Chinese Journal of Electronics
vol. 33, no. 4, pp. 859–874, July 2024
https://doi.org/10.23919/cje.2023.00.264

Special Focus on Multi-Dimensional QoS Provision of Intelligent Edge Computing for IoT

REVIEW

Multi-Dimensional QoS Evaluation and Optimization of Mobile Edge Computing for IoT: A Survey

Jiwei HUANG[1,2], Fangzheng LIU[1,2], and Jianbing ZHANG[1,2]

1. *Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum, Beijing 102249, China*
2. *Department of Computer Science and Technology, China University of Petroleum, Beijing 102249, China*

Corresponding author: Jiwei HUANG, Email: huangjw@cup.edu.cn

Manuscript Received July 29, 2023; Accepted January 23, 2024

Abstract — With the evolvement of the Internet of things (IoT), mobile edge computing (MEC) has emerged as a promising computing paradigm to support IoT data analysis and processing. In MEC for IoT, the differentiated requirements on quality of service (QoS) have been growing rapidly, making QoS a multi-dimensional concept including several attributes, such as performance, dependability, energy efficiency, and economic factors. To guarantee the QoS of IoT applications, theories and techniques of multi-dimensional QoS evaluation and optimization have become important theoretical foundations and supporting technologies for the research and application of MEC for IoT, which have attracted significant attention from both academia and industry. This paper aims to survey the existing studies on multi-dimensional QoS evaluation and optimization of MEC for IoT, and provide insights and guidance for future research in this field. This paper summarizes the multi-dimensional and multi-attribute QoS metrics in IoT scenarios, and then several QoS evaluation methods are presented. For QoS optimization, the main research problems in this field are summarized, and optimization models as well as their corresponding solutions are elaborated. We take notice of the booming of edge intelligence in artificial intelligence-empowered IoT scenarios, and illustrate the new research topics and the state-of-the-art approaches related to QoS evaluation and optimization. We discuss the challenges and future research directions.

Keywords — Internet of things, Mobile edge computing, Quality of service, Quality of service evaluation, Quality of service optimization.

Citation — Jiwei HUANG, Fangzheng LIU, and Jianbing ZHANG, "Multi-Dimensional QoS Evaluation and Optimization of Mobile Edge Computing for IoT: A Survey," *Chinese Journal of Electronics*, vol. 33, no. 4, pp. 859–874, 2024. doi: 10.23919/cje.2023.00.264.

I. Introduction

With the Internet of things (IoT) techniques being applied in increasing aspects of industry and human life, there have been growing requirements on quality of service (QoS). The traditional centralized cloud-based computing infrastructure faces challenges in guaranteeing real-time QoS in IoT scenarios. To this end, mobile edge computing (MEC) has emerged as a novel computing architecture, which makes full use of the computational resources of the communication or computing devices at the edge of the network for processing part of the delay-sensitive tasks. With MEC, the workload of the core network can be reduced and the response time can be decreased especially for data-intensive applications or data-driven services. Furthermore, an emerging advanced architecture namely multi-access MEC can leverage multiple access technologies (cellular, Wi-Fi, Bluetooth, etc.) for handling user tasks with varying network and computing resources. Also, several cutting-edge communication techniques, such as NOMA (non-orthogonal multiple access), SWIPT (simultaneous wireless information and power transfer), and semantic communication, have been integrated into MEC. With the booming of artificial intelligence (AI) technology, MEC has received a tremendous amount of interest from both academia and

Guest Editor: Associate Prof. Yuan WU, University of Macau.

industry.

In MEC for IoT, differentiated requirements on QoS makes QoS a multi-dimensional concept. Firstly, performance is one of the most critical concerns, which includes response time (i.e., end-to-end delay), throughput, utilization, and blocking rate. Secondly, dependability [1], which is the ability of a system to keep on reliable service and avoid failures, becomes an important requirement, especially for critical applications such as traffic control, healthcare and industrial applications. It is an integrating concept that encompasses several attributes, which are availability, reliability, safety, integrity, and maintainability. Thirdly, with the growing scales of IoT systems deployed in reality, energy efficiency is an important consideration. Moreover, pricing, cost of ownership, and other economic factors should also be accounted in QoS provision.

With all these QoS metrics, one of the most fundamental tasks in QoS provision is to precisely evaluate the QoS with acceptable overhead. The most straightforward way is to implant one or several modules with the abilities of data collection and QoS measurement into the real-life running MEC systems. However, such measurement-based approaches require full implementation of the MEC systems before QoS evaluation, which involves significant overhead. To attack this challenge, computer simulation techniques are introduced, which is to design and implement a set of computer programs to simulate or emulate the dynamics of the MEC systems and then collect data for QoS evaluation. Furthermore, analytical models are applied for QoS evaluation, whose fundamental idea is to construct mathematical models and conduct theoretical analysis under some certain circumstances or mathematical assumptions. With simulation-based and analytical approaches, different solutions and policies can be analyzed and compared without being implemented and deployed, and thus those types of approaches are quite economic and efficient especially in the design phase of an MEC system.

After QoS evaluation, the next step is its optimization. It is the foundation of several practical problems in MEC systems, such as task offloading, resource allocation, edge caching, service migration, user allocation, collaborative computing, and resource pricing. The basic procedures of QoS optimization include 1) constructing an optimization model and then 2) solving the model with certain scheme or algorithm. The optimization model can be classified into two categories. The first one is static optimization which can be solved by some mathematical approaches or numerical solutions. The second one is dynamic optimization, where iterative algorithms or intelligent approaches are usually used for solving such type of optimization problem.

Due to the complex hierarchy, high dynamics, data-intensive and large-scale characteristics of MEC infrastructure for IoT, traditional approaches of QoS provision face several challenges, such as state-space explosion in system modeling, high complexity in QoS analysis, trade-off among multiple QoS attributes, and huge search space in QoS optimization. Researchers in this field have made in-depth explorations and achieved certain results in both evaluation theory and optimization techniques.

