessitates a fundamental shift in data processing. This shift entails the localization of data processing at the network’s edge, ensuring that data is handled in proximity to its source or point of utilization. By adopting this approach, latency can be minimized, as data traversal distances are reduced, thereby aligning with the stringent latency requirements set forth by new technology. In essence, the aggregation of 5G/6G and edge computing emerges as a pivotal paradigm for the realization of its transformative potential, enabling seamless and responsive communication across diverse applications and scenarios.[2]

### *3.3. Comparative analysis of similar paradigms*

In the realm of mobile broadband Internet services and the burgeoning field of mobile IoT, the explosive growth of data applications has posed significant challenges in terms of processing vast amounts of data efficiently. To meet this challenge, a range of innovative computing paradigms has emerged over the past few years. These paradigms aim to ensure that application services are not only more timely and reliable but also resource-efficient and flexible. Some of these groundbreaking computing paradigms include the Transparent Computing Paradigm (TC), Fog Computing Paradigm (FC), Mobile Cloud, Cloudlet (Cloudlet), and Follow ME Cloud. Each of these paradigms comes with its unique set of advantages and disadvantages, making them suitable for various practical scenarios. In the future, as the Internet continues to evolve into an interconnected ecosystem where everything is linked, collaboration among these paradigms will become increasingly vital. People

[1] https://www.ibm.com/cloud/smartpapers/5g-edge-computing/.

[2] https://www2.deloitte.com/us/en/pages/consulting/articles/what-is-5g-edge-computing.html.

**Table 2**
Comparison of similarities and differences of computational paradigm schemes.

| Scheme | Calculate position | Function size | Flexibility |
|---|---|---|---|
| Fog computing | Devices on the route | Maximum | Normal |
| Cloudlet | the terminal device and data center | Small | Relatively worst |
| Cloud computing | On the internet | Normal | Better |
| Mobile edge computing | The edge of the local network | Smallest | Best |

will harness the strengths of these paradigms collectively to navigate the complexities of the connected world. Followings are the three of these paradigms: Fog Computing, Mobile Cloud, and Cloudlet. We will explore their attributes and differences to gain a clearer understanding. For a comprehensive comparison, please refer to Table 2.

### *3.3.1. Fog computing*

Fog computing, introduced as a concept by Cisco in 2012, emerged as a new paradigm. This section explores fog computing and its similarities to MEC. Initially designed for the IoT environment, fog computing brings computational power to the network's edge, and it aims to process data, store it on the client side, and transmit data with minimal volume to the cloud. This approach prioritizes edge device processing, offering advantages like wireless access, mobility support, location awareness, and rapid response when compared to traditional cloud computing. It shares striking similarities with the concept of MEC [30,32,33]. Fog computing was originally conceived for IoT environments. Its primary objective is to provide computational resources at the edge of the network, bringing processing closer to where data is generated [34,35]. Fog technology employs edge devices to process data, store data on the client side, and transmit data with minimal volume to the cloud. In contrast to cloud computing, fog computing emphasizes wireless access, mobile support, location awareness, and rapid response times. Fog computing demonstrates advanced capabilities and superior performance in handling user requests when compared to traditional cloud computing. Cloud computing relies on infrastructure, necessitating hardware and software to manage tasks and processes. Conversely, fog computing leverages the resources available on edge devices, making it a more resource-efficient approach [36].

### *3.3.2. Mobile Cloud Computing(MCC)*

In recent decades, sending and storing large volumes of data in the cloud for real-time analysis and calculation has become a prevailing development trend. This section explores the concept of real-time big data processing in the cloud and its significance. It also discuss the role of cloud computing, the challenges posed by mobile IoT applications, and the emergence of Mobile or MEC architecture in addressing these issues. Cloud computing offers users pay-per-use services and computing resources tailored to their specific requirements [37]. It has become a cornerstone in facilitating real-time data processing and analysis. In the era of mobile IoT, the proliferation of MDs and the surge in mobile data flow have strained the Internet, leading to uncontrollable service delays. Mobile IoT applications are particularly sensitive to time delays and context awareness, making it challenging to meet their service requirements solely through mobile cloud services. MEC architecture represents a fundamental shift from the traditional decentralized cloud computing model. However, it is important to recognize that cloud computing and edge computing are not polar opposites; they complement each other. MEC does not seek to replace cloud computing but rather extends and enhances the capabilities of traditional cloud computing services [38].

### *3.3.3. Cloudlet*

When MCC transfers computing-intensive tasks from MDs to the cloud for execution, the data does not just travel through the wireless network; it also goes through the core network. If there is a lot of data interaction during this upload, it can overload the network and significantly increase transmission delays. Cloudlet [39], a service which falls into the category of edge computing and is essentially another form of edge computing [40]. It is a crucial method and tool for effectively addressing the previously mentioned issue of high concurrency delays. The Cloudlet architecture principle is depicted in Fig. 1(b). In this setup, the cloud computing server is positioned near the user's location, usually just one hop away. When a mobile device directly accesses the service interface of mobile intelligent cloud computing, it does not need to navigate through a complex core network. This approach can be referred to as a "near-end cloud implementation mode". In contrast to traditional cloud computing, Cloudlet uses wireless Wi-Fi connections and offloads resource-intensive cloud computing tasks to smaller local clouds. This approach prolongs the battery life of mobile storage devices. A cloudlet processes most of the data locally, transmitting only a smaller portion to the central cloud. This reduces data storage requirements and bandwidth usage, resulting in reduced communication delays, latency, and jitter for a smoother user experience.

## 4. System model for offloading

The typical deployment mode for MEC is illustrated in Fig. 1(a). In this setup, installing a standard x86 server directly onto the base station allows for the immediate handling of various urgent tasks within the station. However, this approach can consume a substantial amount of system resources. When the base station receives a video signal from a camera, it sends it directly to the MEC server for processing. This enables timely updates and adjustments to the vehicle's current driving conditions based on system rules. Nevertheless, rule updates and remote driving execution occur on a remote server. By deploying the MEC server within the base station, MEC can significantly reduce delays and enhance location awareness by processing user requests at the network's edge. This, in turn, leads to an improved user experience and lessens the burden on the core network, as outlined in Hu et al.'s work [23].

MEC has been developed to provide a multi-tenant hosting environment for applications at the network edge. Its system architecture comprises hardware resources and virtualization infrastructure, including computing power and network resources. MEC supports a variety of applications, such as autonomous driving and car networking. Currently, there is a growing demand for high-speed computing in services like HD live broadcasting, video surveillance, and unmanned aerial vehicles. These services are exceptionally sensitive to bandwidth and latency. To address this challenge and fulfill the increasing computational needs on MDs, the concept of decentralizing cloud computing has emerged as a new solution [41]. MEC leverages cost-effective x86 servers to complement expensive communication components, enhancing the network's ability to efficiently manage computing resources. This approach aims to harness the computing power at the network's edge while minimizing delays, aligning with the demands of modern applications.

### *4.1. Computational model*

It enables low-latency and high-performance processing for applications like augmented reality, autonomous vehicles, and IoT devices. In the context of MEC, offloading involves transferring computational tasks from a central cloud server to edge servers to achieve faster processing. Here is an overview of the computational model for MEC offloading technology:

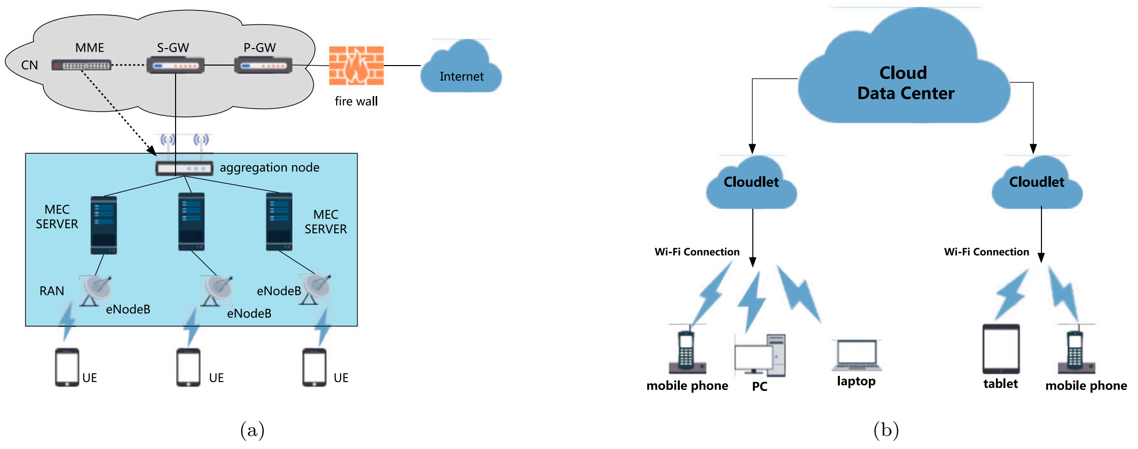


**Fig. 1.** (a) and (b) illustrate the deployment architecture of Cloudlet and Mobile Edge Computing (MEC) systems, highlighting the structural differences and similarities between the two.

- **Edge Servers**: MEC depends on a distributed network of edge servers or nodes positioned at the network's edge, close to end-users or devices. These servers have sufficient computational power and resources to perform real-time processing.
- **Task Offloading**: Computational models for MEC offloading involve deciding whether to execute a task locally on the device or transfer it to an edge server. This decision can be based on factors like task complexity, available network bandwidth, latency requirements, and server load.
- **Edge Computing Frameworks**: Various edge computing frameworks and platforms have emerged to facilitate task offloading. These frameworks provide APIs and tools for developers to deploy and manage applications at the edge. Examples include AWS Greengrass, Azure IoT Edge, and OpenStack Edge Computing.
- **Machine Learning and AI**: Machine learning models and artificial intelligence algorithms are often used in MEC offloading decision-making. These models can analyze data from the device, network conditions, and server capabilities to make intelligent offloading decisions in real-time.
- **Network Slicing**: MEC can involve network slicing, where dedicated virtual network segments are created to prioritize traffic and resources for specific applications or tasks. This ensures that critical tasks get the necessary computational resources.
- **Security and Privacy**: Computational models for MEC offloading need to address security and privacy issues. Data transmitted to edge servers needs to be encrypted, and access control mechanisms must be in place to prevent unauthorized access to sensitive information.
- **Dynamic Adaptation**: MEC offloading is adaptable and can respond to fluctuating network conditions and workloads. Models may need to continuously monitor and adjust offloading decisions to optimize performance. The computational model is discussed in Section 5 in more detail.

## 4.2. Pricing mechanism

To ensure uninterrupted service, edge nodes must pre-install frequently requested service programs [42,43]. This is because the time it takes to install and load these programs can be several tens of seconds, in contrast to their execution time, which is in milliseconds [44]. These commonly requested programs are pre-cached on the edge node servers to enable real-time service delivery. However, given that edge servers have limited resources, they need to prioritize and cache the most frequently requested service programs to ensure efficient operation. Programs that are not cached result in the inability of edge nodes to provide real-time services, a process referred to as service caching [45, 46]. For instance, a program for human face authentication, used in payment or security verification, must be cached by the edge servers. MEC servers determine their pricing based on their computational and storage capabilities. However, their pricing strategy is not solely dependent on their own resources but also takes into account how other users decide to utilize their service. For instance, if there is a high demand for a popular service program, the MEC server may increase its price. Conversely, if the price becomes excessively high and other MEC servers are available, it might lose requests. Therefore, to optimize resource utilization and maximize profits, the pricing mechanism relies on algorithmic game theory. Specifically, it is modeled as a two-stage dynamic game with incomplete information, often referred to as a Stackelberg game. In this game, MEC servers play the role of leaders, while requesting devices act as followers.

### 4.2.1. Two-stage stackelberg game model

In a Stackelberg game, edge computing service provider (ESP) acts as a leader, while service requesting users (referred to as followers) respond with their moves.[3] Likewise, in the context of MEC, the MEC server determines the pricing for its computing resources. This pricing is influenced by the server's own computing resources and the current utilization rate. Subsequently, the MEC server broadcasts this pricing to the network. In the second stage of this process, the applicants must decide which edge to offload their tasks to and purchase the necessary resources. This decision takes into account not only the pricing but also factors such as the amount of data that needs to be offloaded [47].

When edge servers offer their pricing they broadcast in the network. Based on the received pricing the requesting devices compete for the computational resources such as to minimize their own cost. A requesting device is unaware of how many other devices are going to offload its task on a given server which is interesting about this game. Utilizing this phenomenon about prediction at both sides, i.e. the server and receiver both need to make intelligent decisions to maximize their own profit. As the requesting device does not know about the other devices' requesting program, it makes a suitable choice for the Stackelberg game because there is missing information.

### 4.2.2. Pricing types

Two pricing strategies are adopted based on MEC servers namely **uniform** and **differential**[45]. In a uniform strategy, all the service program from a given server is offered at the same price. On the other hand differential pricing assumes that the MEC servers are set differently based on the demand, computational and storage, and many more requirements.

[3] https://web.stanford.edu/~rjohari/teaching/notes/246_lecture7_2007.pdf.

## 5. Offloading technologies in mobile edge computing

MEC is a crucial element in 5G/6G technology, aimed at deploying server devices at the edge of the mobile network. This setup enables quick responses to computing, storage, and service requests from intelligent devices through the radio access network (RAN) [48]. This approach effectively minimizes service delays and bolsters network bandwidth [49]. Contrasting with the vast storage and centralized computing resources found in the cloud computing paradigm, MEC operates with limited server-based storage and computing resources. Consequently, the efficient scheduling of task offloading and allocating resources emerges as a critical challenge in MEC. These aspects significantly contribute to enhancing MEC functionality, underscoring the significance of research into computational offloading technology [11]. As the backbone of the MEC computing paradigm, offloading technology plays a pivotal role in advancing IoT technology. One of the primary research objectives revolves around making informed decisions regarding task offloading to MEC servers. These decisions take into account the task's characteristics within the MEC computing paradigm. Such decision-making becomes particularly vital when terminal devices exhibit high energy consumption requirements or face storage constraints that hinder task completion [50]. Consider, for example, the 5G communication network, where a multitude of terminal devices, including smartphones and IoT devices in smart homes, need to offload their tasks. However, as the number of tasks offloaded onto MDs gradually increases, it can surpass the processing capacity of the MEC server, leading to unmet task requests and subsequently extended task execution times due to queuing scenarios. Additionally, despite the MEC server's proximity to end-users compared to remote cloud centers, the network access mode of RAN raises two common considerations: (1) the congestion caused by the interference of the actual communication links, and (2) task transmission time by the extra delay offloading execution. Considering the total transmission delay of tasks, key factors influencing the quality of communication links are signal-to-noise ratio, co-channel interference, etc. For various edge environments, Shannon equation is generally used to predict the reliable transmission rate of the channel. The task upload rate is as follows:

$$R_j = B\log 2\left(1+\frac{q_j g_j}{N+\sum i q_i g_i}\right) \tag{1}$$

In Eq. (1), the variables are defined as follows: $B$ represents the user bandwidth, $N$ represents ambient noise, $q$ denotes user transmission power, and $g$ represents channel gain. Given that user bandwidth is typically fixed, especially in scenarios involving multiple users, increasing the transmission power primarily serves to enhance the transmission rate. Consequently, the energy consumption for task offloading is expressed as follows:

$$\mathrm{E_t} = q * \frac{Input}{R} \tag{2}$$

Input represents the task offloading amount and $\frac{Input}{R}$ denotes the task offloading delay. When tasks are offloaded to edge servers, there are usually far fewer servers than edge users. The processing delay of a single task is modeled as follows:

$$\mathrm{T} = \frac{\mathrm{C}}{R} \tag{3}$$

In this context, $C$ represents the number of CPU cycles needed for task processing, while $R$ denotes the computing resources allocated by the edge server to users, typically expressed as CPU cycles per unit time. When the local device's task execution time is significantly shorter than the combined communication and server execution times, computational offloading is not an appropriate solution. Moreover, in domains like autonomous driving and the Internet of Vehicles, data frequently updates during task execution, with constant communication base station switches. High delays could render computing task results outdated. For instance, let us consider the 0–1 offloading mode in a single-user scenario, where the focus lies on time delay, energy consumption, and Quality of Experience (QoE) for users [13]. In this scenario, the focus is on offloading tasks to the edge network and achieving optimal processing. Deciding whether to offload tasks locally or to the server, as per Purdue University's task offloading guidelines [51], also takes into account considerations such as transmission energy consumption or restrictions related to wireless transmission energy.

On the graph, the horizontal axis represents ascending calculation complexity from left to right, while the vertical axis illustrates the quality of the wireless communication channel. The coordinate space depicts the task selection for offloading. When the energy consumption and time delay considered together, higher computational costs correlate with a higher likelihood of task execution via offloading. However, as the communication channel quality worsens, the processing time delay increases, making local task processing more favorable. The white section in the figure represents a situation between these extremes, and the decision largely hinges on the bandwidth allocated to the task, as follows:

$$\frac{\mathfrak{I}}{\mathrm{fm}} > \frac{y}{Y} + \frac{\mathfrak{I}}{fs} \tag{4}$$

where $\mathfrak{I}$ denotes the computational task cost (complexity), $f_m$ represents the capability of MDs to process tasks locally, while $y$ signifies the volume of data requiring offloading during task uploads. Additionally, $f_s$ signifies the capacity of servers to handle offloaded tasks.

For instance, in Eq. (4), user delay serves as the benchmark for measuring computational offloading capability. In this scenario, the efficiency of MDs in handling tasks locally is higher. Considering MD energy consumption, the calculation method is as follows:

$$pm * \frac{\mathfrak{I}}{fm} > pt * \frac{y}{Y} + pi * \frac{\mathfrak{I}}{fs} \tag{5}$$

where, $p_m$ signifies MDs' capability to handle tasks locally, $p_t$ represents local transmission energy consumption, and $p_i$ indicates the minimum energy consumption required to keep the equipment operational while waiting for task processing and reception, as detailed in Eq. (5). The previous work did not account for the time-varying nature of the communication channel in calculations. However, in real-world environments, communication channel conditions can fluctuate [52]. Consequently, considering task types and communication networks, it becomes essential to formulate optimal strategies for MEC server task offloading, aiming to enhance user equipment service quality and maximize network-wide benefits.