This paper conducts a comprehensive survey on the existing works in this field, shown as Figure 1. Specifically, multi-dimensional QoS metrics are systematically discussed, and QoS evaluation approaches are reviewed and categorized; then, QoS optimization problems in MEC for IoT are summarized, and the well-known QoS optimization models and their corresponding solutions are presented. Finally, this paper provides some insights on the challenges and future research directions, especially for edge intelligence scenarios.

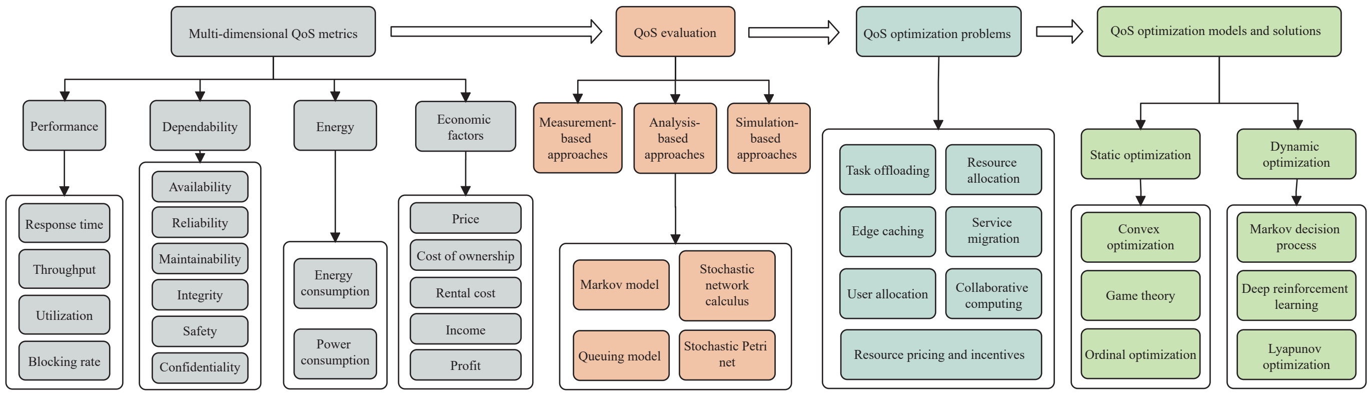

Figure 1 An overview of multi-dimensional QoS evaluation and optimization of mobile edge computing for IoT.

Although there have been several surveys on MEC, they usually focused on some specific issues (such as task offloading [2], incentive mechanisms [3], and auction schemes [4]), or studied some single specific QoS metric (such as energy consumption [5] or safety [6]). This paper aims to study the QoS issue from a multi-dimensional viewpoint, and discuss the state-of-the-art approaches on QoS evaluation and optimization. Also, this paper tries to discuss the challenges in QoS management brought by emerging technologies in the era of edge intelligence, expected to inspire further research in the future.

II. Multi-Dimensional QoS Metrics

With the wide application of IoT, diverse requirements for QoS of MEC systems have been raised. We summarize the QoS metrics into four dimensions, i.e., performance, dependability, energy, and economic factors. In each of the dimensions, there are also several attributes. We illustrate their definitions as follows.

1. Performance

Performance reflects the capability or efficiency of a system providing services. It can be regarded as one of the most important aspects of QoS in MEC. It includes response time, throughput, utilization, and blocking rate [7].

• Response time: Response time is the time between a request for service and the fulfillment of that request. Sometimes, it can also be known as end-to-end delay. In MEC for IoT, the response time can include several parts, which are 1) communication delay for sending the request and the results, 2) waiting time when the request is queued in a buffer, and 3) execution time at the server (or processors). Response time is usually measured in milliseconds (ms), but may be measured in microseconds (μs) in some high-performance systems. It may range from 0 to infinity.

• Throughput: Throughput is the number of services that an MEC system can perform in a unit time. It is a reference metric to evaluate the maximum service capacity of the system, and is usually measured in bits per second (bps), ranging from 0 to infinity.

• Utilization: Utilization reflects the usage of system resources, which can be expressed by the ratio of hardware and software resources being used in the service process, ranging from 0 to 1.

• Blocking rate: The blocking rate is also called the information loss rate (usually expressed by percentage), which is the ratio of information transmission (user requests) loss to total information transmission (user requests). Within an MEC system with bounded buffer, blocking rate should be paid attention to fulfill the satisfaction of users by preventing their requests from being discarded.

2. Dependability

Dependability represents the ability of a system to avoid severe service failures that are more frequent and more severe than is acceptable [1], which is also an important consideration in MEC [8]. Sometimes, dependability is a critical concern in IoT, especially for healthcare systems [9] and commercial IT service providers [10]. Dependability is a comprehensive metric that contains the following six attributes.

• Availability: Availability represents the accessibility of the service at the time that users submit the requests for service. It is usually measured by the steady-state probability of the service being available, ranging from 0% (system unavailable) to 100% (system always available).

• Reliability: Reliability reflects the ability of the system to offer service continuously without interruption. It can be expressed by the probability of the system keeping providing reliable services for a certain amount of time.

• Maintainability: Maintainability describes the ability of the system to undergo repairs and modifications. Sometimes, it can be measured by the mean time to repair (MTTR), ranging from 0 to infinity with the unit of seconds, milliseconds, or microseconds.

• Integrity: Integrity reflects the ability to make the service and data of a system absent of improper alterations or destructions, which is usually expressed by a probability (percentage) ranging from 0% to 100%.

• Safety: Safety reflects the ability of the system to remain without catastrophic consequences to users or the environment once a failure occurs. It can be evaluated by the steady-state probability of the system being in safe state.

• Confidentiality: Some literature also considered confidentiality as a dependability metric, representing the ability of the system denying unauthorized access to information. Confidentiality, integrity, and availability are usually jointly considered as three attributes of security. However, in some traditional dependability study, confidentiality is excluded from dependability [11].

3. Energy

Recently, energy efficiency has attracted significant attention in the design and maintenance of MEC systems [12]. For the energy dimension, there are two attributes as follows.