### 5.1. Task offloading technology overview

The main research goal of MEC is to facilitate the use of extended cloud computing services at the network's edge for MDs with limited resources. Given the constraints posed by terminal equipment energy consumption and computing resources, user devices can offload certain tasks to the MEC server. This task offloading not only reduces computation time, enhancing Quality of Experience (QoE), but also maximizes revenue for network operators and service providers by reducing mobile device energy consumption and extending their lifespan. This is especially crucial for emerging mobile applications that demand low latency and intensive computation. Within the MEC literature, researchers focus on these individual metrics or collectively consider them as indicators of multi-objective challenges. Furthermore, various researchers employ different methods to evaluate task offloading. In this context, our paper concentrates on examining different offloading scenarios, methods, and optimizing the indicators for task offloading.

### 5.2. Background knowledge of offloading algorithms

#### 5.2.1. Markov decision processes and Bellman equation

A Markov Decision Process (MDP) is a stochastic model that encompasses a series of events. The current state of events is independent of the prior sequence of states the process has undergone, forming the fundamental basis of the reinforcement learning process. The Bellman equation is frequently employed to resolve recursive MDP problems [53]. Presently, the Bellman equation features two formulations: Value Iteration and Policy Iteration. These methods involve techniques like dynamic programming and linear programming, and the respective iteration equations are as follows:

$$V\pi(s) = Rs + \max_{a} \gamma \sum_{s'} Psas' V\pi(s') \tag{6}$$

$$\pi(S) = \arg\max_{a} \sum_{s'} Psas' V\pi(s') \tag{7}$$

In current research and analysis, optimization problems are typically converted into calculation offloading schemes using deep reinforcement learning through the Markov decision method. The two methods based on Bellman equations can be more suitable for specific scenarios when solving MDPs.

#### 5.2.2. Q-learning [1]

In the Q-learning-based task offloading algorithm, the state of the next task processed by the server is related to the current state information, but it is independent of earlier states. Typically, it employs time difference learning (TD Learning) for estimating the problem's end, and the iterative equation of the algorithm is given as Eq. (8). Because the system adheres to the Markov property, the TD algorithm converges when the condition of an absolute decrease in $\alpha$ is met. Specifically, *e(s)* denotes the state $S$'s voting degree. In practice, you can calculate it using Eq. (9), where when $s = s_k$, $\delta_{s,s_k}$ equals 1. This implies that the number of visits to state $S$ influences the current reward value, and its value function is iteratively adjusted as per Eq. (8).

$$\mathrm{V}(s) = V(s) + \partial(rt + 1 + \gamma V(st + 1) - V(s))e(\mathrm{s}) \tag{8}$$

$$\mathrm{e}(s) = \sum_{k=1}^{t} (\gamma y)^{t-k} \delta s, sk \tag{9}$$

The learning estimation aspect of Q-learning closely resembles that of State Action Reward State Action (SARSA), involving the iterative update of the Q function at each step. The model is defined by four tuples, and Q-learning utilizes these four tuples (state, action, reward, next-state) to estimate the Q function. As outlined in equation Eq. (10), where $s$ represents states, $a$ represents actions, and $\gamma$ denotes the discount coefficient within the range [0,1], $s$ and $a$ describe the current state behavior, while $\widetilde{s}$,$\widetilde{a}$ characterize the behavior of the next state. It is evident that in this decision-making process, only the preceding measurement influences the offloading decision.

$$\mathrm{Q}(s, a) = R(s, a) + \gamma * \max \widetilde{a}\{Q(\widetilde{s}, \widetilde{a})\} \tag{10}$$

#### 5.2.3. DQN

In task offloading, the algorithm must choose the offloading location, access location, and cache location to minimize the task processing cost. In this algorithm, a neural network utilizes DQN (Deep Q Network) within reinforcement learning to expedite the selection process based on the server and access point's current state. Contrasting with the standard Q-learning algorithm, a significant modification involves the incorporation of a novel "Target-Q-Network". The comprehensive equation can be found in the following expression:

$$Q(\mathrm{s}, a) = (1 - \partial)Q(s, a) + \partial(r + \gamma \max_{a'} Q(s', a')) \tag{11}$$

The equation reveals that the $Q$ function *Q(s,a)* consists of two parts. The first part, represented as (1-$a$)$Q$($s$, $a$), primarily serves to retrieve the current state–action value and update its value (utilizing $Q$ as follows: *Q(s,a)* = (1-a)Q(s, a) + ⋯). In contrast, the second part involves the basic Q-learning algorithm, where these Q functions (networks) are generally identical.

## 6. Classification of mobile edge computing offloading decisions

This section presents an overview of the literature regarding task offloading algorithms, categorizing articles into five distinct groups based on varying research objectives. This section primarily elaborates on and discusses five key aspects: attaining a balance between energy consumption and time delays(QoS), reducing energy consumption, minimizing time delays, high-computing offloading, and different application scenarios.

### 6.1. Joint optimization delay and energy consumption

This section summarizes key studies in MEC task offloading that focus on balancing energy consumption and time delay. A concise overview of their distinctive features can be found in Tables 3, 4, with further elaboration provided below.

- The deep learning literature, as presented in [55], introduces a partial offloading scheme rooted in deep learning principles. Its main goal is to reduce both delay costs and energy consumption. To be more precise, this paper delves into the realm of wireless power-supplied MEC systems, comprising MDs and hybrid access points (HAPs) integrated with MEC infrastructure. It explores the reduction of costs and energy consumption in MDs by harnessing trained Deep Neural Networks (DNNs) [110].
- **Improved NSGA-II:** In the research presented in [56], a task scheduling and offloading strategy is introduced for a multi-user MEC system. Here, computational offloading is treated as a multi-objective optimization problem (CMOP) encompassing processing delay, transmission delay, energy consumption, and scheduling constraints. To address this challenge, the author divides the computational task into multiple subtasks and examines the system overhead, resulting in the proposal of an enhanced NSGA-II algorithm.
- **Multilateral collaborative computing:** Li et al.'s study [57] delves into the realm of integrated task migration for computing offloading within the MEC environment. Traditional MEC architectures often underutilize the idle computing resources of off-site edge servers. In response, this research introduces a collaborative computing offloading strategy to maximize resource utilization. Furthermore, considering the challenge of service migration due to user mobility in the MEC environment, the study also presents a corresponding migration strategy to address this issue.
- **Adaptive task offloading:** In the context of network scenarios, MEC offers vehicles the opportunity to access nearby network resources and computing power, effectively meeting the growing demand for on-board services. Tang L et al.'s research [58] centers on the challenge of task offloading within vehicle-mounted edge computing environments, where the high mobility of vehicles and significant task offloading in the Internet of Vehicles add complexity. Notably, the study aims to ensure that task execution remains uninterrupted even as vehicles are in continuous motion during the offloading process. To achieve this goal, the authors conduct a comprehensive analysis, considering various influencing factors, including vehicle speed, the load on access points within the community, and the load on MEC servers. This analysis informs the optimization of the next offloading strategy, which can be applied to automate network access selection and vehicle edge computing task offloading decisions.

**Table 3**
Summary of algorithms for energy consumption and time delay.

| Reference | Performance parameter | Objective | Advantage | Limitations |
|---|---|---|---|---|
| [54–69] | Balance energy consumption and delay | Solve the relationship between energy consumption and time delay | Solving the relationship between energy consumption and time delay | Not suitable for applications with strict requirements on time delay or energy consumption |
| [70–75] | Minimizing energy consumption | Reduce the energy consumption in the calculation process | Energy consumption of offloading process is minimal | Saving transmission energy consumption may reduce the transmission power of communication equipment |
| [76–84] | Minimizing delay | Reduce the calculation time | Conducive to time-sensitive applications | May cause greater energy consumption or reduce the overall offloading capacity |
| [85–88] | High Computation offloading | Maximize the offloading capacity | Effectively meeting compute-intensive tasks | Generally, energy consumption can be taken into account |
| [89–91] | Multi-task offloading | Meet the requirements of multi-task offloading | Performing multiple uninstallation tasks at the same time | May cause a large time delay |

**Table 4**
MEC task uploading schemes for balancing energy consumption and time delay.

| Algorithm | Limitations | Year | Literature |
|---|---|---|---|
| MCE-GA | Virtual environment | 2021 | [54] |
| DL | Virtual environment and secure | 2021 | [55] |
| Improved NSGA-II | Virtual environment | 2021 | [56] |
| Multilateral collaborative computing | Ignores the competition for resources | 2021 | [57] |
| Adaptive task offloading strategy | N/A | 2021 | [58] |
| DDL-CORA | N/A | 2022 | [60] |
| MAPE-K | Ignores the mobility of the device | 2021 | [61] |
| PSO1 | Weak system model | 2022 | [62] |
| ACDQN | N/A | 2021 | [65] |
| DCO | Low packet loss rate | 2021 | [66] |
| DORS | N/A | 2021 | [67] |
| EDLO | Simple task dependency model | 2022 | [68] |
| DRL | Constrained UAV cooperation capability | 2021 | [69] |
| MGW | Only a few variable system parameters | 2021 | [92] |
| Mukherjee M | Ignores specific service requirements | 2020 | [93] |
| DCOGA | virtual environment | 2021 | [94] |
| DOCRRL | Ignores the competition for resources | 2021 | [95] |
| PSO | Weak system model | 2021 | [96] |
| Li Q | Uncertainty of tasks and networks | 2021 | [97] |
| Attention-Based DDQN | Uncertainty of tasks and networks | 2021 | [98] |
| Lai SW | Simple task dependency model | 2021 | [99] |
| Feng GS | Simple system model | 2021 | [100] |
| MARL | Virtual environment and security | 2021 | [101] |
| DMRO | Ignores the competition for resources | 2021 | [102] |
| HAGP | Ignores the competition for resources | 2021 | [103] |
| Abbas A | Virtual environment | 2021 | [104] |
| QUARTER | Virtual environment in collaborative computing | 2021 | [105] |
| TADPG | The scene studied is simple | 2021 | [106] |
| SCA-based | The UAV scene studied is simple and virtual environment | 2020 | [107] |
| MEC-Agent | The UAV scene studied is simple and virtual environment | 2020 | [108] |
| JOME | N/A | 2023 | [109] |

- **DDL-CORA:** Wang et al. introduced a distributed deep learning-based computational offloading algorithm in their work [60]. This algorithm employs multiple parallel deep neural networks (DNNs) to make optimal offloading decisions. Their study focuses on software-defined mobile edge computing (SD-MEC) within the IoT context, and it involves the construction of a utility function that combines weighted delay and power considerations. This utility function is designed using multiple parallel DNNs.
- **MAPE-K:** Shakarami et al. presented an autonomous computing offloading framework in their work [61], employing the MAPE-K cycle method. This approach effectively reduces the complexity of offloading decision problems by utilizing a combination of deep neural networks (DNNs) and multiple regression models. However, it is worth noting that the paper does not account for the mobility of equipment in real-world applications.
- **PSo1:** Building upon enhancements and optimizations to the particle swarm optimization (PSO) algorithm, Li et al. [62] introduce an offloading strategy tailored for IoT edge computing. This strategy involves the development of a customized particle swarm optimization approach. It encompasses the design, encoding, and calculation of fitness values for particles, facilitating the dynamic updating of particle positions. It is found that compared with the two algorithms in paper [63,64], the average time delay of task execution of this method is lower in different situations.
- **ACDQN:** In their study [65], the authors aimed to jointly optimize partial computation offloading and communication resources within an MEC system. They introduced a low-complexity Deep Reinforcement Learning (DRL) algorithm named ACDQN. This algorithm effectively mitigates computational overhead in NOMA-MEC networks. ACDQN combines actor-critic and deep Q-network methods to streamline its complexity. Furthermore, the authors explored the benefits of merging NOMA and orthogonal multiple access (OMA). Simulation experiments demonstrate the algorithm’s effectiveness when compared to traditional schemes like full offloading in NOMA, random offloading in NOMA, and global execution.

- **DCO:** In the examination of the MIMO MEC system [66], the paper delves into the comprehensive joint optimization of the offloading ratio, transmission covariance matrix, and CPU cycle frequency. It introduces a dynamic computational offloading (DCO) algorithm tailored to the specific needs of the MIMO MEC-EH system.
- **DORS:** In Zhao et al.'s work [67], the focus shifts to task offloading and resource allocation within multi-user scenarios, particularly those involving energy harvesting (EH) equipment. The study takes into account energy consumption, computing resources, and system performance, framing them as a minimum cost problem. To address this, the authors propose an online optimization algorithm known as DORS, founded on Lyapunov optimization theory. This approach dynamically adapts to changes within the MEC system.
- **EDLO:** Cheng et al. [68] explore task offloading strategies in the context of large-scale WDs. They introduce the EDLO algorithm, a deep learning-based offloading strategy designed to minimize both energy consumption and delay. Numerical experiments reveal that when compared to TLP, TEP, and ROS, the proposed algorithm yields lower final costs and superior performance.
- **DRL:** In [69], the advantages of unmanned aerial vehicles (UAVs) in terms of easy deployment and flexibility are explored. This paper focuses on the multi-user UAV-assisted MEC network, aiming to optimize both computation delay and energy consumption. To achieve this, the authors introduce a deep reinforcement learning model that seamlessly learns and optimizes task offloading and UAV trajectory control in an end-to-end manner. It is worth noting, however, that the UAV-assisted MEC system under consideration in this paper does not incorporate UAV cooperation in task division.
- **MGW:** In the study presented in [92], the authors delve into the Cloudlet-deployment problem, with the primary goal of minimizing system energy consumption, delay, and the number of cloud resources used. To address this multi-objective optimization problem, they develop an offloading strategy tailored to optimize the local cloud deployment site. This problem is formulated as a mixed-integer nonlinear programming (MINP).
- **Mukherjee M:** Mukherjee et al. [93] introduce a distributed deep neural network-based offloading strategy. Within this DNN model, a training instance is associated with multiple DNNs, and the authors focus on verifying the minimum loss. Additionally, they expedite the training convergence rate through quadratic constraint linear programming (QCLP) and subsequently optimize the offloading decisions. When compared to baseline schemes, this approach takes into account numerous system parameters and consistently demonstrates superior performance.
- **DCOGA:** In the work by Li et al. [94], a task weight cost model is constructed, taking into consideration energy consumption and delay factors. This model transforms into an optimization problem, which the algorithm is designed to solve. Consequently, they introduce a dynamic computation offloading method termed DCOGA. Furthermore, the authors propose a resource allocation model grounded in utility maximization, with the aim of maximizing both user and edge node utility.
- **DOCRRL:** In the context of task uninstall security for WDs, as discussed in [95], several challenges are considered, including channel time variability, load randomness, and privacy protection. These aspects are modeled as a joint optimization problem encompassing delay, energy consumption, and channel resource allocation, which takes the form of a Markov Decision Process (MDP). In this scenario, where both channel state and task load are random, the authors emphasize the need to fully consider MD task characteristics during offloading. Additionally, they propose a deep reinforcement learning algorithm called DOCRRL, which focuses on partial offloading and resource allocation, facilitating the exploration and learning of optimal decisions without prior knowledge.
- **PSO:** Similar to the improvements made to PSO by Li et al. [62, 96] addresses task offloading with strict delay requirements within the context of time-industrial IoT. To balance energy consumption and delay, the paper introduces a penalty function designed to reduce queuing delays for tasks. Although experimental results show that the PSO strategy is effective, choosing the right parameters for various problem instances can be challenging.
- **Li Q:** Addressing the Quality of Service (QoS) challenge in task offloading, as outlined in [97], this paper extends its focus to energy conservation alongside low latency. It introduces a QoS-guaranteed task offloading strategy rooted in statistical analysis. This approach accounts for the unpredictability of task arrivals and establishes a threshold for task processing delay within the system, allowing for delays beyond this threshold when necessary. In terms of energy conservation, the system can curtail energy consumption by relaxing the QoS requirement. Numerical experiments affirm the superiority of this algorithm over three baseline algorithms.
- **Attention-Based DDQN:** The paper [98] explores task offloading within a multi-base station MEC system characterized by an ultra-dense network. Leveraging reinforcement learning, the problem is transformed into a Markov Decision Process (MDP). The authors design a dual-depth Q network (DDQN) method incorporating an attention mechanism, enabling rapid convergence.
- **Lai SW:** Introducing a task assistance framework with a security focus, as presented in [99], this framework aims to optimize bandwidth allocation and transmitting power offloading rates to enhance performance. Deep learning algorithms are employed, leading to the proposal of a DQN-based offloading strategy for automated multi-objective optimization problem-solving. Experimental results demonstrate the algorithm's capacity to consider security aspects while reducing system overhead.
- **Feng GS:** In the context of task uninstallation within mobile ad hoc networks (MANET), discussed in [100], the paper employs UAVs as MEC servers. It formulates a joint optimization problem encompassing transmission paths, task partitioning, and uninstallation locations. To tackle this challenge, the authors account for the motion state of MANET and employ distinct strategies for dynamic and static MANET scenarios.
- **MARL:** The technology of Multi-Agent Reinforcement Learning (MARL), as proposed in by Sacco et al. [101], addresses dynamic network states to optimize transmission efficiency. Within this system, every node shares the overall state of the network and continually learns the most suitable actions from its environment. This cooperative approach leads to the development of the offloading decision discussed in the paper.
- **DMRO:** In the context of dynamic environments, Qu et al. [102] explore the task offloading problem, specifically focusing on fine-grained task offloading. The author combines deep learning and meta-learning techniques to propose an offloading algorithm based on Deep Meta-Reinforcement Learning (DMRO). This algorithm effectively adapts to various environmental conditions.
- **HAGP:** In the paper authored by Guo et al. [103], the authors investigate the enhancement of task offloading reliability within the industrial internet. They assess the reliability of MDs based on residual energy levels after task completion and introduce a heuristic algorithm employing a greedy strategy (HAGP) for offloading decisions. This algorithm optimally determines the offloading for each MD, considering channel constraints. Numerical experiments demonstrate its superiority over several baseline algorithms.
- **Abbas A:** The study conducted by Abbas et al. [104] focuses on the task offloading problem in multi-objective optimization, aiming to strike a balance between response time and energy consumption. The paper explores meta-heuristic task offloading algorithms and compares the performance of the Ant Colony

**Table 5**
MEC task uploading schemes for reducing energy consumption.

| Algorithm | Limitations | Year | Literature |
|---|---|---|---|
| CMOP | Virtual environment in collaborative computing,not secure | 2021 | [71] |
| CG | Simple task dependency model | 2022 | [72] |
| SRSO | Virtual environment and ignores the competition for resources | 2022 | [112] |
| End-to-end DRL | Tasks cannot be arbitrarily partitioned and offloaded | 2021 | [73] |
| DDQN | Simple task dependency model | 2021 | [74] |
| RAPD | Virtual environment in collaborative computing, not security | 2021 | [75] |
| Sun YY | Output instability | 2021 | [113] |
| RSU-assisted algorithm | Virtual environment, weak system model | 2021 | [114] |
| DRTO | Virtual environment in collaborative computing, not security | 2021 | [115] |
| EFDOT | Single MD network needs to be extended | 2021 | [116] |
| PTAS | Virtual environment with user mobility | 2021 | [117] |
| Ali A | Virtual environment in collaborative computing, not secure | 2021 | [118] |
| Chen X | Virtual environment and ignores the competition for resources | 2021 | [119] |
| QIAC | N/A | 2021 | [120] |
| Wang F | Just a single user scenario | 2020 | [121] |
| Xu C | Uncertainty of tasks and networks | 2020 | [122] |
| EEDOS | Limited to single-user scenarios | 2019 | [123] |
| LiMPO | Without mobility-aware task/VM migration in MEC | 2023 | [124] |

Optimization (ACO), Whale Optimization Algorithm (WOA), and Grey Wolf Optimization (GWO) algorithms in optimizing energy consumption and delay. Simulation results indicate that the Grey Wolf Optimization algorithm outperforms the others.

- **QUARTER:** In the study of Space-air-ground-integrated power IoT (SAG-PIoT) presented in [105], the task can be processed either locally on the device (Plot) or offloaded to Edge Computers (ECs) using Unmanned Aerial Vehicles (UAVs). To tackle the stochastic nature of this problem, Lyapunov optimization decomposes it, and the solution is found through Lagrange dual decomposition, an actor-critic-based task offloading approach, and a low-complexity algorithm based on greediness [111]. Additionally, the QUARTER method is introduced to address the dimension disaster problem.
- **TADPG:** Addressing the slow convergence and instability in model training associated with Deep Reinforcement Learning (DRL) methods, the author explores the task offloading problem in a single-cell multi-Mobile Device (MD) MEC system in [106]. They propose the Time Attention Deterministic Policy Gradient (TADPG) method and compare it with four other DDPG variants, demonstrating the algorithm's superior convergence properties.
- **SCA-based:** In the work by Yu et al. [107], the task offloading of a UAV-MEC system is discussed, focusing on a cooperative UAV-Edge Cloud (EC) offloading scheme involving IoT devices, UAVs, and edge servers. This joint scheme aims to minimize system delay and UAV energy consumption, considering the service delay of all IoT devices. The paper transforms this into an optimization problem, proving its high non-convexity. To address this, the author proposes a Successive Convex Approximation (SCA) algorithm. Numerical experiments demonstrate that this offloading scheme, accounting for UAVs, IoT equipment, and ECs, outperforms the baseline scheme relying solely on local MEC offloading. However, it is worth noting that the single-hop, single-UAV scenario considered may not be entirely realistic.
- **MEC-Agent:** Wang et al. [108], the focus is on the UAV-assisted Mobile Edge Computing (UMEC) system. The paper addresses the task offloading of MDs, UAVs, and edge cloud servers, particularly in situations involving unreliable requests for computing resources due to information asymmetry between MDs and edge servers. Specifically, the authors improve the reliability of task offloading by introducing agent into the framework of task offloading system to model the position and resource request of an MD.
- **JOME:** Huang et al. [109] propose a novel MEC protocol and optimize the operation mode selection and resource allocation in each mode with a task execution delay constraint.

### 6.2. Energy consumption reduce as the goal

This section focus on the research work aiming at reducing energy consumption in MEC task offloading. Characteristics of the papers on this subject are described in detail below and are summarized in Table 5.

- **CMOP:** Peng et al. [71] address the optimization of UAV flight safety and offloading from a multi-objective perspective. Currently, most research focuses on optimizing UAV offloading based on a single criterion, overlooking the essential considerations of UAV flight safety. Consequently, this paper introduces a Constrained Multi-Objective Optimization Problem (CMOP) incorporating two objective functions: energy-efficient offloading and safe path planning. It also presents a multi-objective evolutionary algorithm designed to tackle this complex problem.
- **CG:** In contrast to the assumption that MEC systems are influenced by various uncertainties, the work by Ji et al. [72] addresses uncertainties in real MEC systems. They start by constraining extreme event probabilities using extreme value theory. When dealing with time-sensitive applications, they formulate the expected worst-case energy consumption. Additionally, the authors introduce a-bounded approximation algorithm based on column generation (CG) technology. This algorithm identifies the b-bounded approximation for offloading strategies, and it reaches an optimal solution when b equals zero. To validate their approach, the authors conducted experiments by running a real mobile application on the Android platform. The results confirm that explicitly considering uncertainty in practical applications leads to lower energy consumption, outperforming the current scheme in terms of energy savings.
- **SRSO:** In the context of the MEC remote relay assistance system with relay nodes (RNs), the work by Chen et al. [112] takes a comprehensive approach. Unlike previous works that solely consider RNs, this study jointly considers RNs, Remote Servers (RS), and local execution to optimize the computational load of relay nodes. The proposed Smart Relay Selection Optimization (SRSO) strategy, which utilizes Dynamic Programming (DP) technology, outperforms other benchmark strategies in optimizing energy consumption.
- **End-to-end DRL:** In their work, Ale et al. [73] tackle the challenge of computing offloads in dynamic MEC networks. Their goal is to maximize timely task completion while also minimizing energy consumption. To achieve this, they optimize both the selection of edge servers and the allocation of computing power for offloading. The authors introduce an end-to-end Deep Reinforcement Learning (end-to-end DRL) algorithm. This model takes

input data from the MEC network as observations, and the DRL model generates control parameters. To ensure the model's stability during training, they employ experience buffer playback and editing techniques. It is worth noting that, during the offloading process, tasks cannot be arbitrarily divided and offloaded to edge servers.

- **DDQN:** In order to reduce energy consumption in the MEC system and develop a unified strategy for computing offloading and resource allocation, Zhou et al. [74] propose a Q-Learning reinforcement learning algorithm based on value iteration. They further introduce a method based on the Double-Depth Q-Network (DDQN) to reduce dimensionality. This approach effectively approximates the value function used in Q-learning. Simulation experiments demonstrate that the algorithm's performance closely matches that of exhaustive methods. With verified parameters, this method yields results similar to offloading decisions.
- **RAPD:** In the context of MEC networks, smart MDs often offload resource-intensive tasks to reduce processing time and battery consumption. The paper [75] delves into the intricacies of multi-server MEC networks, which consist of multiple base stations. It addresses the challenge of offloading calculation and resource allocation, especially when multiple devices compete for limited resources. The authors prioritize task offloading by combining strategies to minimize overall energy consumption and a penalty function. They employ a weighted aggregation method to convert this issue into a single-objective problem, despite its high complexity. The paper also presents a heuristic solution approach known as Resource Allocation for Priority Devices (RAPD).
- **Sun YY:** To combat energy shortages in sensor nodes within MEC networks, Sun et al. [113] proposes a task offloading method that incorporates energy collection. This method effectively extends the lifespan of equipment.
- **RSU-assisted algorithm:** Shang et al. in 2021 [114] explores the computational offloading of Vehicular Edge Computing (VEC) systems with a focus on efficiency and energy savings. To enable fine-grained computational offloading, the authors utilize Roadside Units (RSUs) to offload tasks to VEC. They employ deep learning techniques to optimize user delay, transmission power, and Edge Cloud (EC) computing capacity constraints, ultimately deriving an energy-saving convergence algorithm. However, it is important to mention that the paper lacks a thorough analysis of the algorithm's convergence and complexity.
- **DRTO:** Zhu et al. in 2021 [115] introduces a novel approach that utilizes satellite links to aid urban terrestrial clouds (TC) in the task offloading process. In this scenario, offloading decisions are made based on predefined user models, and the authors design an algorithm employing deep learning techniques. This algorithm dynamically updates the offloading strategy in response to the changing channel conditions. Experimental results show that the algorithm can effectively handle real-time offloading even in fading channels.
- **EFDOT:** Ali et al. [116], the focus is on UAV task offloading. The optimization objectives in this research include user scheduling, communication channel management, and resource allocation within UAVs. Notably, the authors account for the time-varying nature of the communication channel during task offloading. To address this non-convex problem, they employ a Markov Decision Process (MDP) approach and further propose a task offloading algorithm based on the Depth Deterministic Strategy Gradient (DDPG) method. The objective is to reduce task delay.
- **PTAS:** In contrast to the task allocation problem where the objective is minimizing total energy consumption, Liu et al. [117] focus on the task offloading of heterogeneous resources. Unlike partitioning tasks optimally, this study involves offloading tasks to different servers to meet varying resource requirements. The paper introduces heuristic methods and a series of PTAS algorithms to address this problem.
- **Ali :** The scheduling algorithm discussed by Ali et al. [118] is specifically designed for performing offloading calculations in the cloud. This algorithm aims to make offloading decisions that execute calculation tasks with lower energy consumption.
- **Chen X:** Chen et al. [119], the topic of task offloading in Augmented Reality (AR) is explored, considering both single MEC and multi-MEC systems. The authors' optimization objectives include minimizing system delay and energy consumption while adhering to resource constraints. They model the AR application task offloading as a directed acyclic graph. Given the unpredictable nature of the scheduling channel in this environment, the authors devise a multi-agent deep decision strategy gradient using a framework known as Multi-Agent Deep Deterministic Policy Gradient (MADDPG).
- **QIAC:** In the context of the Virtual Reality (VR) assisted Wireless Virtual Medical Treatment (WVMT) system, Lin et al. [120] delve into the task offloading within a blockchain scenario. Blockchain technology is leveraged for globally sharing task offloading and data processing information. To ensure blockchain consistency and account for channel conditions, the task is transformed into a Markov Decision Process (MDP) problem. The authors introduce a Collective Reinforcement Learning (CRL) algorithm, which uses the Actor-Critic approach to enable adaptive resource allocation within a blockchain-based system.
- **Wang F:** Wang et al. [121] investigate the problem of single-user task offloading within a MEC system powered by wireless energy supply. The study takes into account the impact of channel fluctuations and task causality constraints. The primary objective is to reduce the overall transmission energy consumption in the MEC system. To achieve this, the authors combine Wireless Power Transmission (WPT) and energy allocation techniques, leading to a partial task offloading strategy known as WPT-MEC.
- **Xu C:** In the context of heterogeneous networks and MEC task offloading, Xu et al. [122] aim to minimize system energy consumption. To enhance system performance, the authors employ Non-Orthogonal Multiple Access (NOMA) within the network. The problem is divided into subproblems, and an optimal solution is iteratively obtained. Comparative experiments demonstrate the superiority of this scheme over baseline methods.
- **EEDOS:** Ali et al. [123], the focus is on task offloading within a single-user scenario. The study takes various factors into account, such as data offloading volume, energy consumption, network conditions, and computational load. A cost function that encompasses all these factors is established, transforming the problem into one of calculating the cost for all possible offloading strategies. To avoid exhaustive solutions, the authors first create a mathematical model to generate a dataset, then train a deep learning network. This training process results in an energy-efficient deep learning-based offloading scheme called EEDOS. It is worth noting that this approach assumes sequential task execution without addressing the uncertainties present in real-world scenarios.
- **LiMPO:** Zaman et al. [124] introduce a lightweight mobility prediction and offloading (LiMPO) framework that uses artificial neural networks with reduced complexity to offload compute-intensive tasks to the predicted user location.

### *6.3. The optimization objective is to reduce the delay*

This section guides towards research efforts aimed at diminishing latency in MEC task offloading. It delve into the particulars of the papers related to this subject, offering detailed insights below. Additionally, a summary of these characteristics can be found in Table 6.

**Table 6**
MEC task uploading schemes for reducing the time delay.

| Algorithm | Limitations | Year | Literature |
|---|---|---|---|
| LBTO | Task processing delay | 2021 | [77] |
| GABO | The algorithm has high complexity | 2020 | [78] |
| DQN, DDPG | N/A | 2022 | [79] |
| DRL | Simple task dependency model | 2022 | [81] |
| Jeon Y | The optimal problem in this paper is NP-Hard | 2021 | [80] |
| BCD | N/A | 2021 | [82] |
| Q-learning | Simple system model | 2021 | [83] |
| Heuristic algorithm | N/A | 2021 | [84] |
| GTGP | Simple system model | 2021 | [125] |
| DDPG | Virtual environment in collaborative computing, not secure | 2021 | [126] |
| QCQP | Ignores the competition for resources, not secure | 2021 | [127] |
| TSHA | Virtual environment in collaborative computing, not secure | 2021 | [128] |
| DODQN | Virtual environment in collaborative computing, not secure | 2021 | [129] |
| SSCCTO | Virtual environment in collaborative computing, not secure | 2021 | [130] |
| BDO | N/A | 2021 | [70] |
| iTOA | Virtual environment in collaborative computing, not secure | 2020 | [131] |
| DRL-Based | Virtual environment | 2020 | [132] |
| MRLCO | Bottleneck of computing resources and communication | 2020 | [133] |
| D-QLOA | Virtual environment in collaborative computing, not secure | 2020 | [134] |
| DROOO | Mobility of WDs not considered | 2019 | [135] |
| SDTO | Virtual environment | 2018 | [136] |
| OREO | N/A | 2018 | [136] |
| DRLCO | Without multiple MEC servers | 2023 | [137] |

- **LBTO:** He et al. [77] present an architecture for an unmanned aerial vehicle (UAV)-assisted ad hoc network (VANET). Their primary objective is to minimize task processing delays. The paper formulates the vehicle task offloading problem as a multi-objective optimization challenge. This problem is further divided into two sub-problems: MEC server selection and task offloading. The authors also present a novel method that tackles the combined issues of task offloading, resource allocation, and security guarantees.
- **GABO:** The work by Li et al. [78] introduces an optimization approach, known as Genetic Algorithm-Based Offloading (GABO), for task offloading within MEC systems. This method involves combining various strategies to create an overall matrix. For each combination, the task with the shortest completion time is considered the optimal task.
- **DQN & DDPG:** In the work by Liu et al. [79], the authors leverage Deep Reinforcement Learning (DRL) techniques, specifically employing a Deep Q-Network (DQN) and a Deep Deterministic Policy Gradient (DDPG) algorithm. These methods are designed for offshore unmanned aerial vehicles (UAVs) to enable multi-task parallel computing on a server.
- **DRL:** Qiao et al. [81], the main objective is to improve user service quality by reducing system delays. The authors introduce a joint deep reinforcement learning (DRL) approach, based on convex optimization algorithms, which effectively reduces system delays. This method leverages a Deep Q-Network (DQN) to make offloading decisions and utilizes the Lagrange multiplier method to allocate computing power from the server to multiple users.
- **Jeon Y:** Jeon et al. [80], the focus is on task offloading within a distributed edge computing (DEC) environment. The research problem centers on minimizing response times for mobile device task offloading in DEC settings. The authors propose a method that selects edge devices and manages task offloading, taking mobility into account within wireless DEC environments. However, it is important to emphasize that the optimal solution in this paper is NP-Hard, but the authors provide a heuristic algorithm with low complexity to address it.
- **Block Coordinate Descent (BCD) :** Feng et al. [82], focus on the joint optimization of offloading decisions, computation, and bandwidth resource allocation. This problem is notably different from the approach in another paper (referenced as [83]) where subproblems in the RAPD algorithm are merged into a single high-complexity problem. In this paper, the original problem is decomposed into two low-complexity subproblems, sequentially considering offloading decisions and bandwidth, as well as the allocation of computing resources. The authors also propose a coordinate reduction approach based on blocks.
- **Q-learning:** Wang et al. [83] delve into the MEC system within vehicle networks. In this study, after task offloading, Software-Defined Networking (SDN) and MEC are integrated to achieve unified network resource allocation. The authors introduce models for computing offloading and resource allocation and employ Q-learning to solve them. However, it is crucial to emphasize that the experiments are conducted without considering the complex environmental factors inherent in vehicular networking systems.
- **Heuristic algorithm:** Recognizing the limitations of existing working hypotheses that often overlook crucial network and processing model details, hierarchical topology, and more, Rodrigues et al. [84] propose a mathematical model for MEC systems that simulates complex environments with multiple parameters. This model captures the intricacies of MEC systems, including server mobility, task processing possibilities, and network conditions. The paper further introduces a heuristic algorithm designed to determine when and where mobile MEC users should offload tasks. The algorithm optimizes for minimal task transmission and processing times, considering multiple environments and the effectiveness of execution on local, edge cloud, and remote cloud servers. The evaluation includes expected parameters and technologies relevant to 6G networks.
- **GTGP:** In contrast to directly offloading tasks to the cloud, Naouri et al. [125] introduce a novel approach that considers the computational capabilities of MDs. Specifically, the authors employ a three-tier task offloading strategy (Device Layer, Cloudlet Layer, and Cloud Layer - DCC) based on the computational requirements of each task. Tasks with minimal computation needs are processed locally to reduce the communication overhead associated with task transmission. The authors further propose Greedy Task Graph Partition Algorithm (GTGP), which is also applied in face recognition systems.
- **DDPG:** In the work by Wang et al. [126], the focus is on UAV computation offloading. The paper addresses optimization objectives related to user scheduling, communication channel management, and resource allocation within UAVs. Importantly, the authors account for the time-varying nature of the communication

channel during task offloading. To address this non-convex issue, they model it as a Markov Decision Process (MDP) and propose a task offloading algorithm based on Deep Deterministic Policy Gradient (DDPG) to reduce task delay.