• Energy consumption: The energy consumption is the total energy consumed by the system during a time period. Energy consumption can have a very wide range of values, from μJ (e.g., micro-electronic devices) to kWh (e.g., household appliances) and even higher (e.g., industrial manufacturing). We let $E(t)$ denote the energy consumption from time 0 to t.

• Power consumption: The power consumption of a system is the rate at which the system consumes the energy at a given time point, which is usually measured in Watts (W). We let $P(\tau)$ express the power consumption at an instant time τ.

It can be concluded from the above definitions that the energy consumption is an integral of the power consumption, which can be mathematically expressed as

$$E(t) = \int_0^t P(\tau)\mathrm{d}\tau \tag{1}$$

4. Economic factors

Besides the QoS metrics mentioned above, economic factors are also key metrics in related literature when addressing QoS provision issues in MEC for IoT. It consists of the five main attributes of price, cost of ownership, cost of lease, revenue, and profit. All the economic attributes can be measured by the local or global currency.

• Price: Price is the amount of money that the user has to pay for purchasing a service or using some resources.

• Cost of ownership: Cost of ownership represents the total cost of a service provider owning and maintaining an edge computing infrastructure, which includes the initial hardware and software purchase costs as well as ongoing maintenance and upgrade costs.

• Rental cost: Rental cost represents the cost of a service provider for leasing edge computing resource or infrastructure, which usually depends on factors such as lease term, resource type, QoS level (also called service-level agreement, SLA), and competitive environment.

• Income: Income of a service provider is usually defined by the total revenue collected from its users for providing all the services or resources during a time period.

• Profit: Profit of a service provider is the difference between its income and the total cost for providing the services or resources including cost of ownership and rental cost.

5. Discussions

After deeply studying the QoS metrics, some researchers found that there are certain interrelationships among them, and the metrics may influence each other.

On the one hand, there are interrelationships between metrics within the same dimension. We present some examples as follows. Firstly, within the performance dimension, there is some correlation between response time and throughput, which are roughly inversely related. It has been well studied in computer network scenarios, and a well-known comprehensive evaluation function namely power formula can be used as a criterion when jointly optimizing these two attributes [13]. Secondly, for dependability dimension, availability can be sometimes regarded as "transient reliability", and it has been well-accepted that a reliable system must have high availability but an available system may or may not be very reliable. Also, it has been theoretically proved and formally illustrated the interrelationships between maintainability and safety, and that between availability and integrity in [14]. Thirdly, it has been shown that there is an integral relationship between power consumption and energy consumption as in (1). Fourthly, in the economic factors, it is quite straightforward that the income is related to the price, and the profit is determined by the income minus cost of ownership and rental cost.

On the other hand, although it has not been well explored, existing literature demonstrated that metrics of different dimensions can also be interrelated. For example, there is a complex relationship between performance and reliability. The failure of system components and their recovery or repair will make the system unable to respond and process services. Thus, user requests have to be discarded or to be left waiting in the queue, resulting in the increase of response time or blocking rate [15]. Another research work has shown that the workload can simultaneously affect the performance and reliability of the system, which also reveals the interrelationship between the metrics of both performance and reliability dimensions [16]. Another well-known example is the trade-off between performance and energy. High performance usually means high power consumption, which has to be carefully considered in most computing systems [17]. In addition, there is also a trade-off between reliability and energy consumption. To improve reliability, it is often necessary to add system component backups to prevent component failures from affecting serviceability. However, adding backups will bring additional energy costs [18].

III. QoS Evaluation

QoS evaluation can provide a benchmark for system design and optimization. The major task of QoS evaluation is to analyze the influence of system configurations, workload, strategies, and other factors on the QoS metrics. Well-known approaches of QoS evaluation can be classified into three categories, which are measurement-based approaches, simulation-based approaches, and analytical approaches. A comparison of the advantages and disadvantages of the different QoS evaluation methods is shown in Table 1.

1. Measurement-based approaches

The basic idea of measurement-based QoS evaluation approaches is to measure the QoS metrics or their closely related metrics directly from computer systems by designing and implementing certain measurement devices or computer programs and deploying them in real-life systems or simulators. Measurement-based approaches are the most straightforward and accurate among all the three types of approaches. Meanwhile, however, they are the most expensive, because the system to be measured has to be implemented and deployed before injecting the measurement into the system.

At present, several research works exist dedicated to the study of measurement-based QoS evaluation. Said [19] presented a methodology for guaranteeing QoS in IoT environments and measured the performance of QoS/QoE oriented scenarios in simulation environments built with the NS-3 software package. Truong *et al.* [20] developed a mobile edge cloud cornering assistance (MECCA) tool which is used to check performance of mobile edge cloud applications.

2. Simulation-based approaches

The simulation-based approach is to design a program to dynamically simulate all or part of the system's behaviors, and to statistically analyze the results by collecting data during the program operation. Although it is not necessary for simulation-based approaches to implement the real-life systems, the simulation processes are still time-consuming and costly. It is quite challenging to simplify the simulation models or speed up the simulation processes, without sacrificing acceptable accuracy of QoS evaluation.

Table 1 Summary of QoS evaluation approaches

Approaches	Advantages	Disadvantages
Measurement-based	• Authenticity: Able to provide actual system performance data.	• Hardware cost: May require expensive measurement equipment, thus increasing the difficulty of system deployment and maintenance.
	• Applicability: Measurement results can be used as the input parameters of other approaches.	• Software cost: May occupy system computing, storage, and network resources, involving additional performance overhead.
	• Real-time: Able to monitor system performance in real time.	• Difficult to scale up: Difficult to perform comprehensive measurements in large-scale systems.
Simulation-based	• Controllability: Able to simulate a variety of scenarios and variables; researchers can precisely control and adjust the parameters and conditions in the simulation experiments.	• Discrepancy: Simulation results may differ from the performance of the actual system.
	• Dependability: No impact on the actual system.	• Limitations: The accuracy of a simulation model is limited by the complexity and precision of the simulation model.
	• Low cost: Low cost compared to actual testing.	• Complexity: Conducting precise simulations (or emulations) in real-life scenarios may be extremely expensive.
Analysis-based	• Flexibility: Able to analyse different scenarios by simply tuning the parameters.	• Data requirement: A large amount of input data is needed to support the analysis.
	• Predictability: Able to predict system performance under different conditions without system implementation and deployment.	• Limited accuracy: The analysis results are affected by the accuracy of the input data and the model.
	• Low cost: Is the cheapest among the three approaches, no need for deployment or coding.	• Dependence on assumptions: Analysis-based methods are usually based on certain mathematical assumptions, which may not be able to cover all the actual situations.