- **QCQP:** Zhang et al. [127] explore quadratic constrained quadratic programming (QCQP) optimization within a multi-task offloading scenario. The study takes into account user constraints related to computational costs while aiming to optimize edge server cache for enhanced system performance. The joint optimization of caching, resource allocation, and offloading in this context is recognized as an NP-hard problem. To address it, the authors employ an approximation algorithm to enable MDs to make cost-budget decisions. Simulation experiments validate the feasibility of this approach.
- **TSHA:** In [128], the topic is task uninstallation in the Industrial Internet of Things (IIoT) environment with assistance from edge computing. The paper acknowledges the detrimental impact of uncertain wireless channels on end-to-end delay. To address this challenge, the authors construct a non-convex Mixed-Integer Nonlinear Programming (MINLP) problem, considering both average and worst-case delay scenarios. They decompose this problem into two sub-problems and propose a two-stage heuristic algorithm to mitigate these risks.
- **DODQN:** In the work by Chen et al. [129], the focus is on task offloading within Vehicular Ad-Hoc Networks (VANET). The research goal is to minimize task execution delays. To address the challenge of limited MEC resources, the authors utilize surrounding vehicles as a resource pool (RP). In this context, they model vehicle movement and determine the service time that Moving Vehicles (MVs) can provide to requested Vehicles (RVs) based on their relative distances. The paper introduces a distributed computing offloading strategy known as DODQN, which is based on a deep Q-learning network for vehicle task allocation.
- **SSCCTO:** In [130], the research objective is to reduce the average delay of system services. However, the paper takes into account the leasing costs incurred by application service providers. It comprehensively considers task scheduling and task caching, modeling task offloading as a chain structure. The authors employ Lyapunov optimization to transform the NP problem and propose the SSCCTO algorithm to address it.
- **BDO:** In [70], the authors introduce an extensible neural network-based task offloading technology designed for blockchain scenarios and applied to mobile social networks. Their approach involves using Deep Learning (DL) technology to verify offloaded blocks in Proof-of-Work (POW) scenarios. The algorithm is robust against data loss and efficiently manages scarce communication resources.
- **iTOA:** In [131], the focus is on task offloading within a UAV edge computing network using wireless communication. The offloading problem is modeled as a Markov Decision Process (MDP). To achieve optimal decision-making, the authors present an Intelligent Task Offloading Algorithm (iTOA). Additionally, the paper includes the design of a UAV edge computing platform, where the algorithm is simulated and tested.
- **DRL-based:** In response to the time-sensitive 0–1 task offloading problem, Tang et al. [132] introduce a distributed offloading algorithm based on modelless Deep Reinforcement Learning (DRL). This algorithm leverages dual Deep Q-Network (DQN), dual DQN technology, and Long Short-Term Memory (LSTM) technology to make offloading decisions even when other equipment decisions are unknown. In comparison to the approach presented in Yang et al. [138], the proposed algorithm exhibits improved task processing efficiency and a lower packet loss rate.
- **MRLCO:** In consideration of the reliance of Deep Reinforcement Learning (DRL) offloading schemes on ample training data to update offloading strategies, Wang et al. [133] propose a task offloading strategy based on Meta Reinforcement Learning (MRL). This strategy is designed for environments with limited gradient updates and fewer samples. It enhances sample learning efficiency for dynamic offloading scenarios, but it does not incorporate mechanisms to filter out stragglers during training.
- **D-QLOA:** Guo et al. [134], focus on a MEC system designed for vehicle networks. This study aims to propose a task offloading scheme specifically designed for Vehicle Edge Computing Networks (VECNs) using Software-Defined Networking (SDN). This scheme enables centralized collection and management of network information. Importantly, it takes into account the mobility of Intelligent Connected Vehicles (ICVs). To minimize processing delays during task offloading, the paper constructs an offloading algorithm based on Q-learning.
- **DROOO:** In [135], deep neural networks are employed to learn binary offloading decisions based on experience. These decisions determine whether the computation task of a MD should be executed locally or completely offloaded to the MEC server. The authors introduce the Deep Reinforcement Learning Online Offloading (DROOO) framework as a flexible solution. This framework optimally adapts to varying wireless channel conditions while considering task offloading decisions and resource allocation. Experimental results demonstrate that the DROOO algorithm effectively reduces CPU execution delays while maintaining similar performance levels. However, it should be noted that convergence becomes challenging when considering the mobility of WDs.
- **SDTO:** In the work by Chen et al. [136], the task offloading problem within dense network scenarios is addressed. Leveraging Software-Defined Ultra Dense Networks (SD-UDN), the paper introduces an efficient Software-Defined Task Offloading (SDTO) scheme capable of minimizing task processing delays. Comparative results indicate that this scheme outperforms random and uniform task offloading approaches.
- **OREO:** In contrast to the study presented in [136], which primarily focuses on improving resource allocation efficiency to enhance computational offloading performance, the work by Xu et al. [139] delves into the optimization of service caching by MEC within dense network scenarios. The core aim here is to minimize computational delays. To achieve this goal, the authors employ Lyapunov optimization and introduce the Online Service Caching for MEC (OREO) algorithm. Additionally, in order to optimize decentralized coordination between Base Stations (BSs), they extend the OREO algorithm and propose the Distributed algorithm for OREO. This distributed algorithm is based on a variation of Gibbs sampling, offering a comprehensive approach to enhance computational offloading performance through efficient service cache optimization within dense network environments.
- **DRLCO:** Liao et al. [137] propose the double reinforcement learning computation offloading(DRLCO) algorithm, which simultaneously determines offloading decisions, CPU frequency, and transmit power for computation offloading.

### *6.4. MEC task uploading schemes for high computing offloading*

This section focus on high computing offloading research. Their characteristics are described in detail below and summarized in Table 7.

- **BIIA:** Through a study of sub-game optimization problems in each stage, the reference by Zuo et al. [85] introduces an iterative algorithm designed to achieve the Nash equilibrium of a Stackelberg game. Furthermore, the paper delves into the interaction among the three stages of the game, providing a comprehensive exploration of this subject.

**Table 7**
MEC task uploading schemes for high computing offloading.

| Algorithm | Limitations | Year | Literature |
|---|---|---|---|
| BIIA | High consumed energy | 2021 | [85] |
| FL-DRL | Ignore network usage, delay and fairness | 2021 | [86] |
| MITA | Weak conclusion | 2021 | [87] |
| ADMM | Considers only single MEC server scenario | 2021 | [140] |
| DBR | Virtual environment in collaborative computing, not secure | 2021 | [141] |
| Li W | Simple system model | 2021 | [142] |
| DMEC | Virtual environment in collaborative computing, not secure | 2021 | [143] |
| K-NJTA | Algorithm does not adapt to dynamic environment | 2021 | [144] |
| TATO | Bottleneck of Transmission and SN Perception Capability | 2021 | [145] |
| Ma C | Virtual environment in collaborative computing, not secure | 2021 | [146] |
| OPD | Virtual environment, without considering communication factors | 2021 | [147] |
| Ma X | High throughput with high robustness | 2021 | [148] |
| BRL | Virtual environment and ignores the competition for resources | 2021 | [149] |
| GRL-based | Virtual environment in collaborative computing, not secure | 2021 | [150] |
| STMTO | The UAV scene studied is simple, virtual environment | 2021 | [151] |
| FEA | The UAV scene studied is simple, virtual environment | 2021 | [152] |
| OST | Virtual environment and ignores the competition for resources | 2021 | [153] |
| Xue J | Virtual environment in collaborative computing,not secure | 2021 | [154] |
| GSO | Virtual environment, weak system model | 2021 | [155] |
| ILP | High computational cost | 2020 | [88] |
| OD-SARSA | Virtual environment | 2020 | [156] |
| CQNV | Considers only partial offloading is supported | 2020 | [157] |
| HGOS | Not applicable to multi-user scenarios | 2018 | [158] |
| JTORA | Simple task dependency model | 2018 | [159] |
| MCVCO | Only vehicular computation scene | 2023 | [160] |

- **FL-DRL:** In the work by Shahidinejad et al. [86], the impact of context information on the task uninstallation decision is considered. The paper employs autonomous management in the form of a MAPE loop to gather context during the uninstallation process. An uninstallation algorithm based on deep reinforcement learning and federated learning (FL-DRL) is proposed, leveraging the distributed capabilities of MEC to update weights between MDs and Edge Devices (Eds). While simulation results demonstrate the method's effectiveness, it is essential to emphasize that this context-aware algorithm fails to take into account indicators including energy consumption, execution cost, network usage, time delay, and fairness.
- **MITA:** Zhao et al. [87] explore the offloading of computing tasks within a vehicle-moving edge network. The main goal of this offloading is to optimize the utilization of the system's computing resources. This paper transforms the offloading problem into a multi-device decision-making challenge within a multi-objective sequential game. To address this challenge, the authors introduce the Minimum Incremental Task Allocation (MITA) algorithm. Additionally, the paper proposes a solution involving the deployment of multiple drones to provide computing services and allocate computing tasks to various available devices.
- **ADMM:** With the aim of enhancing the computational performance of a MEC system by leveraging surrounding resources for collaborative computing, Lv et al. [140] apply machine learning methods to MEC servers. The authors introduce a distributed Stackelberg game task scheduling algorithm. Test results demonstrate that this algorithm achieves fast convergence and maintains stability, even in large-scale networks.
- **DBR:** In the work by Zhou et al. [141], the focus is on a distributed multi-agent MEC system, specifically addressing the task offloading optimization problem while considering the dynamic nature of task processing and communication. Initially, the paper explores the optimization of offloading thresholds for multiple agents within the system, effectively transforming the problem into a game where the objective is to maximize the expected offloading rate. Building upon game theory analysis, the authors introduce a Distributed Best-Response (DBR) framework. Furthermore, an unconstrained Lagrangian optimization (ULO) method is proposed for optimizing the threshold of a single agent. The numerical results conclusively demonstrate the effectiveness of this approach.
- **Li W:** The work by Li et al. [142] is dedicated to enhancing the energy utilization efficiency of a heterogeneous single-user MEC system. This system offers the flexibility to offload tasks to local resources, MEC servers, or remote cloud servers. The study aims to balance factors such as average delay, energy consumption, and system overhead related to task processing. To achieve this, a task cost function is formulated, enabling the derivation of a corresponding task offloading strategy. This research primarily focuses on the allocation of task offloading positions. Simulation results underscore the efficacy of this offloading strategy in improving system performance and reducing allocation overhead.
- **DMEC:** Luo et al. [143] introduce a drone-aided Mobile Edge Computing (DMEC) scheme incorporating blockchain technology, applied within the context of the Ultra-Reliable Vehicular Edge Computing (UVEC) system for task offloading. This innovative scheme dynamically caches data generated by equipment onto drones, facilitating data forwarding even when direct access to the MEC network is unavailable. Furthermore, data not directly reaching the MEC network is forwarded to a private blockchain edge server, expanding the coverage of MEC service provisioning. The effectiveness of this scheme is substantiated through practical experiments conducted within a real-world blockchain network.
- **K-NJTA:** Wang et al. [144], focus on task offloading within vehicle networking MEC systems. The authors introduce a novel PP-VEC mobile edge system architecture, which prioritizes the protection of context information during vehicle task offloading, resulting in higher throughput. Specifically, when offloading information is transmitted to Roadside Units (RSUs) and Base Stations (BS), the privacy mechanism within this architecture employs differential privacy technology to disturb context information. Furthermore, the paper presents a task offloading and resource allocation algorithm using K-Nearest Neighbor Branch and Bound (K-NJTA). This algorithm optimizes task processing delay by simultaneously optimizing task scheduling and resource allocation.
- **TATO:** Task offloading in vehicle networks often faces the problem of heavy traffic scenarios. In [145], the authors propose a Binary Search and Feasibility Test (BSFC) algorithm designed to optimize the Overall Response Time (ORT) within specific thresholds. This algorithm aims to ensure that all subtasks are

finished within the ORT threshold while also addressing perceptual fusion constraints. Recognizing the sensing capabilities of service nodes (SNs), the paper presents a Traffic-Aware Task Offloading (TATO) mechanism that leverages environmental input as SNs' input. Extensive comparisons reveal that this offloading mechanism, enhanced by environmental perception, significantly reduces the Overall Response Time (ORT) when compared to pure communication-based and perception-based offloading mechanisms.

- **Ma C:** Addressing the stringent requirements for high reliability and low latency in Vehicular Edge Computing (VEC), Ma et al. [146] introduce an innovative approach. It organizes idle vehicles into clusters, effectively transforming them into virtual edge servers within the VEC infrastructure. This approach aims to alleviate resource bottleneck constraints commonly faced by servers while expanding the VEC service area through a concept known as parking edge computing. The authors of this framework propose methods for task scheduling and resource allocation, considering the delay threshold of tasks and the energy consumption of vehicles carrying out these tasks. The framework's effectiveness is validated through simulation, which incorporates real city maps and traffic scenarios, demonstrating its ability to stably handle a higher volume of task requests.
- **OPD:** In their work [147], Wang H. et al. concentrate on enhancing the robustness of task offloading. Specifically, they address scenarios like battlefield monitoring and post-disaster rescue, which demand a high level of reliability for task assistance. The authors introduce online primal–dual algorithms that effectively reduce edge server failures and enhance the throughput of Dynamic Edge Computing (DEC) systems. This paper discusses the algorithm's effectiveness across various dynamic edge computing (DEC) tolerance thresholds.
- **Ma X:** In the context of MEC, it is essential to strike a balance between reducing computational offloading costs and preserving the information capacity of edge nodes, which in turn can boost wireless channel bandwidth. However, prior research has not adequately addressed the inherent conflicts in this trade-off. In contrast, the work presented in [148] jointly optimizes channel allocation and offloading strategies to minimize computational offloading costs while maintaining data freshness. This challenging problem is framed as a dynamic nonlinear integer optimization problem. To address it, the authors employ Lyapunov optimization techniques, effectively transforming the problem into a series of solvable static optimization subproblems.
- **BRL:** In their paper [149], the authors propose an offloading framework centered around Lagrange coding calculations (LCC). Within this framework, base stations are categorized into "masters" and "workers", managed either privately or by operators. A master base station selects the most suitable worker base stations within its observation range to offload its computation tasks, effectively solving the selected worker base stations' assigned tasks. The paper establishes a random auction model, which is subsequently transformed into a random Bayesian game and addressed using machine learning algorithms. Moreover, the authors introduce a Bayesian Reinforcement Learning (BRL) algorithm to match the optimal strategies within this context.
- **GRL-Based:** In their work [150], the authors employ a Graph Convolutional Network (GCN) optimization approach combined with a Deep Reinforcement Learning (DRL) model for task offloading. They break down the task offloading process into a task set represented as a directed acyclic graph. The decision-making process is implemented at the user layer, and the task set is further modeled as a Markov Decision Process (MDP) to enable continuous offloading.
- **STMTO:** Huang et al. [151] investigate a multi-UAV task offloading system (STMTO) in which tasks are gathered by UAVs and subsequently offloaded to edge servers. The system initially partitions the UAV's working area and facilitates the collection of collaborative tasks by UAVs to enhance system performance. Subsequently, MDs, Edge Computers (ECs), and UAVs assume the roles of buyers, sellers, and auctioneers, respectively. They establish a multi-pair multi-task double auction model. Both analysis and experimental data demonstrate that the system effectively achieves efficient task processing.
- **FEA:** In the study presented in [152], the authors jointly consider Orthogonal Frequency-Division Multiple Access (OFDMA) and backscattering-assisted Wireless Power Mobile Edge Computing (WPMEC). MDs can acquire energy either from a central gateway or perform local computations with backscattered data. Given that this problem is NP-hard, the authors propose a two-level alternating algorithm and solve it using the block coordinate descent (BCD) method. Furthermore, they introduce a fast and efficient algorithm (FEA) based on the BCD approach.
- **OST:** Alghamdi et al. [153] explore the application of Optimal Stopping Theory (OST) in the sequential decision-making process of task offloading. In the context of sequential task offloading problems, quality perception plays a significant role in decision evaluation. Factors like data freshness and the state of the best server influence the optimal decision probability (Odds). Moreover, the authors enhance the timeliness function algorithm. Numerical experiments demonstrate the algorithm's effectiveness in ensuring Quality of Service (QoS).
- **Xue J:** Similar to the approach presented in [161], which organizes idle vehicles into clusters, the paper [154] utilizes vehicles with unused computing resources to provide computing capabilities to nearby users, addressing the surge in short-term resource demands during peak hours. The author leverages idle intelligent vehicles as vehicle-edge nodes and introduces a Vehicle Mobile Edge Computing (VMEC) paradigm, allowing offloading to MEC servers and Vehicle Edge Nodes (VENs).
- **GSO:** In the work [155], the paper discusses the problem of minimizing time delay while adhering to energy consumption constraints in multi-user MEC task offloading scenarios. Recognizing the tradeoff between energy consumption and time delay optimization, the authors propose a Glowworm Swarm Optimization (GSO) algorithm to enhance computational offloading methods. By expanding the number of ideas, the GSO algorithm outperforms Particle Swarm Optimization (PSO) and Ant Lion Algorithm for Search Optimization (AFAS) individually regarding energy consumption and time delay under the same conditions. Numerical experiments illustrate how the proposed algorithm effectively balances both energy consumption and time delay through performance optimization.
- **ILP:** The paper [88] delves into a multi-user, multi-server MEC system. Focusing on the computational offloading process involving an energy harvesting device, the authors propose a MEC optimization computational offload technology based on Integer Linear Programming (ILP). This approach involves deploying servers in proximity to Distance Energy Collection (DEC) devices, enabling wireless communication between MDs and the MEC server.
- **OD-SARSA:** In the context of limited MEC system resources, the paper [156] addresses the efficient fulfillment of high Quality of Service (QoS) requirements through judicious resource allocation. To optimize both delay and energy consumption, the authors introduce an Offloading Decision-based SARSA method (OD-SARSA). This approach also accounts for the mobility of MDs. Experimental results demonstrate the superiority of this algorithm compared to the Reinforcement Learning-based Q Learning (RL-QL) algorithm.

- **Choosing Qualified Nearby Vehicle(CQNV):** Taking into account tolerance for delay and vehicle mobility, and effectively utilizing nearby vehicles and VEC servers for task processing, [157] introduces a task offloading strategy with perceptual mobility. This strategy optimizes task allocation by evaluating the relative portions of tasks to be processed, while assuming a maximum tolerable delay for the system.
- **HGOS:** In comparison to studies that focus on single MEC scenarios with multiple users, [158] delves into computational offloading between multiple MEC servers. It addresses the task offloading challenges within multi-user ultra-dense networks. The model accounts for task processing time, including queuing time post-offloading and execution time on the edge server. The paper presents a Heuristic Greedy Offloading Scheme (HGOS) designed to reduce overall energy consumption effectively. However, it is important to note that this paper specifically examines the case of a single Mobile Device (MD).
- **JTORA:** In the context of multi-cell wireless network MEC systems, [159] explores ways to maximize task offloading. This paper addresses the decision-making process for task offloading, MEC server resource allocation, and user task transmission power. Acknowledging that this constitutes a mixed-integer nonlinear programming (MINLP) problem, the paper decomposes it into sub-problems to find an optimal solution and introduces a heuristic algorithm named JTORA. Numerical simulations confirm the algorithm's practical performance.
- **MCVCO:** Liu et al. [160] propose a multi-MEC cooperative vehicular computation offloading (MCVCO) scheme, which adopt a heat-aware task offloading strategy to capture the time-varying multi-link relations between vehicle and MEC nodes. A parallel transmission and execution based dynamic scheduling algorithm is developed to make the most of available resources.

### 6.5. Comparison of various algorithms and its applications

The Table 8 presents a comprehensive overview of various studies focusing on performance metrics related to computation offloading in mobile edge computing (MEC) across multiple domains since 2016. It categorizes the research based on evaluation tools, applications, performance metrics, and publication years. Notably, simulation remains the predominant method for evaluation, indicating a strong reliance on theoretical modeling to assess offloading strategies. The studies cover a diverse range of applications including healthcare, virtual reality, vehicular networks, and smart cities. Performance metrics vary widely but primarily focus on aspects like delay, energy consumption, cost, and quality of experience (QoE). For instance, healthcare applications leverage tools like iFogSim and CPLEX solver to analyze delay and cost, while vehicular networks emphasize metrics like response time and reliability. The increasing complexity of applications analyzed over the years suggests a growing acknowledgment of the need to optimize MEC strategies for improved efficiency and user satisfaction in strained network environments. As the field evolves, the integration of novel evaluation tools and methodologies reflects an ongoing effort to refine offloading mechanisms and adapt to emerging computational challenges.

## 7. Discussion and future research directions

### 7.1. Key findings and insights

In recent years, the proliferation of mobile devices (MDs) driven by emerging applications such as medical care, augmented reality (AR), virtual reality (VR), media identification, intelligent transportation systems (ITS), and unmanned aerial vehicles has accentuated the significance of computational offloading as a method to address the inherent limitations of MDs [198]. Consequently, computational offloading is poised to remain a focal point in the MEC research landscape.

As MEC technology advances, it has largely resolved latency concerns when accessing cloud services. This paper starts with an introduction to MEC, delineating its foundational concepts and technological underpinnings, especially in the context of 5G and the IoT. Furthermore, this work expound on various computing paradigms. In our discussion of task offloading algorithms, it delve into the mathematical foundations of commonly employed algorithms in the literature, elucidating concepts such as Markov Decision Processes (MDP), Bellman Equations, Q-Learning, and Deep Q-Networks (DQN) within the context of reinforcement learning. These algorithms are then systematically classified based on their performance parameters.

### 7.2. Challenges and future trends in MEC offloading technology

Future exploration and development should focus on the effectiveness of the previously discussed offloading strategies under demanding practical conditions. Researchers have identified several challenges and research directions, as outlined below:

(1) **Resource Management:** Efficient resource management in a distributed environment is complex due to the interactions between devices and edge servers. Coordinating resource availability across these elements is crucial to optimize performance and prevent bottlenecks [199].
(2) **Predicting Stochastic behavior:** Edge server demand is stochastic in nature. Therefore stochastic model based prediction should be applied for task offloading. The identified model must predict the system's future behavior to efficiently offload tasks while monitoring the required metrics. However, only a few of the reviewed papers proposed methods addressing this need [200].
(3) **Security and Privacy Protection Issues:** Data safety during offloading is crucial, with privacy and protection against attacks being paramount. Breaches can significantly impact sensitive information and MEC system integrity. Key challenges include confidentiality, service reliability, and availability, which are interrelated aspects of dependability and security [201,202]. Data security: In MEC systems, the secure transmission and storage of data is a key issue. Future research will focus on technologies such as data encryption, access control, and security protocols to ensure the security of data during transmission and storage. Privacy protection: With the enhancement of privacy protection awareness, how to achieve efficient computing and uninstallation under the premise of protecting users' privacy has become an important topic. Future research will explore privacy protection schemes based on differential privacy, federated learning, and other technologies to balance the computational efficiency and privacy protection needs.
(4) **Mobility Management:** The high mobility of users and devices adds complexity to task offloading decisions. This mobility impacts resource allocation and overall efficiency, requiring strategies that consider the dynamic locations of users and devices to ensure optimal performance. Due to the unpredictable nature of task demand, effective mobility management is essential [203, 204], including the management of user positioning, resources, and services [205].
(5) **Task Deployment and Scheduling:** The deployment and scheduling of tasks across various edge servers is challenging, requiring sophisticated algorithms to decide which tasks to offload to which servers and when to execute them to minimize latency and maximize resource utilization. Dynamic resource allocation: In MEC systems, the dynamic allocation of resources is critical to improve system performance and resource utilization. Future

**Table 8**
Comparison of various offloading mechanisms.

| Reference | Evaluation tools | Case study/application | Performance metrics | Year |
|---|---|---|---|---|
| [162] | Simulation (NA) | General apps | Security, Energy, QoE | 2016 |
| [163] | Simulation (NA) | VR | Delay, Energy | 2017 |
| [164] | Simulation (iFogSim + CPLEX solver) | Healthcare, Media Streaming, Augmented Reality, Smart Homes | Delay | 2018 |
| [165] | Numerical | Social networking | Delay, Cost | 2018 |
| [166] | Simulation (Python) | Vehicular Networks | Delay | 2018 |
| [167] | Implementation (TensorFlow) | Vehicular Networks | Cost, Reliability | 2018 |
| [168] | Numerical | General apps | Energy, Response time, Delay | 2018 |
| [169] | Simulation (NA) | Gaming, AR, Multimedia processing | Delay | 2018 |
| [170] | Simulation (CPLEX solver) | Healthcare, Intelligent Transportation Systems | Delay, Cost | 2019 |
| [171] | Simulation (Choco solver) | Intelligent Transportation Systems, Augmented Reality, Wearable Devices | Resolution Times | 2019 |
| [172] | Simulation (Java + C) | Smart Building | Energy | 2019 |
| [173] | Simulation (Java) | Media Streaming, Mobile Phones | Delay, Costs | 2019 |
| [174] | Simulation (NA) | Virtual Reality, Augmented Reality | Delay | 2019 |
| [175] | Simulation (NA) | IoT Applications | Energy, Costs | 2019 |
| [176] | Simulation (iFogSim + CPLEX solver) | Intelligent Transportation Systems, Smart Cities, Agriculture | Energy, Costs, Response Time | 2019 |
| [177] | Simulation (NA) | Virtual Reality, Augmented Reality | QoS, Bandwidth | 2019 |
| [178] | Simulation (CloudSim) | IoT Applications | Energy, Costs, Performance | 2019 |
| [179] | Simulation (NA) | IoT Applications | Response Time, Throughput | 2019 |
| [180] | Implementation (NA) | VR | Throughput | 2019 |
| [181] | Simulation (NA) | VR, Gaming, Multimedia processing | Delay, Energy | 2019 |
| [182] | Simulation (C++) | IoT Applications | Energy, Delay | 2019 |
| [183] | Implementation (AMPL) | Smart Cities | Delay, Response Time | 2019 |
| [184] | Simulation (NA) | Augmented Reality, Autonomous Driving | Response Time | 2019 |
| [185] | Simulation (NA) | General apps | Delay, Energy | 2019 |
| [186] | Simulation (NA) | Vehicular Networks | Reliability | 2019 |
| [187] | Simulation (CPLEX solver) | Healthcare, Augmented Reality | Cost | 2020 |
| [188] | Simulation (Gurobi) | Healthcare, Surveillance, Agriculture | Energy | 2020 |
| [189] | Simulation (Python) | Intelligent Transportation Systems | Delay | 2020 |
| [190] | Simulation (NA) | Smart Cities | Costs, Bandwidth Utilization | 2020 |
| [191] | Simulation (Gurobi) | IoT Applications | Response Time | 2020 |
| [192] | Simulation (TensorFlow) | General apps | Delay | 2020 |
| [193] | Simulation (C++) | General apps | Response time, Throughput, QoS, Cost | 2020 |
| [194] | Simulation (Python + TensorFlow) | Social networking | Security | 2020 |
| [195] | Simulation (NA) | Virtual Reality, Online Gaming | Delay | 2021 |
| [196] | Simulation (Python) | IoT Applications | QoS | 2022 |
| [197] | Simulation (MATLAB) | Virtual Reality, Augmented Reality | Costs | 2022 |

studies will explore more flexible and efficient dynamic resource allocation algorithms to accommodate variable task demands and system states. Collaborative scheduling: The MEC system requires collaborative scheduling with the cloud server, other MEC nodes, and terminal equipment to achieve global optimal resource allocation. Future studies will focus on multi-level collaborative scheduling mechanisms to improve the overall performance and stability of the system.

(6) **Interoperability**: Addressing the challenges posed by heterogeneity in future networks involves managing communication between different types of networks, devices, providers, and user participation.Edge computing environments consist of a wide variety of devices, platforms, and networks from different vendors. Interoperability ensures that these heterogeneous systems can work together seamlessly. Lack of universally accepted standards and protocols makes it difficult to achieve interoperability across different edge computing platforms and devices. Further interoperability might introduce performance overheads due to additional layers of abstraction or protocol conversion [206].

(7) **Task offloading in large scale connected networks**: Task offloading in large scale social networks poses the following three major challenges:

(a) **Complexity:** As the network grows, task offloading decisions become more complex, leading to delays that increase overall latency [207].

(b) **Centralized Methods:** Optimization and machine learning methods need to collect a lot of data from the whole network before they can make decisions. In a big network, this process of gathering information can put too much strain on the system, making it difficult to make decisions quickly [208].

(c) **Real-Time Constraints:** For systems that must operate in real-time, such as smart transportation or healthcare networks, the model for offloading tasks has to manage various kinds of hardware and data. This makes the process even more complicated [209].

(8) **Machine Learning Integration**: Machine learning (ML) models can address various challenges in MEC systems, such as task allocation, ranking, and delay reduction. Conversely, MEC systems can support ML-based solutions for delay-sensitive applications like autonomous driving and smart cities. Developing an MEC system that effectively integrates with ML systems presents a significant challenge [140,198].

(9) **Economic and Pricing Models**: Exploring economic models and pricing policies to optimize caching strategies represents both a significant challenge and a valuable research opportunity. This area of study involves understanding how different pricing mechanisms and economic principles can be applied to enhance caching efficiency and effectiveness. Addressing this challenge could open up numerous research avenues, leading to innovative solutions and improvements in caching strategies [210].

(10) **Energy Consumption**: Energy-aware offloading decisions face challenges like managing diverse resources, handling substantial computation and communication demands, and dealing with intermittent connectivity and network capacity limitations. Developing energy-efficient caching mechanisms to reduce the power consumption of edge devices is, therefore, a significant challenge. Addressing these issues is crucial for optimizing the performance and sustainability of edge computing systems [63,211,

212]. Future research will explore green and energy-saving technologies, such as low-power hardware design, energy-sensing unloading strategies, and energy recovery technologies, to reduce the overall energy consumption of the system. Energy and performance balance: While pursuing low energy consumption, the system performance should also be guaranteed. Future studies will focus on how to find a balance between energy consumption and performance to enable the sustainable operation of the system.

(11) **Autonomic Systems**: Autonomy allows MEC systems to continue functioning even when disconnected from the central cloud, enhancing reliability and resilience, especially in remote or disaster-stricken areas. Advancing the development of autonomic systems for IoT service placement to improve efficiency and adaptability [213]. Therefore efficiently managing and allocating computational resources, storage, and network bandwidth autonomously across diverse and dynamic environments is a research challenge.

(12) **Optimization Algorithms**: Enhancing optimization algorithms, like the Gray Wolf optimizer, to better handle the complexities of IoT service placement. Intelligent offloading algorithm: With the continuous development of artificial intelligence technology, the intelligent unloading algorithm based on machine learning and deep learning will become a research hotspot. These algorithms are able to analyze the system state and task requirements in real time and dynamically adjust the offloading strategy to achieve the optimal offloading decision.

(13) **Service Placement Strategies**: Developing advanced strategies for placing services in IoT environments to optimize performance and resource utilization. Dynamic resource allocation: In MEC systems, the dynamic allocation of resources is critical to improve system performance and resource utilization. Future studies will explore more flexible and efficient dynamic resource allocation algorithms to accommodate variable task demands and system states. Collaborative scheduling: The MEC system requires collaborative scheduling with the cloud server, other MEC nodes, and terminal equipment to achieve global optimal resource allocation. Future studies will focus on multi-level collaborative scheduling mechanisms to improve the overall performance and stability of the system.

(14) **Real-time perception of the environment**: When users offload tasks to edge servers, the quality of service (QoS) is affected by fluctuations in mobile edge computing (MEC) networks. Factors such as varying wireless link data rates, changes in transmission bandwidth, base station power, and unpredictable wireless conditions can significantly impact QoS. Researchers have proposed various solutions to address these challenges, including game theory-based approaches [214,215].

## 8. Summary

In the paper, we have studied the task offloading method of mobile edge computing, and discussed the existing review articles. Our work mainly focuses on the task unloading decision for mobile edge computing. Through the introduction of mobile edge computing related technologies and unloading models, we can have a comprehensive understanding of it. In addition, there are five dimensions for determining task offloading decisions, including Energy Consumption Minimization, QoS, Time Delay Minimization, High-computing Offloading and different application scenarios. The paper also offer insights into the research landscape's hot topics, and future trends for edge computing of MEC. Looking ahead, as MEC application scenarios continue to proliferate, and communication technologies advance, the research value and importance of task-offloading algorithms will undoubtedly increase. We trust that this work will provide valuable references and insights for researchers working in related fields, guiding them in their pursuit of innovative solutions within the evolving MEC ecosystem.

## CRediT authorship contribution statement

**Shi Dong:** Methodology, Conceptualization. **Junxiao Tang:** Validation, Resources, Investigation. **Khushnood Abbas:** Writing – review & editing, Validation, Software. **Ruizhe Hou:** Writing – review & editing. **Joarder Kamruzzaman:** Writing – review & editing. **Leszek Rutkowski:** Writing – review & editing. **Rajkumar Buyya:** Writing – review & editing.

## Declaration of competing interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work; there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

## Data availability

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

## Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments to improve the technical quality of the paper. This paper is supported by Key Scientific Research Projects of Colleges and Universities in Henan Province (Grant No. $23A520054$), Henan Provincial Science and Technology Research and Development Program Joint Fund (Application Research)(Grant No. 232103810045), Henan Province High end Foreign Expert Introduction Program Project(Grant No. $HNGD2024034$), and Open Foundation of State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) (Grant No. SKLNST-2020-2-01).

## References

[1] H. van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, in: D. Schuurmans, M.P. Wellman (Eds.), Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA, AAAI Press, 2016, pp. 2094–2100, http://dx.doi.org/10.1609/aaai.v30i1.10295.

[2] J. Ren, D. Zhang, S. He, Y. Zhang, T. Li, A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet, ACM Comput. Surv. 52 (6) (2019) 1–36.

[3] U. Cisco, Cisco Annual Internet Report (2018–2023) White Paper. 2020, 2021, Acessado em 10 (01).

[4] K. Yoshii, What does the post-Moore era mean for research software engineering? 2021, arXiv preprint arXiv:2111.05999.

[5] P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1628–1656.

[6] H. Guo, J. Liu, Collaborative computation offloading for multiaccess edge computing over fiber–wireless networks, IEEE Trans. Veh. Technol. 67 (5) (2018) 4514–4526.

[7] Q.-V. Pham, L.B. Le, S.-H. Chung, W.-J. Hwang, Mobile edge computing with wireless backhaul: Joint task offloading and resource allocation, IEEE Access 7 (2019) 16444–16459.

[8] C. Jiang, X. Cheng, H. Gao, X. Zhou, J. Wan, Toward computation offloading in edge computing: A survey, IEEE Access 7 (2019) 131543–131558.

[9] B. Wang, C. Wang, W. Huang, Y. Song, X. Qin, A survey and taxonomy on task offloading for edge-cloud computing, IEEE Access 8 (2020) 186080–186101.

[10] H. Lin, S. Zeadally, Z. Chen, H. Labiod, L. Wang, A survey on computation offloading modeling for edge computing, J. Netw. Comput. Appl. 169 (2020) 102781.

[11] A. Shakarami, A. Shahidinejad, M. Ghobaei-Arani, A review on the computation offloading approaches in mobile edge computing: A g ame-theoretic perspective, Softw. - Pract. Exp. 50 (9) (2020) 1719–1759.

[12] T. Zheng, J. Wan, J. Zhang, C. Jiang, G. Jia, A survey of computation offloading in edge computing, in: 2020 International Conference on Computer, Information and Telecommunication Systems, CITS, IEEE, 2020, pp. 1–6.

[13] E. Mustafa, J. Shuja, A.I. Jehangiri, S. Din, F. Rehman, S. Mustafa, T. Maqsood, A.N. Khan, et al., Joint wireless power transfer and task offloading in mobile edge computing: a survey, Cluster Comput. (2021) 1–20.

[14] J. Qadir, B. Sainz-De-Abajo, A. Khan, B. García-Zapirain, I. De La Torre-Díez, H. Mahmood, Towards mobile edge computing: Taxonomy, challenges, applications and future realms, IEEE Access 8 (2020) 189129–189162.

[15] C. Feng, P. Han, X. Zhang, B. Yang, Y. Liu, L. Guo, Computation offloading in mobile edge computing networks: A survey, J. Netw. Comput. Appl. 202 (2022) 103366.

[16] S.A. Huda, S. Moh, Survey on computation offloading in UAV-Enabled mobile edge computing, J. Netw. Comput. Appl. 201 (2022) 103341.

[17] M.Y. Akhlaqi, Z.B.M. Hanapi, Task offloading paradigm in mobile edge computing-current issues, adopted approaches, and future directions, J. Netw. Comput. Appl. 212 (2023) 103568.

[18] X. Deng, J. Li, E. Liu, H. Zhang, Task allocation algorithm and optimization model on edge collaboration, J. Syst. Archit. 110 (2020) 101778.

[19] C. Dou, S. Zhang, H. Wang, L. Sun, Y. Huang, W. Yue, Adhd fmri short-time analysis method for edge computing based on multi-instance learning, J. Syst. Archit. 111 (2020) 101834.