There are many simulation toolkits and simulators related to edge computing environments, including Edgecloudsim [21], iFogSim [22], DeFog [23], etc. Sharma *et al.* [24] used iFogSim to verify the effectiveness of resource scheduling strategies. Benadji *et al.* [25] conducted simulations using Cooja/Contiki to analyze the performance of Constrained Application Protocol (CoAP) congestion control at runtime, and the simulation results show the limitations of CoAP. Ashouri *et al.* [26] identified metrics and quality characteristics which can be evaluated by simulation, and studied existing simulators to assess which of the identified qualities they support.

3. Analysis-based approaches

The main idea of analysis-based approaches is to establish a mathematical model of the system and calculate the QoS metrics through mathematical analysis. They usually need to introduce some mathematical assumptions in modeling to facilitate the mathematical analysis, and thus the accuracy in QoS evaluation might be sacrificed. However, such type of approaches is the most efficient in terms of computation, and also reveals the correlation between QoS attributes and system parameters. Hence, they are widely adopted especially in system design phase.

The commonly used mathematical models include Markov model, queueing model, stochastic network calculus (SNC), stochastic Petri net (SPN), etc.

a) Markov model

Markov model has been widely applied in performance evaluation, which uses Markov process to describe the dynamic behaviors of a system. In Markov model, the transitions among system states have to be memoryless (also called Markov property), which can be formally illustrated by the state transition probability.

Zhao *et al.* [27] presented an automated performance tracking, data management, and analysis framework for multi-tier cloud service systems, which supports fine-grained analysis of hybrid workloads via a discrete-time Markov modulated Poisson process. In [28], the authors considered reliability-aware task offloading and data compression for data-intensive applications in MEC systems, using Markov models to analyze state transitions during edge server processing tasks and obtain the quantitative relationship between reliability and system performance.

b) Queueing model

Queueing model is a mathematical approach to formulate the stochastic processes of task arrival and departure of a buffered system. With known task arrival and service process distributions and task scheduling policy, through rigorous mathematical deduction, some metrics can be derived, e.g., queue length, waiting time, sojourn time, and server utilization. Then, QoS metrics can be calculated from the analytical results.

The dynamic behavior of an atomic IoT service can be formulated by a queuing model, and thus a system consisted by a large number of IoT services which have complex interrelationships can be captured in a queuing network model. By solving the queuing network model, the QoS metrics can be obtained. Based on Jackson theorem and Burke theorem, some queueing network models can be independently solved in a parallel way, which is quite efficient in solving complex stochastic models. In

[29], Yousefpour *et al.* developed a queuing model for edge computing nodes and gave a latency analysis model for each link, after which the impact of the offloading policy on the system related hardware and software was quantified based on the model. Huang *et al.* [30] investigated age of information (AoI)-aware energy control and computational offloading challenges, where the dynamics of IoT devices and edge servers are captured by constructing a Markov queuing model.

c) Stochastic network calculus

SNC is another effective theoretical tool for evaluating QoS. It is a successor of queueing model which is general enough to handle almost all task arrival and service process distributions. The main idea of SNC is to use stochastic arrival curve and stochastic service curve for modeling and mathematical derivation through min-plus convolution theory. The results of SNC are usually expressed as probability bounds, which are mainly used to analyze the QoS metrics.

Mei *et al.* [31] were dedicated to analyze and optimize the latency of MEC networks under two orthogonal frequency division multiple access policies through SNC. Narimani *et al.* [32] used SNC for QoS-aware resource allocation and fault-tolerant operations in hybrid software defined networking (SDN). Wang *et al.* [33] modeled the data flow arrival and service process of narrow band Internet of things (NB-IoT) under uniform distribution and Beta distribution, respectively, and proposed to derive access delay bounds for NB-IoT using SNC.

d) Stochastic Petri net

SPN is an evolution of original Petri net (PN) by introducing time dimension and random variables. It has the similar powerful ability in describing complex dynamic systems as PN, and meanwhile is able to quantitatively analyze the QoS metrics. It is quite good at modeling the systems with service dependencies, task parallelism, and synchronizations, and thus has been widely adopted in QoS evaluation of IoT systems.

Zuberek [34] demonstrated that if the implementation delay time of a variation in an SPN model obeys an exponential distribution, then the SPN model is isomorphic to a Markov chain, and most system performance analysis based on SPN models relies on the property whose state space is isomorphic to a Markov chain. After converting the SPN model into a Markov chain, the performance metrics of the system can be analyzed by the theory of Markov process. Carvalho *et al.* [35] used the SPN model to describe and evaluate the MEC system, which can roughly calculate the average response time and resource utilization.

IV. QoS Optimization Problems

With QoS evaluation, the next step is to optimize the QoS metrics according to requirements from users or service providers. There are several research problems that could be addressed from QoS optimization perspective, including task offloading, resource allocation, edge caching, service migration, user allocation, collaborative computing, pricing incentives, etc.

1. Task offloading

Task offloading aims to transfer resource-intensive tasks to edge servers or cloud servers to address the shortcomings of end devices in terms of computing resource, storage, performance, and battery life, so as to improve QoS. The challenges of task offloading come from the following three aspects. Firstly, multiple objectives (e.g., latency, bandwidth, energy consumption, dependability, and security) may make the problem extremely difficult to formulate and solve, requiring multi-dimensional model analysis and multi-objective optimization approaches. Secondly, in fine-grained offloading schemes, a task can be decomposed into small subtasks and processed in parallel, which requires additional fine-grained QoS management and partial offloading techniques. Thirdly, the dynamic environment of IoT services requires adaptive or real-time approaches for QoS management and optimization. Huang *et al.* [36] presented a distributed offloading scheme for the overlapping areas of MEC for IoT, which effectively reduces the response time performance metrics and improves QoS. Li *et al.* [37] proposed a task offloading scheme with statistical QoS guarantees based on convex optimization theory and Gibbs sampling method.