[20] P. Cong, J. Zhou, L. Li, K. Cao, T. Wei, K. Li, A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud, ACM Comput. Surv. 53 (2) (2020) 1–44.

[21] C. Wu, Q. Peng, Y. Xia, Y. Ma, W. Zheng, H. Xie, S. Pang, F. Li, X. Fu, X. Li, et al., Online user allocation in mobile edge computing environments: A decentralized reactive approach, J. Syst. Archit. 113 (2021) 101904.

[22] Y. Mao, C. You, J. Zhang, K. Huang, K.B. Letaief, A survey on mobile edge computing: The communication perspective, IEEE Commun. Surv. Tutor. 19 (4) (2017) 2322–2358.

[23] Y.C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, Mobile Edge Computing a Key Technology Towards 5G, ETSI white paper 11, 2015, pp. 1–16, (11).

[24] C. Parada, F. Fontes, C. Marques, V.A. Cunha, C. Leitao, Multi-access edge computing: A 5G technology, in: 2018 European Conference on Networks and Communications, EuCNC 2018, Ljubljana, Slovenia, June 18-21, 2018, IEEE, 2018, pp. 277–279.

[25] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1657–1681.

[26] Cisco, New Cisco Annual Internet Report Forecasts 5G to Support More than 10% of Global Mobile Connections by 2023, 2020, cisco.com.

[27] P.L. Parcu, A.R. Pisarkiewicz, C. Carrozza, N. Innocenti, The future of 5G and beyond: Leadership, deployment and European policies, Telecommun. Policy 47 (9) (2023) 102622.

[28] A. Dogra, R.K. Jha, S. Jain, A survey on beyond 5G network with the advent of 6G: Architecture and emerging technologies, IEEE Access 9 (2020) 67512–67547.

[29] N. Zhang, P. Yang, J. Ren, D. Chen, L. Yu, X. Shen, Synergy of big data and 5G wireless networks: opportunities, approaches, and challenges, IEEE Wirel. Commun. 25 (1) (2018) 12–18.

[30] K. Dolui, S.K. Datta, Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing, in: 2017 Global Internet of Things Summit, GIoTS, IEEE, 2017, pp. 1–6.

[31] T.X. Tran, A. Hajisami, P. Pandey, D. Pompili, Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges, IEEE Commun. Mag. 55 (4) (2017) 54–61.

[32] F. Bonomi, R.A. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: M. Gerla, D. Huang (Eds.), Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC@SIGCOMM 2012, Helsinki, Finland, August 17, 2012, ACM, 2012, pp. 13–16.

[33] F. Lordan, D. Lezzi, J. Ejarque, R.M. Badia, An architecture for programming distributed applications on fog to cloud systems, in: Euro-Par 2017: Parallel Processing Workshops - Euro-Par 2017 International Workshops, Santiago de Compostela, Spain, August 28-29, 2017, Revised Selected Papers, in: Lecture Notes in Computer Science, vol. 10659, Springer, 2017, pp. 325–337.

[34] C. Mouradian, D. Naboulsi, S. Yangui, R.H. Glitho, M.J. Morrow, P.A. Polakos, A comprehensive survey on fog computing: State-of-the-art and research challenges, IEEE Commun. Surv. Tutor. 20 (1) (2017) 416–464.

[35] M. Salimian, M. Ghobaei-Arani, A. Shahidinejad, Toward an autonomic approach for Internet of Things service placement using gray wolf optimization in the fog computing environment, Softw. Pract. Exp. 51 (8) (2021) 1745–1772, http://dx.doi.org/10.1002/SPE.2986.

[36] V. Kumar, A.A. Laghari, S. Karim, M. Shakir, A.A. Brohi, Comparison of fog computing & cloud computing, Int. J. Math. Sci. Comput. 1 (2019) 31–41.

[37] A.A. Laghari, H. He, M. Shafiq, A. Khan, Impact of storage of mobile on quality of experience (QoE) at user level accessing cloud, in: 2017 IEEE 9th International Conference on Communication Software and Networks, ICCSN, IEEE, 2017, pp. 1402–1409.

[38] M.S. Aslanpour, S.S. Gill, A.N. Toosi, Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research, Internet Things 12 (2020) 100273.

[39] M. Babar, M.S. Khan, F. Ali, M. Imran, M. Shoaib, Cloudlet computing: recent advances, taxonomy, and challenges, IEEE Access 9 (2021) 29609–29622.

[40] U. Nandhini, U. S, L. Tamilselvan, S. Nancy, Client aware scalable cloudlet to augment edge computing with mobile cloud migration service, Int. J. Interact. Mob. Technol. 14 (12) (2020) 165–178.

[41] A.J. Ferrer, J.M. Marquès, J. Jorba, Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing, ACM Comput. Surv. 51 (6) (2019) 1–36.

[42] S. Bi, Y.J. Zhang, Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading, IEEE Trans. Wirel. Commun. 17 (6) (2018) 4177–4190.

[43] C. You, K. Huang, H. Chae, Energy efficient mobile cloud computing powered by wireless energy transfer, IEEE J. Sel. Areas Commun. 34 (5) (2016) 1757–1771.

[44] E. Jonas, J. Schleier-Smith, V. Sreekanti, C. Tsai, A. Khandelwal, Q. Pu, V. Shankar, J. Carreira, K. Krauth, N.J. Yadwadkar, J.E. Gonzalez, R.A. Popa, I. Stoica, D.A. Patterson, Cloud programming simplified: A berkeley view on serverless computing, 2019, CoRR abs/1902.03383. arXiv:1902.03383.

[45] J. Yan, S. Bi, L. Duan, Y.A. Zhang, Pricing-driven service caching and task offloading in mobile edge computing, IEEE Trans. Wirel. Commun. 20 (7) (2021) 4495–4512.

[46] R. Aghazadeh, A. Shahidinejad, M. Ghobaei-Arani, Proactive content caching in edge computing environment: A review, Softw. Pract. Exp. 53 (3) (2023) 811–855, http://dx.doi.org/10.1002/SPE.3033.

[47] P. Qi, Task offloading and scheduling strategy for intelligent prosthesis in mobile edge computing environment, Wirel. Commun. Mob. Comput. 2022 (2022) 2890473:1–2890473:13.

[48] I. Ahammad, M.A.R. Khan, Z.U. Salehin, QoS performance enhancement policy through combining fog and SDN, Simul. Model. Pract. Theory 109 (2021) 102292.

[49] S. Safavat, N.N. Sapavath, D.B. Rawat, Recent advances in mobile edge computing and content caching, Digit. Commun. Netw. 6 (2) (2020) 189–194.

[50] R. Xie, Q. Tang, Q. Wang, X. Liu, F.R. Yu, T. Huang, Satellite-terrestrial integrated edge computing networks: architecture, challenges, and open issues, IEEE Netw. 34 (3) (2020) 224–231.

[51] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, D.O. Wu, Energy-optimal mobile cloud computing under stochastic wireless channel, IEEE Trans. Wireless Commun. 12 (9) (2013) 4569–4581.

[52] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, Mobile-edge computing: Partial computation offloading using dynamic voltage scaling, IEEE Trans. Commun. 64 (10) (2016) 4268–4282.

[53] K. Arulkumaran, M.P. Deisenroth, M. Brundage, A.A. Bharath, Deep reinforcement learning: A brief survey, IEEE Signal Process. Mag. 34 (6) (2017) 26–38.

[54] J. Fang, J. Shi, S. Lu, M. Zhang, Z. Ye, An efficient computation offloading strategy with mobile edge computing for IoT, Micromachines 12 (2) (2021) 204.

[55] A. Irshad, Z.H. Abbas, Z. Ali, G. Abbas, T. Baker, D. Al-Jumeily, Wireless powered mobile edge computing systems: Simultaneous time allocation and offloading policies, Electronics 10 (8) (2021) 965.

[56] Y. Cui, D. Zhang, T. Zhang, P. Yang, H. Zhu, A new approach on task offloading scheduling for application of mobile edge computing, in: 2021 IEEE Wireless Communications and Networking Conference, WCNC, IEEE, 2021, pp. 1–6.

[57] C. Li, Q. Cai, Y. Luo, Multi-edge collaborative offloading and energy threshold-based task migration in mobile edge computing environment, Wirel. Netw. 27 (7) (2021) 4903–4928.

[58] L. Tang, B. Tang, L. Zhang, F. Guo, H. He, Joint optimization of network selection and task offloading for vehicular edge computing, J. Cloud Comput. 10 (1) (2021) 1–13.

[59] J. Zeng, J. Sun, B. Wu, X. Su, Mobile edge communications, computing, and caching (MEC3) technology in the maritime communication network, China Commun. 17 (5) (2020) 223–234.

[60] Z. Wang, T. Lv, Z. Chang, Computation offloading and resource allocation based on distributed deep learning and software defined mobile edge computing, Comput. Netw. (2022) 108732.

[61] A. Shakarami, A. Shahidinejad, M. Ghobaei-Arani, An autonomous computation offloading strategy in Mobile Edge Computing: A deep learning-based hybrid approach, J. Netw. Comput. Appl. 178 (2021) 102974.

[62] A. Li, L. Li, S. Yi, Computation offloading strategy for IoT using improved particle swarm algorithm in edge computing, Wirel. Commun. Mob. Comput. 2022 (2022) 9319136.

[63] M. Zhao, K. Zhou, Selective offloading by exploiting ARIMA-BP for energy optimization in mobile edge computing networks, Algorithms 12 (2) (2019) 48.

[64] Y. Shi, Y. Xia, Y. Gao, Cross-server computation offloading for multi-task mobile edge computing, Information 11 (2) (2020) 96.

[65] T.P. Truong, T.-V. Nguyen, W. Noh, S. Cho, et al., Partial computation offloading in NOMA-assisted mobile-edge computing systems using deep reinforcement learning, IEEE Internet Things J. 8 (17) (2021) 13196–13208.

[66] W. Zhou, L. Xing, J. Xia, L. Fan, A. Nallanathan, Dynamic computation offloading for MIMO mobile edge computing systems with energy harvesting, IEEE Trans. Veh. Technol. 70 (5) (2021) 5172–5177.

[67] F. Zhao, Y. Chen, Y. Zhang, Z. Liu, X. Chen, Dynamic offloading and resource scheduling for mobile-edge computing with energy harvesting devices, IEEE Trans. Netw. Serv. Manag. 18 (2) (2021) 2154–2165.

[68] X. Cheng, J. Liu, Z. Jin, Efficient deep learning approach for computational offloading in mobile edge computing networks, Wirel. Commun. Mob. Comput. 2022 (2022).

[69] L. Zhang, Z.-Y. Zhang, L. Min, C. Tang, H.-Y. Zhang, Y.-H. Wang, P. Cai, Task offloading and trajectory control for UAV-assisted mobile edge computing using deep reinforcement learning, IEEE Access 9 (2021) 53708–53719.

[70] C.-H. Chu, Task offloading based on deep learning for blockchain in mobile edge computing, Wirel. Netw. 27 (1) (2021) 117–127.

[71] C. Peng, X. Huang, Y. Wu, J. Kang, Constrained multi-objective optimization for UAV-enabled mobile edge computing: Offloading optimization and path planning, IEEE Wirel. Commun. Lett. (2022).

[72] T. Ji, C. Luo, L. Yu, Q. Wang, S. Chen, A. Thapa, P. Li, Energy-efficient computation offloading in mobile edge computing systems with uncertainties, IEEE Trans. Wirel. Commun. 21 (8) (2022) 5717–5729.

[73] L. Ale, N. Zhang, X. Fang, X. Chen, S. Wu, L. Li, Delay-aware and energy-efficient computation offloading in mobile-edge computing using deep reinforcement learning, IEEE Trans. Cogn. Commun. Netw. 7 (3) (2021) 881–892.

[74] H. Zhou, K. Jiang, X. Liu, X. Li, V.C.M. Leung, Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing, IEEE Internet Things J. 9 (2) (2022) 1517–1530.

[75] Y. Hmimz, T. Chanyour, M. El Ghmary, M.O.C. Malki, Joint radio and local resources optimization for tasks offloading with priority in a mobile edge computing network, Pervasive Mob. Comput. 73 (2021) 101368.

[76] H. Zhang, Y. Yang, X. Huang, C. Fang, P. Zhang, Ultra-low latency multi-task offloading in mobile edge computing, IEEE Access 9 (2021) 32569–32581.

[77] Y. He, D. Zhai, F. Huang, D. Wang, X. Tang, R. Zhang, Joint task offloading, resource allocation, and security assurance for mobile edge computing-enabled UAV-assisted VANETs, Remote Sens. 13 (8) (2021) 1547.

[78] Z. Li, Q. Zhu, Genetic algorithm-based optimization of offloading and resource allocation in mobile-edge computing, Information 11 (2) (2020) 83.

[79] Y. Liu, J. Yan, X. Zhao, Deep reinforcement learning based latency minimization for mobile edge computing with virtualization in maritime UAV communication network, IEEE Trans. Veh. Technol. 71 (4) (2022) 4225–4236.

[80] Y. Jeon, H. Baek, S. Pack, Mobility-aware optimal task offloading in distributed edge computing, in: 2021 International Conference on Information Networking, ICOIN, IEEE, 2021, pp. 65–68.

[81] B. Qiao, C. Liu, J. Liu, Y. Hu, K. Li, K. Li, Task migration computation offloading with low delay for mobile edge computing in vehicular networks, Concurr. Comput.: Pract. Exper. 34 (1) (2022) e6494.

[82] W. Feng, H. Liu, Y. Yao, D. Cao, M. Zhao, Latency-aware offloading for mobile edge computing networks, IEEE Commun. Lett. 25 (8) (2021) 2673–2677.

[83] K. Wang, X. Wang, X. Liu, A high reliable computing offloading strategy using deep reinforcement learning for iovs in edge computing, J. Grid Comput. 19 (2) (2021) 1–15.

[84] T.K. Rodrigues, J. Liu, N. Kato, Offloading decision for mobile multi-access edge computing in a multi-tiered 6G network, IEEE Trans. Emerg. Top. Comput. (2021).

[85] Y. Zuo, S. Jin, S. Zhang, Y. Zhang, Blockchain storage and computation offloading for cooperative mobile-edge computing, IEEE Internet Things J. 8 (11) (2021) 9084–9098.

[86] A. Shahidinejad, F. Farahbakhsh, M. Ghobaei-Arani, M.H. Malik, T. Anwar, Context-aware multi-user offloading in mobile edge computing: a federated learning-based approach, J. Grid Comput. 19 (2) (2021) 1–23.

[87] L. Zhao, K. Yang, Z. Tan, H. Song, A. Al-Dubai, A.Y. Zomaya, X. Li, Vehicular computation offloading for industrial mobile edge computing, IEEE Trans. Ind. Inform. 17 (11) (2021) 7871–7881.

[88] P.W. Khan, K. Abbas, H. Shaiba, A. Muthanna, A. Abuarqoub, M. Khayyat, Energy efficient computation offloading mechanism in multi-server mobile edge computing an integer linear optimization approach, Electronics 9 (6) (2020) 1010.

[89] S. Song, S. Ma, J. Zhao, F. Yang, L. Zhai, Cost-efficient multi-service task offloading scheduling for mobile edge computing, Appl. Intell. 52 (4) (2022) 4028–4040.

[90] J. Liang, K. Li, C. Liu, K. Li, Joint offloading and scheduling decisions for DAG applications in mobile edge computing, Neurocomputing 424 (2021) 160–171.

[91] T. Zhang, W. Chen, Computation offloading in heterogeneous mobile edge computing with energy harvesting, IEEE Trans. Green Commun. Netw. 5 (1) (2021) 552–565.

[92] X. Zhu, M. Zhou, Multiobjective optimized cloudlet deployment and task offloading for mobile-edge computing, IEEE Internet Things J. 8 (20) (2021) 15582–15595.

[93] M. Mukherjee, V. Kumar, A. Lat, M. Guo, R. Matam, Y. Lv, Distributed deep learning-based task offloading for UAV-enabled mobile edge computing, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, IEEE, 2020, pp. 1208–1212.

[94] C. Li, Q. Cai, C. Zhang, B. Ma, Y. Luo, Computation offloading and service allocation in mobile edge computing, J. Supercomput. 77 (12) (2021) 13933–13962.

[95] H. Ke, H. Wang, H. Zhao, W. Sun, Deep reinforcement learning-based computation offloading and resource allocation in security-aware mobile edge computing, Wirel. Netw. 27 (5) (2021) 3357–3373.

[96] Q. You, B. Tang, Efficient task offloading using particle swarm optimization algorithm in edge computing for industrial internet of things, J. Cloud Comput. 10 (1) (2021) 1–11.

[97] Q. Li, S. Wang, A. Zhou, X. Ma, F. Yang, A.X. Liu, QoS driven task offloading with statistical guarantee in mobile edge computing, IEEE Trans. Mob. Comput. 21 (1) (2020) 278–290.

[98] T. Liu, Y. Zhang, Y. Zhu, W. Tong, Y. Yang, Online computation offloading and resource scheduling in mobile-edge computing, IEEE Internet Things J. 8 (8) (2021) 6649–6664.

[99] S. Lai, R. Zhao, S. Tang, J. Xia, F. Zhou, L. Fan, Intelligent secure mobile edge computing for beyond 5G wireless networks, Phys. Commun. 45 (2021) 101283.

[100] G. Feng, X. Li, Z. Gao, C. Wang, H. Lv, Q. Zhao, Multi-path and multi-hop task offloading in mobile ad hoc networks, IEEE Trans. Veh. Technol. 70 (6) (2021) 5347–5361.

[101] A. Sacco, F. Esposito, G. Marchetto, P. Montuschi, Sustainable task offloading in UAV networks via multi-agent reinforcement learning, IEEE Trans. Veh. Technol. 70 (5) (2021) 5003–5015.

[102] G. Qu, H. Wu, R. Li, P. Jiao, DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing, IEEE Trans. Netw. Serv. Manag. 18 (3) (2021) 3448–3459.

[103] M. Guo, X. Huang, W. Wang, B. Liang, Y. Yang, L. Zhang, L. Chen, Hagp: A heuristic algorithm based on greedy policy for task offloading with reliability of mds in mec of the industrial internet, Sensors 21 (10) (2021) 3513.