2. Resource allocation

The resources on different edge servers are usually highly heterogeneous, and different computing tasks have different demands on resources such as computation resource and communication resource. Hence, how to allocate the heterogeneous resources to different tasks is important to ensure the QoS in edge computing for IoT. Resource allocation faces challenges due to user mobility, dynamic system states, multi-user multi-task competition, security, etc. In multi-access MEC systems, the problem may become more complex, and it is necessary to consider heterogeneous resources among multiple networks in QoS management and optimization. Due to the complexity and dynamic characteristics of resource allocation problem, dynamic optimization models and heuristic approaches are highly demanded. Also, some literature proposed joint task offloading and resource allocation scheme for QoS optimization. Shao *et al.* [38] proposed a two-stage business collaboration computing mechanism including resources allocation and task allocation to optimize the business delay and energy consumption of unmanned aerial vehicle (UAV). An *et al.* [39] studied the joint optimization challenge of task offloading and resource allocation under slow and fast fading channels, which is solved using convex optimization theory.

3. Edge caching

Edge caching technology can well reduce the workload of the network through caching the content that end users need at the edge nodes. The challenges are to answer the questions including what to cache, where to cache, and when to cache, which relies on the judgement or prediction of user requirements and content popularity. Cache prediction and content optimization (usually their combination) schemes should be designed for edge caching. Liu *et al.* [40] investigated the challenge of data caching in MEC systems from the perspective of a service provider, and proposed an approximate approach to find a near-optimal solution, with the aim of improving data caching revenue under access latency constraints. Zhang *et al.* [41] studied the joint service caching, computation offloading and resource allocation problems in a multi-user multi-tasking MEC environment, and proposed an efficient approximation algorithm based on semi-infinite relaxation and alternating optimization to minimize the total computational and latency costs for all users.

4. Service migration

Since the limited coverage of edge servers, the mobility of IoT users can increase access latency and even cause disruptions to ongoing edge services. Service migration is an effective way to ensure the continuity of services. The primary purpose of service migration is to determine whether a service should be migrated from the original edge server hosting the service to another edge server after a dynamic change in user location and service requirements [42]. Finding the optimal service migration decision is challenging due to the mobility of users, the dynamic nature of service demand, and the limited and heterogeneous nature of edge server resources. The main solution is to develop a dynamic and seamless switching mechanism to migrate services between different edge nodes without affecting the user experience, while at the same time using real-time state synchronisation techniques to ensure data consistency during service migration. If scaling to a complex multi-access MEC system, it is necessary to consider seamless service migration between different network access technologies to ensure QoS. Liu *et al.* [43] investigated the joint optimization problem of service migration and resource allocation in MEC environments to minimize the access delay of IoT users.

5. User allocation

User allocation focuses on solving the load balancing problem of edge nodes. Edge servers can serve multiple users, but a user can usually connect to only one edge node. Since the resources of edge servers are heterogeneous and limited, an irrational user allocation strategy may result in some edge nodes serving too many users while other nodes are idle. Overloading leads to degradation of service quality, while idling of nodes will reduce system resource utilization. Therefore, system resources can be utilized more efficiently and service quality can be ensured by reasonably allocating users to edge nodes that are idle or have lower loads. However, solving the user allocation problem becomes challenging due to factors such as mobility of users, heterogeneity of edge server resources, and dynamics of system state. To cope with the changing environment, a dynamic load balancing mechanism needs to be designed. Peng *et al.* [44] designed a mobility-aware and migration-enabled online decision scheme for solving the real-time allocation challenge of edge users. Liu *et al.* [45] focused on the user allocation problem in overlapping areas, aiming at balancing the workload and minimizing the communication time.

6. Collaborative computing

Collaborative computing refers to the co-operation between multiple edge devices or edge servers to share computational resources, data, and tasks for more efficient and intelligent processing of computational tasks. It can be categorized into: resource collaboration, data collaboration, intelligent collaboration, application management collaboration, business orchestration collaboration, and service collaboration. However, edge devices usually have different hardware and software configurations, and this heterogeneity makes QoS optimization in collaborative computing more complex. Also, the constant changes in the edge environment require collaborative computing systems to be dynamically adaptable and scalable. To address these challenges, collaboration among cloud, edge, and devices as well as the task dependencies [46] should be carefully considered in the mechanism design of collaborative computing.

7. Resource pricing and incentives

In edge computing, edge servers may be deployed by different infrastructure providers with certain computing and storage capabilities. In addition, users' resource requirements are usually price-sensitive, and when the prices of edge server resources are set at a low price, more users will be attracted to buy edge server resources; and when edge server resource providers raise the price of edge server resources, the resource requirements of users will be tightened. How to design an appropriate pricing strategy that encourages users to use edge services while maintaining the profitability of the provider is a challenge [47]. In addition, the operating costs of edge server resource providers and the resource demands of users fluctuate over time. Hence, it is important to design an elastic pricing strategy that dynamically adjusts prices based on demand, time, and resource supply and demand. For the more complex problem of resource pricing in multi-access MEC system, the cost and performance of multiple network access technologies need to be considered in order to implement a reasonable resource pricing strategy. Li *et al.* [48] defined a cloud resource pricing and requirement allocation model with the optimization objective of service revenue maximization, and designed a price-incentivized cloud resource auction mechanism.

8. Energy control

Energy control is to manage and adjust the energy usage of MEC system to minimize energy consumption while ensuring user-level QoS. Firstly, the energy control problem can be solved by speed scaling technique whose basic idea is to adjust the energy consumption of the devices or servers according to dynamic workload and environment. Secondly, energy harvesting technologies (solar, thermal, vibration, etc.) and energy storage technologies (super-capacitors or lithium batteries) become popular recently for the energy control in battery-equipped devices. The challenges of energy control is how to dynamically adjust the energy source and power consumption while fulfilling end-to-end QoS requirements. Stochastic optimization techniques and intelligent dynamic optimization approaches are commonly applied. Lu *et al.* [49] designed a low-delay packet delivery scheme that adapts to variation in the harvested energy. Zhao *et al.* [50] investigated the dynamic offloading and resource scheduling problem between local devices, base stations, and back-end clouds. The goal is to minimize energy and computational resource consumption in MEC systems with energy harvesting devices.

V. QoS Optimization Models and Solutions

In order to solve a QoS optimization problem, a mathematical optimization model that formulates the problem has to be constructed. And then, corresponding solutions have to be designed and implemented. There have been a number of optimization models as well as numerous approaches for solving them. Generally, in this paper, we classify them into two categories, which are static optimization and dynamic optimization. A comparison of the advantages and disadvantages of static and dynamic optimization approaches is shown in Table 2.

1. Static optimization

A generic static optimization model consists of three components: optimization objectives, decision variables, and constraints. The generic form of an optimization model can be expressed as follows.

$$\max_x/\min \{f_1(x), f_2(x), \dots, f_k(x)\} \tag{2a}$$

$$\text{s.t.} \quad f_i(x) \leq 0, \ i = 1, 2, \dots, m \tag{2b}$$

$$h_i(x) = 0, \ i = 1, 2, \dots, p \tag{2c}$$

The optimization model can be roughly classified into single-objective optimization model (when $k = 1$) and multi-objective optimization model (when $k \geq 2$) according to the number of optimization objectives. Also, a multi-objective optimization problem can be transformed into some single-objective optimization problems using

Table 2 Summary of QoS optimization approaches

Approaches	Advantages	Disadvantages
Convex optimization (static)	• Global optimality: The solution obtained by convex optimization is globally optimal.	• Convex limitation: Can only be used for convex functions, but many practical problems are not convex.
	• Mathematical foundation: Ensures the stability and reliability of the algorithm.	• Constraint restriction: The strict requirement of problem constraints limits its applicability in solving real-world problems.
	• Efficiency: In most cases, convex optimization problems can be solved in polynomial time.	
Game theory (static)	• Modelling of multiple parties: Can describe the decision-making process involving multiple parties and helps to analyze the best strategy for each party.	• Complexity: Game models may be very complex and difficult to solve.
		• Diversity of Nash equilibrium: A game may have multiple Nash equilibrium solutions.
Ordinal optimization (static)	• Efficiency: Can usually saves the computing budget by at least one order of magnitude.	• Optimality: Can only find good enough solutions.
	• Mathematical foundation: Can be shown in a mathematically rigorous way.	• Rough: Uses rough models in performance estimation.
Markov decision process (dynamic)	• Considering uncertainty: MDP allows uncertainty to be taken into account in decision making, making it more robust.	• Dimensionality catastrophe: When the state space is large, the algorithmic solution becomes very difficult, i.e., the dimensionality explosion.
	• Applicability: Can be used to model and solve a variety of sequential decision-making problems.	• Need for models: Usually needs to model the environment, but it is difficult to get an accurate model in real-world problems.
Deep reinforcement learning (dynamic)	• Adaptation to complex environments: Can handle problems with high uncertainty and complex decision spaces.	• Resource consumption: A large amount of computational resources are required to train complex deep networks.
	• Efficiency: Able to solve MDP problems with large-scale search space.	• Unstable training and poor interpretation: Difficult to converge to a suitable policy and difficult to explain the decision process inside the model.
Lyapunov optimization (dynamic)	• Stability: Can be proved that the queue stability is guaranteed.	• Manual construction: The Lyapunov function needs to be constructed manually, and the construction process may be relatively complicated.
	• Efficiency: Can solve dynamic optimization problem with static optimization algorithms with low overhead.	• Limitations: Can only be used for specific types of queueing systems.

some techniques such as simple additive weighting (SAW), ε-constraint, and lexicographical order.

Taking the task offloading and resource allocation (TORA) problem as an example, in an end-edge-cloud collaborative computing scenario for IoT, the optimization objective can be set to minimize the offloading latency $\sum_{k\in K} T_k(x)$ for all tasks, and the constraints are the resource limits of edge servers and cloud servers. A generic formulation of the TORA problem can be expressed as follows.

$$\max_{x} \sum_{k\in K} T_k(x) \tag{3a}$$

$$\text{s.t.} \quad \sum_{k\in K} f_k \le F \tag{3b}$$

where x is the offloading decision variable, K is the set of tasks, f_k is the amount of resources allocated to the task $k \in K$, and F is the available resources at the edge or cloud server. For the binary offloading problem, it is necessary to determine whether the task is offloaded to edge server or cloud server, and the offloading decision x is usually set to 0 or 1. The constraint usually contains $\sum_{k\in K} x_k = 1$. For the partial offloading problem, the decision variable can be set as the offloading ratio between 0 and 1.

1) Convex optimization

For a static optimization model, (e.g., (3)), if the objective function is convex and the constraint set is convex set, it can be classified as a convex optimization problem. The convexity of the multivariate objective function needs to be determined by whether the Hessian matrix is a semi-positive definite matrix, and if this convex optimization problem also satisfies the Slater condition, the original problem has the same optimal solution as the dual problem (i.e., strong duality), and solving the original problem is equivalent to solving its Lagrangian dual problem. For example, the Lagrangian function of (3) is

$$L(x,\lambda) = \sum_{k\in K} T_k(x) + \alpha\left(F - \sum_{k\in K} f_k\right) \tag{4}$$

where λ is the non-negative Lagrangian multipliers for the corresponding constraints.

The dual functions is $g(\alpha) = \max L(x,\alpha)$, and the dual problem is $\min_\alpha g(\alpha)$ where $\alpha \ge 0$. The dual problem can be solved iteratively using the gradient method with the following iterative formula

$$\alpha_k^{(z+1)} = \left[\alpha_k^{(z)} - \phi^{(z)}\left(F - \sum_{k\in K} f_k\right)\right]^+ \tag{5}$$

where ϕ is the step length.