[104] A. Abbas, A. Raza, F. Aadil, M. Maqsood, Meta-heuristic-based offloading task optimization in mobile edge computing, Int. J. Distrib. Sens. Netw. 17 (6) (2021) 55–75.

[105] H. Liao, Z. Zhou, X. Zhao, Y. Wang, Learning-based queue-aware task offloading and resource allocation for space–air–ground-integrated power IoT, IEEE Internet Things J. 8 (7) (2021) 5250–5263.

[106] J. Chen, H. Xing, Z. Xiao, L. Xu, T. Tao, A DRL agent for jointly optimizing computation offloading and resource allocation in MEC, IEEE Internet Things J. 8 (24) (2021) 17508–17524.

[107] Z. Yu, Y. Gong, S. Gong, Y. Guo, Joint task offloading and resource allocation in UAV-enabled mobile edge computing, IEEE Internet Things J. 7 (4) (2020) 3147–3159.

[108] R. Wang, Y. Cao, A. Noor, T.A. Alamoudi, R. Nour, Agent-enabled task offloading in UAV-aided mobile edge computing, Comput. Commun. 149 (2020) 324–331.

[109] X. Huang, G. Huang, Joint optimization of energy and task scheduling in wireless-powered IRS-assisted mobile edge computing systems, IEEE Internet Things J. (2023).

[110] S. Dong, P. Wang, K. Abbas, A survey on deep learning and its applications, Comput. Sci. Rev. 40 (2021) 100379.

[111] S. Bi, L. Huang, H. Wang, Y.-J.A. Zhang, Lyapunov-guided deep reinforcement learning for stable online computation offloading in mobile-edge computing networks, IEEE Trans. Wireless Commun. 20 (11) (2021) 7519–7537.

[112] C. Chen, R. Guo, W. Zhang, J. Yang, C.K. Yeo, Optimal sequential relay-remote selection and computation offloading in mobile edge computing, J. Supercomput. 78 (1) (2022) 1093–1116.

[113] Y. Sun, C. Song, S. Yu, Y. Liu, H. Pan, P. Zeng, Energy-efficient task offloading based on differential evolution in edge computing system with energy harvesting, IEEE Access 9 (2021) 16383–16391.

[114] B. Shang, L. Liu, Z. Tian, Deep learning-assisted energy-efficient task offloading in vehicular edge computing systems, IEEE Trans. Veh. Technol. 70 (9) (2021) 9619–9624.

[115] D. Zhu, H. Liu, T. Li, J. Sun, J. Liang, H. Zhang, L. Geng, Y. Liu, Deep reinforcement learning-based task offloading in satellite-terrestrial edge computing networks, in: 2021 IEEE Wireless Communications and Networking Conference, WCNC, IEEE, 2021, pp. 1–7.

[116] Z. Ali, Z.H. Abbas, G. Abbas, A. Numani, M. Bilal, Smart computational offloading for mobile edge computing in next-generation Internet of Things networks, Comput. Netw. 198 (2021) 108356.

[117] X. Liu, J. Liu, H. Wu, Energy-efficient task allocation of heterogeneous resources in mobile edge computing, IEEE Access 9 (2021) 119700–119711.

[118] A. Ali, M.M. Iqbal, H. Jamil, F. Qayyum, S. Jabbar, O. Cheikhrouhou, M. Baz, F. Jamil, An efficient dynamic-decision based task scheduler for task offloading optimization and energy management in mobile cloud computing, Sensors 21 (13) (2021) 4527.

[119] X. Chen, G. Liu, Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks, IEEE Internet Things J. 8 (13) (2021) 10843–10856.

[120] P. Lin, Q. Song, F.R. Yu, D. Wang, L. Guo, Task offloading for wireless VR-enabled medical treatment with blockchain security using collective reinforcement learning, IEEE Internet Things J. 8 (21) (2021) 15749–15761.
[121] F. Wang, J. Xu, S. Cui, Optimal energy allocation and task offloading policy for wireless powered mobile edge computing systems, IEEE Trans. Wireless Commun. 19 (4) (2020) 2443–2459.
[122] C. Xu, G. Zheng, X. Zhao, Energy-minimization task offloading and resource allocation for mobile edge computing in NOMA heterogeneous networks, IEEE Trans. Veh. Technol. 69 (12) (2020) 16001–16016.
[123] Z. Ali, L. Jiao, T. Baker, G. Abbas, Z.H. Abbas, S. Khaf, A deep learning approach for energy efficient computational offloading in mobile edge computing, IEEE Access 7 (2019) 149623–149633.
[124] S.K.u. Zaman, A.I. Jehangiri, T. Maqsood, N.u. Haq, A.I. Umar, J. Shuja, Z. Ahmad, I.B. Dhaou, M.F. Alsharekh, LiMPO: Lightweight mobility prediction and offloading framework using machine learning for mobile edge computing, Cluster Comput. 26 (1) (2023) 99–117.
[125] A. Naouri, H. Wu, N.A. Nouri, S. Dhelim, H. Ning, A novel framework for mobile-edge computing by optimizing task offloading, IEEE Internet Things J. 8 (16) (2021) 13065–13076.
[126] Y. Wang, W. Fang, Y. Ding, N. Xiong, Computation offloading optimization for UAV-assisted mobile edge computing: a deep deterministic policy gradient approach, Wirel. Netw. 27 (4) (2021) 2991–3006.
[127] G. Zhang, S. Zhang, W. Zhang, Z. Shen, L. Wang, Joint service caching, computation offloading and resource allocation in mobile edge computing systems, IEEE Trans. Wireless Commun. 20 (8) (2021) 5288–5300.
[128] X. Hao, R. Zhao, T. Yang, Y. Hu, B. Hu, Y. Qiu, A risk-sensitive task offloading strategy for edge computing in industrial Internet of Things, EURASIP J. Wireless Commun. Networking 2021 (1) (2021) 1–18.
[129] C. Chen, Y. Zhang, Z. Wang, S. Wan, Q. Pei, Distributed computation offloading method based on deep reinforcement learning in ICV, Appl. Soft Comput. 103 (2021) 107108.
[130] K. Peng, J. Nie, N. Kumar, C. Cai, J. Kang, Z. Xiong, Y. Zhang, Joint optimization of service chain caching and task offloading in mobile edge computing, Appl. Soft Comput. 103 (2021) 107142.
[131] J. Chen, S. Chen, S. Luo, Q. Wang, B. Cao, X. Li, An intelligent task offloading algorithm (iTOA) for UAV edge computing network, Digit. Commun. Netw. 6 (4) (2020) 433–443.
[132] M. Tang, V.W. Wong, Deep reinforcement learning for task offloading in mobile edge computing systems, IEEE Trans. Mob. Comput. (2020).
[133] J. Wang, J. Hu, G. Min, A.Y. Zomaya, N. Georgalas, Fast adaptive task offloading in edge computing based on meta reinforcement learning, IEEE Trans. Parallel Distrib. Syst. 32 (1) (2020) 242–253.
[134] H. Guo, J. Liu, J. Ren, Y. Zhang, Intelligent task offloading in vehicular edge computing networks, IEEE Wirel. Commun. 27 (4) (2020) 126–132.
[135] L. Huang, S. Bi, Y.-J.A. Zhang, Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks, IEEE Trans. Mob. Comput. 19 (11) (2019) 2581–2593.
[136] M. Chen, Y. Hao, Task offloading for mobile edge computing in software defined ultra-dense network, IEEE J. Sel. Areas Commun. 36 (3) (2018) 587–597.
[137] L. Liao, Y. Lai, F. Yang, W. Zeng, Online computation offloading with double reinforcement learning algorithm in mobile edge computing, J. Parallel Distrib. Comput. 171 (2023) 28–39.
[138] L. Yang, H. Zhang, X. Li, H. Ji, V.C. Leung, A distributed computation offloading strategy in small-cell networks integrated with mobile edge computing, IEEE/ACM Trans. Netw. 26 (6) (2018) 2762–2773.
[139] J. Xu, L. Chen, P. Zhou, Joint service caching and task offloading for mobile edge computing in dense networks, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 207–215.
[140] Z. Lv, D. Chen, R. Lou, Q. Wang, Intelligent edge computing based on machine learning for smart city, Future Gener. Comput. Syst. 115 (2021) 90–99.
[141] J. Zhou, D. Tian, Z. Sheng, X. Duan, X. Shen, Distributed task offloading optimization with queueing dynamics in multiagent mobile-edge computing networks, IEEE Internet Things J. 8 (15) (2021) 12311–12328.
[142] W. Li, S. Jin, Performance evaluation and optimization of a task offloading strategy on the mobile edge computing with edge heterogeneity, J. Supercomput. 77 (11) (2021) 12486–12507.
[143] S. Luo, H. Li, Z. Wen, B. Qian, G. Morgan, A. Longo, O. Rana, R. Ranjan, Blockchain-based task offloading in drone-aided mobile edge computing, IEEE Netw. 35 (1) (2021) 124–129.
[144] S. Wang, J. Li, G. Wu, H. Chen, S. Sun, Joint optimization of task offloading and resource allocation based on differential privacy in vehicular edge computing, IEEE Trans. Comput. Soc. Syst. 9 (1) (2022) 109–119.
[145] Y. Qi, Y. Zhou, Y.-F. Liu, L. Liu, Z. Pan, Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing, IEEE Internet Things J. 8 (24) (2021) 17762–17777.
[146] C. Ma, J. Zhu, M. Liu, H. Zhao, N. Liu, X. Zou, Parking edge computing: parked-vehicle-assisted task offloading for urban VANETs, IEEE Internet Things J. 8 (11) (2021) 9344–9358.
[147] H. Wang, H. Xu, H. Huang, M. Chen, S. Chen, Robust task offloading in dynamic edge computing, IEEE Trans. Mob. Comput. 22 (1) (2023) 500–514.
[148] X. Ma, A. Zhou, Q. Sun, S. Wang, Freshness-aware information update and computation offloading in mobile-edge computing, IEEE Internet Things J. 8 (16) (2021) 13115–13125.
[149] A. Asheralieva, D. Niyato, Fast and secure computational offloading with lagrange coded mobile edge computing, IEEE Trans. Veh. Technol. 70 (5) (2021) 4924–4942.
[150] L. Leng, J. Li, H. Shi, Y. Zhu, Graph convolutional network-based reinforcement learning for tasks offloading in multi-access edge computing, Multimedia Tools Appl. 80 (19) (2021) 29163–29175.
[151] X. Huang, X. Yang, Q. Chen, J. Zhang, Task offloading optimization for UAV-assisted Fog-enabled Internet of Things networks, IEEE Internet Things J. 9 (2) (2021) 1082–1094.
[152] P.X. Nguyen, D.-H. Tran, O. Onireti, P.T. Tin, S.Q. Nguyen, S. Chatzinotas, H.V. Poor, Backscatter-assisted data offloading in OFDMA-based wireless-powered mobile edge computing for IoT networks, IEEE Internet Things J. 8 (11) (2021) 9233–9243.
[153] I. Alghamdi, C. Anagnostopoulos, D.P. Pezaros, Data quality-aware task offloading in mobile edge computing: an optimal stopping theory approach, Future Gener. Comput. Syst. 117 (2021) 462–479.
[154] J. Xue, Q. Hu, Y. An, L. Wang, Joint task offloading and resource allocation in vehicle-assisted multi-access edge computing, Comput. Commun. 177 (2021) 77–85.
[155] K. Fu, J. Ye, Computation offloading based on improved glowworm swarm optimization algorithm in mobile edge computing, in: Journal of Physics: Conference Series, vol. 1757, IOP Publishing, 2021, 012195.
[156] T. Alfakih, M.M. Hassan, A. Gumaei, C. Savaglio, G. Fortino, Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA, IEEE Access 8 (2020) 54074–54084.
[157] S. Raza, W. Liu, M. Ahmed, M.R. Anwar, M.A. Mirza, Q. Sun, S. Wang, An efficient task offloading scheme in vehicular edge computing, J. Cloud Comput. 9 (1) (2020) 1–14.
[158] H. Guo, J. Liu, J. Zhang, Computation offloading for multi-access mobile edge computing in ultra-dense networks, IEEE Commun. Mag. 56 (8) (2018) 14–19.
[159] T.X. Tran, D. Pompili, Joint task offloading and resource allocation for multi-server mobile-edge computing networks, IEEE Trans. Veh. Technol. 68 (1) (2018) 856–868.
[160] J. Liu, K. Xue, Q. Miao, S. Li, X. Cui, D. Wang, K. Li, Mcvco: Multi-mec cooperative vehicular computation offloading, IEEE Trans. Intell. Veh. (2023).
[161] 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, Appl. Soft Comput. 112 (2021) 107790.
[162] L. Xiao, C. Xie, T. Chen, H. Dai, H.V. Poor, A mobile offloading game against smart attacks, IEEE Access 4 (2016) 2281–2291, http://dx.doi.org/10.1109/ACCESS.2016.2565198.
[163] S. Ko, K. Huang, S. Kim, H. Chae, Energy efficient mobile computation offloading via online prefetching, in: IEEE International Conference on Communications, ICC 2017, Paris, France, May 21-25, 2017, IEEE, 2017, pp. 1–6, http://dx.doi.org/10.1109/ICC.2017.7997341.
[164] A.R. Benamer, H. Teyeb, N.B. Hadj-Alouane, Latency-aware placement heuristic in fog computing environment, in: H. Panetto, C. Debruyne, H.A. Proper, C.A. Ardagna, D. Roman, R. Meersman (Eds.), On the Move to Meaningful Internet Systems. OTM 2018 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Malta, October 22-26, 2018, Proceedings, Part II, in: Lecture Notes in Computer Science, vol. 11230, Springer, 2018, pp. 241–257, http://dx.doi.org/10.1007/978-3-030-02671-4_14.
[165] X. Zhang, Y. Cao, Mobile data offloading efficiency: A stochastic analytical view, in: 2018 IEEE International Conference on Communications Workshops, ICC Workshops 2018, Kansas City, MO, USA, May 20-24, 2018, IEEE, 2018, pp. 1–6, http://dx.doi.org/10.1109/ICCW.2018.8403702.
[166] Y. He, N. Zhao, H. Yin, Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach, IEEE Trans. Veh. Technol. 67 (1) (2018) 44–55, http://dx.doi.org/10.1109/TVT.2017.2760281.
[167] L.T. Tan, R.Q. Hu, Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning, IEEE Trans. Veh. Technol. 67 (11) (2018) 10190–10203, http://dx.doi.org/10.1109/TVT.2018.2867191.
[168] H. Wu, K. Wolter, Stochastic analysis of delayed mobile offloading in heterogeneous networks, IEEE Trans. Mob. Comput. 17 (2) (2018) 461–474, http://dx.doi.org/10.1109/TMC.2017.2711014.
[169] S. Ko, K. Han, K. Huang, Wireless networks for mobile edge computing: Spatial modeling and latency analysis, IEEE Trans. Wirel. Commun. 17 (8) (2018) 5225–5240, http://dx.doi.org/10.1109/TWC.2018.2840120.
[170] N. Kherraf, H.A. Alameddine, S. Sharafeddine, C.M. Assi, A. Ghrayeb, Optimized provisioning of edge computing resources with heterogeneous workload in IoT networks, IEEE Trans. Netw. Serv. Manag. 16 (2) (2019) 459–474, http://dx.doi.org/10.1109/TNSM.2019.2894955.
[171] F. Aït-Salaht, F. Desprez, A. Lebre, C. Prud'homme, M. Abderrahim, Service placement in fog computing using constraint programming, in: E. Bertino, C.K. Chang, P. Chen, E. Damiani, M. Goul, K. Oyama (Eds.), 2019 IEEE International Conference on Services Computing, SCC 2019, Milan, Italy, July 8-13, 2019, IEEE, 2019, pp. 19–27, http://dx.doi.org/10.1109/SCC.2019.00017.

[172] D. Munoz, J.A. Montenegro, M. Pinto, L. Fuentes, Energy-aware environments for the development of green applications for cyber-physical systems, Future Gener. Comput. Syst. 91 (2019) 536–554, http://dx.doi.org/10.1016/J.FUTURE.2018.09.006.

[173] A. Yousefpour, A. Patil, G. Ishigaki, I. Kim, X. Wang, H.C. Cankaya, Q. Zhang, W. Xie, J.P. Jue, FOGPLAN: A lightweight QoS-aware dynamic fog service provisioning framework, IEEE Internet Things J. 6 (3) (2019) 5080–5096, http://dx.doi.org/10.1109/JIOT.2019.2896311.

[174] S. Pasteris, S. Wang, M. Herbster, T. He, Service placement with provable guarantees in heterogeneous edge computing systems, in: 2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019, IEEE, 2019, pp. 514–522, http://dx.doi.org/10.1109/INFOCOM.2019.8737449.

[175] P. Kayal, J. Liebeherr, Autonomic service placement in fog computing, in: 20th IEEE International Symposium on "A World of Wireless, Mobile and Multimedia Networks", WoWMoM 2019, Washington, DC, USA, June 10-12, 2019, IEEE, 2019, pp. 1–9, http://dx.doi.org/10.1109/WOWMOM.2019.8792989.

[176] Q.T. Minh, D.T. Nguyen, V.A. Le, D.H. Nguyen, T.V. Pham, Task placement on fog computing made efficient for IoT application provision, Wirel. Commun. Mob. Comput. 2019 (2019) 6215454:1–6215454:17, http://dx.doi.org/10.1155/2019/6215454.

[177] Y. Qian, L. Hu, J. Chen, X. Guan, M.M. Hassan, A. Alelaiwi, Privacy-aware service placement for mobile edge computing via federated learning, Inform. Sci. 505 (2019) 562–570, http://dx.doi.org/10.1016/J.INS.2019.07.069.

[178] A.-Y. Son, E.-N. Huh, Multi-objective service placement scheme based on fuzzy-AHP system for distributed cloud computing, Appl. Sci. 9 (17) (2019) 3550.

[179] C. Lin, T. Wang, K. Chen, B. Lee, J. Kuo, Distributed deep neural network deployment for smart devices from the edge to the cloud, in: Proceedings of the ACM MobiHoc Workshop on Pervasive Systems in the IoT Era, PERSIST-IoT@MobiHoc 2019, Catania, Italy, July 2, 2019, ACM, 2019, pp. 43–48, http://dx.doi.org/10.1145/3331052.3332477.