The local optimal solution of a convex optimization problem must be its global optimal solution, while for general non-convex optimization problems, it is difficult to obtain the global optimal solution. The solution methods for nonconvex optimization problems usually use block coordinate descent, successive convex approximation, projective gradient descent, etc. For nonconvex optimization problems with complex constraints, the Lagrangian relaxation method can be used to relax the complex constraints to the objective function, realize the decoupling of multiple decision variables, and finally decompose the original problem into multiple subproblems for solution. Although the Lagrangian relaxation method can reduce the difficulty of solving, it will also reduce the quality of the solution.

Currently, convex optimization theory has been widely used in QoS optimization. Sundar *et al.* [51] used 0-1 variables in the relaxed integer programming model to achieve a transformation from non-convex to convex optimization and thus design heuristics. Liang *et al.* [52] optimized migration/switching strategies between base stations by jointly managing computational and radio resources and devised an relaxation and rounding-based solution method to maximize the total offload rate, quantify the MEC throughput and minimize the migration cost.

2) Game theory

Game theory is a powerful theoretical framework that can be used to analyze the interaction behavior between multiple participants, and can help for designing decentralized mechanisms. Each participant in a game is selfish and aims to maximize the value of their respective utilities. We can model the optimization problem as a classical game model and then find the optimal strategy for each participant, i.e., each participant chooses the strategy that is the most beneficial to themselves.

Game theory can be classified into many types. From the perspective of how well the participants know the other participants, game theory can be categorized into complete information games and incomplete information games. From the perspective of whether have binding agreements, game theory can be categorized into cooperative and non-cooperative games. From the perspective of the time-series nature of behavior, game theory can be categorized into static and dynamic games.

The core of game theory is the Nash equilibrium [53]. In the game $G = \{S_1, S_2, \dots, S_n : u_1, u_2, \dots, u_n\}$, if a certain strategy combination $(s_1^*, s_2^*, \dots, s_n^*)$ consists of each strategy of each party to the game, the strategy s_i^* of any game party i is the best response to the rest of the game strategy of the combination $(s_1^*, \dots, s_{i-1}^*, \dots, s_{i+1}^*, \dots, s_n^*)$, that is, $u_i(s_1^*, \dots, s_{i-1}^*, s_i^*, s_{i+1}^*, \dots, s_n^*) \ge u_i(s_1^*, \dots, s_{i-1}^*, s_{ij}^*, s_{i+1}^*, \dots, s_n^*)$ for any $s_{ij} \in S_i$ is hold, then $(s_1^*, s_2^*, \dots, s_n^*)$ is said to be a "Nash equilibrium" of G.

Huang *et al.* [54] proposed an incentive mechanism based on a two-stage Stackelberg game to inspire users to contribute computing resources for federated learning (FL). Chen *et al.* [55] modelled the channel selection problem for multiple UAVs communicating with a base station in an MEC system as a non-cooperative game and proved the existence of Nash equilibrium. After that, the authors designed a multi-UAV communication channel selection approach to find the equilibrium policy status for all UAVs.

3) Ordinal optimization

Ordinal optimization (OO) is another efficient approach for solving large-scale search-based optimization problems. The main idea of OO is "soft optimization of hard problems", that is, the original optimization goal of "finding the best solution" is softened into "finding a good enough set with high probability". Mathematically, this objective can be expressed as

$$\Pr\left[(G\cap S)\geq k\right]\geq\alpha \tag{6}$$

where G represents the "good enough set", S is the set of solutions obtained by OO, k is called the alignment level, and $\Pr(G\cap S)\geq k$ is the alignment probability.

In OO-based optimization scheme, a crude model is constructed for QoS evaluation. The crude model is used for ordinal comparison among the solutions and thus it can be computationally fast but rough in performance estimation, which can easily screen out good enough designs and thus save the computing budget. With OO theory, the size of S can be precisely calculated according to the noise level and problem type, and then precise model for QoS evaluation is applied on the good enough set for finding the near-optimal solution. It has been shown that OO usually saves at least an order of magnitude in computational budget and is easily combined with other optimization methods.

Tan *et al.* [56] investigated the joint optimization problem of task offloading and resource allocation in large-scale MEC environment, and proposed to use OO theory to find a near-optimal strategy in a reduced search space, solving the search space explosion problem in the QoS optimization process. Huang *et al.* [57] presented a novel QoS-aware dynamic service selection scheme, where goal softening was used for the original optimization problem and the service selection algorithm was designed by OO techniques.

2. Dynamic optimization

In IoT environment, the internal state of the MEC system and the external environment are time-varying. In order to capture the high dynamics of MEC in IoT, dynamic optimization is applied, which introduces the time dimension into original optimization problem and aims at achieving the long-term QoS revenue.

1) Markov decision process

Markov decision process (MDP) is a popular tool to formulate a dynamic optimization problem. In general, an MDP can be defined by a tuple (S,A,P,R,T) whose the elements represent states, actions, transfer probability functions, finite set of all possible rewards, and time series, respectively. During each decision time slot t, the current state s_t of the environment is observed and an action a_t is selected according to a certain policy, where the policy can be considered as a mapping from an arbitrary state s $(s\in S)$ to an action a $(a\in A)$. After executing the action a $(a\in A)$, it is transferred to the next state according to the transfer probability $P\left(s_{t+1}\mid s_t,a_t\right)$ and the corresponding reward $r_t=R(s_t,a_t)$ is obtained through the immediate reward function, in which case the current policy and the transfer probability function determine the long-term cumulative reward. The objective of MDP is to obtain the optimal policy π^* that maximizes the long-term cumulative reward. To this end, the optimization objective of MDP can be expressed as

$$V(s_t)=\min_{a_t\in A}\left\{r_t+\gamma\sum_{s_{t+1}\in S}P\left[s_{t+1}\mid s_t,\ a_t\right]V(s_{t+1})\right\} \tag{7}$$

where γ is the discount factor. The optimal solution of the MDP model can usually obtained by the value iteration, the policy iteration, or Q-learning.