[180] X. Zhao, K. Yang, Q. Chen, D. Peng, H. Jiang, X. Xu, X. Shuang, Deep learning based mobile data offloading in mobile edge computing systems, Future Gener. Comput. Syst. 99 (2019) 346–355, http://dx.doi.org/10.1016/J.FUTURE.2019.04.039.

[181] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, W. Zhuang, Learning-based computation offloading for IoT devices with energy harvesting, IEEE Trans. Veh. Technol. 68 (2) (2019) 1930–1941, http://dx.doi.org/10.1109/TVT.2018.2890685.

[182] V. Yadav, N.B. V., R.M.R. Guddeti, GA-PSO: service allocation in fog computing environment using hybrid bio-inspired algorithm, in: TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), Kochi, India, October 17-20, 2019, IEEE, 2019, pp. 1280–1285, http://dx.doi.org/10.1109/TENCON.2019.8929234.

[183] C. Canali, R. Lancellotti, GASP: genetic algorithms for service placement in fog computing systems, Algorithms 12 (10) (2019) 201, http://dx.doi.org/10.3390/A12100201.

[184] L. Liu, H. Tan, S.H. Jiang, Z. Han, X. Li, H. Huang, Dependent task placement and scheduling with function configuration in edge computing, in: Proceedings of the International Symposium on Quality of Service, IWQoS 2019, Phoenix, AZ, USA, June 24-25, 2019, ACM, 2019, pp. 20:1–20:10, http://dx.doi.org/10.1145/3326285.3329055.

[185] W. Zhou, W. Fang, Y. Li, B. Yuan, Y. Li, T. Wang, Markov approximation for task offloading and computation scaling in mobile edge computing, Mob. Inf. Syst. 2019 (2019) 8172698:1–8172698:12, http://dx.doi.org/10.1155/2019/8172698.

[186] J. Zhou, D. Tian, Y. Wang, Z. Sheng, X. Duan, V.C.M. Leung, Reliability-oriented optimization of computation offloading for cooperative vehicle-infrastructure systems, IEEE Signal Process. Lett. 26 (1) (2019) 104–108, http://dx.doi.org/10.1109/LSP.2018.2880081.

[187] G. Tanganelli, L. Cassano, A. Miele, C. Vallati, A methodology for the design and deployment of distributed cyber-physical systems for smart environments, Future Gener. Comput. Syst. 109 (2020) 420–430, http://dx.doi.org/10.1016/J.FUTURE.2020.02.047.

[188] D. Zeng, L. Gu, H. Yao, Towards energy efficient service composition in green energy powered Cyber-Physical Fog Systems, Future Gener. Comput. Syst. 105 (2020) 757–765, http://dx.doi.org/10.1016/J.FUTURE.2018.01.060.

[189] I. Shaer, A. Haque, A. Shami, Multi-component V2X applications placement in edge computing environment, in: 2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, June 7-11, 2020, IEEE, 2020, pp. 1–6, http://dx.doi.org/10.1109/ICC40277.2020.9148960.

[190] X. Xu, X. Liu, Z. Xu, F. Dai, X. Zhang, L. Qi, Trust-oriented IoT service placement for smart cities in edge computing, IEEE Internet Things J. 7 (5) (2020) 4084–4091, http://dx.doi.org/10.1109/JIOT.2019.2959124.

[191] Z.Á. Mann, Secure software placement and configuration, Future Gener. Comput. Syst. 110 (2020) 243–253, http://dx.doi.org/10.1016/J.FUTURE.2020.03.064.

[192] X. Chen, T. Chen, Z. Zhao, H. Zhang, M. Bennis, Y. Ji, Resource awareness in unmanned aerial vehicle-assisted mobile-edge computing systems, in: 91st IEEE Vehicular Technology Conference, VTC Spring 2020, Antwerp, Belgium, May 25-28, 2020, IEEE, 2020, pp. 1–6, http://dx.doi.org/10.1109/VTC2020-SPRING48590.2020.9128981.

[193] K. Li, Quantitative modeling and analytical calculation of elasticity in cloud computing, IEEE Trans. Cloud Comput. 8 (4) (2020) 1135–1148, http://dx.doi.org/10.1109/TCC.2017.2665549.

[194] Y. He, C. Liang, F.R. Yu, Z. Han, Trust-based social networks with computing, caching and communications: A deep reinforcement learning approach, IEEE Trans. Netw. Sci. Eng. 7 (1) (2020) 66–79, http://dx.doi.org/10.1109/TNSE.2018.2865183.

[195] Y. Liang, J. Ge, S. Zhang, J. Wu, L. Pan, T. Zhang, B. Luo, Interaction-oriented service entity placement in edge computing, IEEE Trans. Mob. Comput. 20 (3) (2021) 1064–1075, http://dx.doi.org/10.1109/TMC.2019.2952097.

[196] N. Wang, B. Varghese, Context-aware distribution of fog applications using deep reinforcement learning, J. Netw. Comput. Appl. 203 (2022) 103354, http://dx.doi.org/10.1016/J.JNCA.2022.103354.

[197] D.T. Nguyen, H.T. Nguyen, N. Trieu, V.K. Bhargava, Two-stage robust edge service placement and sizing under demand uncertainty, IEEE Internet Things J. 9 (2) (2022) 1560–1574, http://dx.doi.org/10.1109/JIOT.2021.3090442.

[198] A. Shakarami, M. Ghobaei-Arani, A. Shahidinejad, A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective, Comput. Netw. 182 (2020) 107496.

[199] C. Feng, P. Han, X. Zhang, B. Yang, Y. Liu, L. Guo, Computation offloading in mobile edge computing networks: A survey, J. Netw. Comput. Appl. 202 (2022) 103366, http://dx.doi.org/10.1016/J.JNCA.2022.103366.

[200] K. Li, Quantitative modeling and analytical calculation of elasticity in cloud computing, IEEE Trans. Cloud Comput. 8 (4) (2020) 1135–1148, http://dx.doi.org/10.1109/TCC.2017.2665549.

[201] M.L. Shooman, Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design, John Wiley & Sons, 2003.

[202] N. Thomas, M. Forshaw (Eds.), Analytical and Stochastic Modelling Techniques and Applications - 24th International Conference, ASMTA 2017, Newcastle-upon-Tyne, UK, July 10-11, 2017, Proceedings, in: Lecture Notes in Computer Science, vol. 10378, Springer, 2017, http://dx.doi.org/10.1007/978-3-319-61428-1.

[203] G.H.S. Carvalho, I. Woungang, A. Anpalagan, E. Degtiareva, J.J.P.C. Rodrigues, A Semi-Markov Decision Model-based brokering mechanism for mobile cloud market, in: IEEE International Conference on Communications, ICC 2017, Paris, France, May 21-25, 2017, IEEE, 2017, pp. 1–6, http://dx.doi.org/10.1109/ICC.2017.7997256.

[204] A. Shakarami, M. Ghobaei-Arani, M. Masdari, M. Hosseinzadeh, A survey on the computation offloading approaches in mobile edge/cloud computing environment: A stochastic-based perspective, J. Grid Comput. 18 (4) (2020) 639–671, http://dx.doi.org/10.1007/S10723-020-09530-2.

[205] C. Zhang, Z. Zheng, Task migration for mobile edge computing using deep reinforcement learning, Future Gener. Comput. Syst. 96 (2019) 111–118, http://dx.doi.org/10.1016/J.FUTURE.2019.01.059.

[206] J. Esch, Software-defined networking: A comprehensive survey, Proc. IEEE 103 (1) (2015) 10–13, http://dx.doi.org/10.1109/JPROC.2014.2374752.

[207] B. Guo, C. Chen, D. Zhang, Z. Yu, A. Chin, Mobile crowd sensing and computing: when participatory sensing meets participatory social media, IEEE Commun. Mag. 54 (2) (2016) 131–137, http://dx.doi.org/10.1109/MCOM.2016.7402272.

[208] Y. Zhang, Z. Zhao, C. Shu, G. Min, Z. Wang, Embedding virtual network functions with backup for reliable large-scale edge computing, in: M. Qiu (Ed.), 5th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2018 / 4th IEEE International Conference on Edge Computing and Scalable Cloud, EdgeCom 2018, Shanghai, China, June 22-24, 2018, IEEE Computer Society, 2018, pp. 190–195, http://dx.doi.org/10.1109/CSCLOUD/EDGECOM.2018.00041.

[209] F. Cicirelli, A. Guerrieri, G. Spezzano, A. Vinci, O. Briante, A. Iera, G. Ruggeri, Edge computing and social internet of things for large-scale smart environments development, IEEE Internet Things J. 5 (4) (2018) 2557–2571, http://dx.doi.org/10.1109/JIOT.2017.2775739.

[210] N.C. Luong, P. Wang, D. Niyato, Y. Wen, Z. Han, Resource management in cloud networking using economic analysis and pricing models: A survey, IEEE Commun. Surv. Tutor. 19 (2) (2017) 954–1001, http://dx.doi.org/10.1109/COMST.2017.2647981.

[211] H. Wu, Multi-objective decision-making for mobile cloud offloading: A survey, IEEE Access 6 (2018) 3962–3976, http://dx.doi.org/10.1109/ACCESS.2018.2791504.

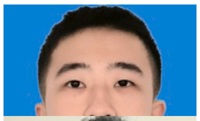
[212] F. Jazayeri, A. Shahidinejad, M. Ghobaei-Arani, A latency-aware and energy-efficient computation offloading in mobile fog computing: a hidden Markov model-based approach, J. Supercomput. 77 (5) (2021) 4887–4916, http://dx.doi.org/10.1007/S11227-020-03476-8.

[213] M. Salimian, M. Ghobaei-Arani, A. Shahidinejad, An evolutionary multi-objective optimization technique to deploy the IoT services in fog-enabled networks: An autonomous approach, Appl. Artif. Intell. 36 (1) (2022) http://dx.doi.org/10.1080/08839514.2021.2008149.

[214] Y. Wang, P. Lang, D. Tian, J. Zhou, X. Duan, Y. Cao, D. Zhao, A game-based computation offloading method in vehicular multiaccess edge computing networks, IEEE Internet Things J. 7 (6) (2020) 4987–4996, http://dx.doi.org/10.1109/JIOT.2020.2972061.

[215] M. Liu, Y. Liu, Price-based distributed offloading for mobile-edge computing with computation capacity constraints, IEEE Wirel. Commun. Lett. 7 (3) (2018) 420–423, http://dx.doi.org/10.1109/LWC.2017.2780128.

**Shi Dong** received Ph.D in computer science from Southeast University in 2013. He is a Visiting Postdoctoral fellow in Washington University in St.Louis in 2016, Visiting Professor in the University of Melbourne in 2024, and a professor with Zhoukou Normal University. He has published more than 100 papers in top conferences and journals. He is a reviewer of IEEE Transactions on Parallel and Distributed Systems, IEEE/ACM Transactions on Networking; Computer Networks; IEEE Transactions on Network Science and Engineering; IEEE Transactions on Industrial Informatics; IEEE Internet of Things Journal; Information Processing & Management. He also is an associate Editor for IEEE Systems Journal, Physical Communication, IET Wireless Sensor Systems, IEICE Transactions on Communications, and International Journal on Artificial Intelligence Tools. His major interests are network management, network security.



**Junxiao Tang** received BS in computer science from Zhoukou Normal University in 2022. His major interests are metaverse, blockchain, network security.

**Khushnood Abbas** received his bachelor and master's degree from Aligarh Muslim University, India in 2011. Further he has earned his Ph.D. from University of Electronic Science and Technology of China (UESTC) in 2018. He was visiting scholar to Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences. Currently he is working as an Assistant Professor Zhoukou Normal University.

**Ruizhe Hou** received BS in Automation from Tianjin University in 2016. MS in Computer Science from Roland University, Hungary in 2019. Nowadays, he is pursuing PHD in South China University of Technology. His major interests are computer network, network security.

**Joarder Kamruzzaman** received the B.Sc. and M.Sc. degrees in Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology, Dhaka, and the Ph.D. degree in Information Systems Engineering from Muroran Institute of Technology, Hokkaido, Japan. He is a Professor of Information Technology and the Director of the Centre for Smart Analytics at Federation University Australia. His interests include Internet of Things, machine learning and cybersecurity. Kamruzzaman has published 300+ peer-reviewed articles which include over 90 journal and 180 conference papers. He is the recipient of Best Paper award in four international conferences: ICICS15, Singapore; APCC14, Thailand; IEEE WCNC10, Sydney, Australia and IEEE-ICNNSP03, Nanjing, China. He was the founding Program Co-Chair of the First International Symposium on Dependability in Sensor, Cloud, and Big Data Systems and Applications (DependSys), China, in 2015. He has served many conferences in leadership capacities including Program co-Chair, Publicity Chair, Track Chair and Session Chairs, and since 2012 as an Editor of the Elsevier Journal of Network and Computer Applications, and had served as the lead guest editor of Elsevier Journal Future Generation Computer Systems.

**Leszek Rutkowski**(Fellow, IEEE) is with the Institute of Computer Science, AGH University of Science and Technology, 30-059 Krak¨®w, and also with the Systems Research Institute of the Polish Academy of Sciences, 01-447 Warsaw, Poland. He received the M.Sc. and Ph.D. degrees from the Technical University of Wroclaw, Poland, in 1977 and 1980, respectively. Since 1980, he has been with the Technical University of Czestochowa, where he is currently a professor and director of the Institute of Computational Intelligence. From 1987 to 1990, he held a visiting position at the School of Electrical and Computer Engineering, Oklahoma State University. His research interests include data stream mining, big data analysis, neural networks, fuzzy systems, computational intelligence, pattern classification, and expert systems. He has published more than 200 technical papers, including 22 in various series of IEEE Transactions. He is the author of the following books: Computational Intelligence (Springer, 2008), New Soft Computing Techniques for System Modeling, Pattern Classification and Image Processing (Springer, 2004), Flexible Neuro-Fuzzy Systems (Kluwer Academic, 2004), Methods and Techniques of Artificial Intelligence (2005, in Polish), Adaptive Filters and Adaptive Signal Processing (1994, in Polish), and coauthor of two others (1997 and 2000, in Polish) Neural Networks, Genetic Algorithms and Fuzzy Systems and Neural Networks for Image Compression. He is the president and founder of the Polish Neural Networks Society. He organized and served as a General Chair of the International Conferences on Artificial Intelligence and Soft Computing held in 1996, 1997, 1999, 2000, 2002, 2004, 2006, 2008, 2010, 2012, 2013, 2014, 2015, 2016 and 2017. He was an associate editor of the IEEE Transactions on Neural Networks (1998–2005) and IEEE Systems Journal (2007–2010). He is an editor-in-chief of the Journal of Artificial Intelligence and Soft Computing Research, and he is on the editorial board of the IEEE Transactions on Cybernetics, International Journal of Neural Systems, International Journal of Applied Mathematics and Computer Science and International Journal of Biometric. He is a recipient of the IEEE Transactions on Neural Networks 2005 Outstanding Paper Award. He served in the IEEE Computational Intelligence Society as the chair of the Distinguished Lecturer Program (2008–2009) and the chair of the Standards Committee (2006–2007). He is the founding chair of the Polish chapter of the IEEE Computational Intelligence Society, which won the 2008 Outstanding Chapter Award. In 2004, he was elected as a member of the Polish Academy of Sciences. In 2004, he was awarded by the IEEE Fellow membership grade for contributions to neurocomputing and flexible fuzzy systems. He received a degree honoris causa from prestigious AGH University of Science and Technology in Cracow in recognition of outstanding scientific achievements in the field of artificial intelligence in particular neuro-fuzzy systems.

**Rajkumar Buyya**(Fellow, IEEE) is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also serving as the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in Cloud Computing. He served as a Future Fellow of the Australian Research Council during 2012–2016. He serving/served as Honorary/Visiting Professor for several elite Universities including Imperial College London (UK), University of Birmingham (UK), University of Hyderabad (India), and Tsinghua University (China). He has authored over 850 publications and seven text books including "Mastering Cloud Computing " published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets respectively. He also edited several books including "Cloud Computing: Principles and Paradigms " (Wiley Press, USA, Feb 2011). He is one of the highly cited authors in computer science and software engineering worldwide (h-index=162, g-index=360, 141,800+ citations). "A Scientometric Analysis of Cloud Computing Literature " by German scientists ranked Dr. Buyya as the World's Top-Cited (#1) Author and the World's Most-Productive (#1) Author in Cloud Computing. Dr. Buyya is recognized as a "Web of Science Highly Cited Researcher " for seventimes since 2016, a Fellow of IEEE, Foreign Fellow of Academia Europaea, Scopus Researcher of the Year 2017 with Excellence in Innovative Research Award by Elsevier, and "Lifetime Achievement Awards " from two Indian universities for his outstanding contributions to Cloud computing and distributed systems. He has been recognized as the "Best of the World " twice for research fields (in Computing Systems in 2019 and Software Systems in 2021) as well as "Lifetime Achiever " and "Superstar of Research " in "Engineering and Computer Science " discipline twice (2019 and 2021) by the Australian Research Review. Recently, he received "Research Innovation Award " from IEEE Technical Committee on Services Computing and "Research Impact Award " from IEEE Technical Committee on Cloud Computing. Software technologies for Grid and Cloud computing developed under Dr. Buyya's leadership have gained rapid acceptance and are in use at several academic institutions and commercial enterprises in 50+ countries around the world. He acknowledges all researchers and institutions worldwide for their consideration in building on software systems created by his CLOUDS Lab and recognizing them through citations and contributing to their further enhancements. Dr. Buyya has led the establishment and development of key community activities, including serving as foundation Chair of the IEEE Technical Committee on Scalable Computing and five IEEE/ACM conferences. These contributions and international research leadership of Dr. Buyya are recognized through the award of "2009 IEEE Medal for Excellence in Scalable Computing " from the IEEE Computer Society TCSC. Manjrasoft's Aneka Cloud technology developed under his leadership has received "2010 Frost & Sullivan New Product Innovation Award ". Dr. Buyya received "Mahatma Gandhi Award " along with Gold Medals for his outstanding and extraordinary achievements in Information Technology field and services rendered to promote greater friendship and India-International cooperation. He served as the founding Editor-in-Chief of the IEEE Transactions on Cloud Computing. He is currently serving as Co-Editor-in-Chief of Journal of Software: Practice and Experience, which was established 50+ years ago. For further information on Dr.Buyya, please visit his cyberhome:www.buyya.com