MDP has been widely used in QoS optimization problems of MEC for IoT. Wang *et al.* [58] considered two-dimensional migration in the MDP setting of the service migration problem, with the objective of reducing service delay. Zhang *et al.* [59] investigated the vehicle edge computing task allocation problem, which mainly addresses the problem of when to assign tasks and to whom, and describes it as a finite-time domain MDP to minimize the latency in communication, computation, switching, and migration.

2) Deep reinforcement learning

MDP is a model-based approach that requires to model dynamic features of the environment including state transfer probabilities and reward functions. Therefore, it is challenging to solve MDP problems with large-scale action spaces or continuous state spaces using traditional iterative MDP algorithms. For this reason, deep reinforcement learning (DRL) has been introduced, which involves Q-learning and deep neural network (DNN) techniques. Compared to MDP, DRL adopts a more flexible and adaptive learning approach, where the DRL agent collects empirical data through continuous interaction with the environment and uses this data to update and optimize the DNN to find the optimal policy. Through this experience-driven learning approach, DRL is able to adapt to unknown, complex, and unstable environments through continuous trial and feedback to gradually improve its policy performance.

The DRL algorithms can be classified into value-based DRL and policy gradient-based DRL depending on the optimization policy. The value-based DRL is the approximation of the reward value function using DNN. Similarly, the method of approximating a policy with a DNN and finding the optimal policy using the policy gradient method is called policy gradient-based DRL. The value-based DRL algorithms mainly include deep Q-network (DQN) and various improved algorithms based on DQN. In DQN, DNN estimates the Q-value function by receiving the state as input and outputting the Q-value of each action, and then selects the optimal action by a greedy strategy. DQN mainly uses two techniques, empirical replay and target network, to solve the problem of instability or even non-convergence when approximating action-valued functions with DNN.

In the policy gradient, policies are classified into stochastic policy $\pi_\theta(a|s) = P[a|s;\theta]$ and deterministic policy $a = \mu_\theta(s)$. For stochastic policies, actions a satisfy a certain probability distribution with parameter θ when the current state is s, so the same state will correspond to different actions, while deterministic policies correspond to unique actions for each state for deterministic policies $\mu_\theta(s)$. Corresponding to policies, policy gradients are classified into stochasticity policy gradient (SPG) and deterministic policy gradient (DPG). The common DRL algorithms based on policy gradient include deep deterministic policy gradient (DDPG), trust region policy optimization (TRPO), and asynchronous advantage actor-critic (A3C).

At present, DRL has been widely applied to dynamic optimization problems in high-dimensional state and action spaces. Huang *et al.* [60] proposed a deep reinforcement learning based dueling double deep Q-network (dueling DDQN) algorithm for solving the problem of navigation for UAV pair-supported relaying in unknown IoT systems. Ke *et al.* [61] investigated how to guarantee computational offloading efficiency of vehicular networks and presented an adaptive computational offloading approach based on the DRL to obtain the optimal policy.

3) Lyapunov optimization

Lyapunov optimization has shown to be another way for solving dynamic optimization problem in MEC for IoT. It is good at solving constrained dynamic optimization problems for queueing systems, especially in ensuring the queue stability. By introducing the Lyapunov drift-plus-penalty function, the original dynamic optimization problem can be solved by transforming it into a static optimization problem for each time slot. The main steps of Lyapunov optimization are as follows.

Firstly, we need to create a virtual queue for each time-averaged constraint and convert the constraints to the stability of the virtual queue. Secondly, we use the queue vector $\boldsymbol{\Theta}(t) = \{Q_i(t)\}$ to represent all the queue states in the system, then the Liapunov drift plus penalty function can be defined as

$$L[\boldsymbol{\Theta}(t)] = \frac{1}{2}\sum Q_i(t)^2 \tag{8}$$

The Lyapunov function represents the squeezed state of the queue in the system. To jointly consider queue stability and optimization objectives, the Liapunov drift plus penalty function is defined as

$$\begin{aligned}\Delta L[\boldsymbol{\Theta}(t)] + VE\left[\sum f_i(t) \mid \boldsymbol{\Theta}(t)\right] \\ = E\left[L\boldsymbol{\Theta}(t+1) - (L\boldsymbol{\Theta}(t)) + V\sum f_i(t) \mid \boldsymbol{\Theta}(t)\right]\end{aligned} \tag{9}$$

where the parameter V is used to weigh the drift function against the penalty function. If V takes a larger value, it means more focus on queue stability, otherwise more focus on optimization goals. Finally, the original dynamic optimization problem is transformed to minimize the upper bound of (9), which is a static optimization problem for each time slot t and can be solved using static optimization methods. The Lyapunov optimization method proves that the difference between the solution of the problem and the optimal solution is bounded if the original problem has a feasible solution.

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T-1} E\{f(t)\} \le f^* + \frac{B}{V} \tag{10}$$

where B is a constant and f^* represents the optimal solution of the original problem. Moreover, the average queue length needs to satisfy the following inequality

$$\sum_{k=1}^{K} E\{|Q_k(t)|\} \le \frac{B + V(f^* - f)}{\varepsilon} \tag{11}$$

where ε is a constant.

The Lyapunov optimization algorithm can obtain an effective trade-off between queue stability and performance, and is an online algorithm that requires only the current state of the system when making decisions. However, the Lyapunov optimization algorithm is only applicable to solve Markovian decision problems where the objective function and constraint function are time-averaged functions.

Liapunov optimization has been widely used in constrained dynamic optimization problems. Zhou *et al.* [62] applied Lyapunov optimization technique to address the challenge of task offloading and resource allocation in UAV-assisted multi-cloud systems with the objective of minimizing UAV energy consumption while ensuring queue stability. Huang *et al.* [63] applied the Liapunov optimization framework to solve the joint admission control and computational resource allocation challenges in MEC-enabled small cell networks (SCNs).

VI. Challenges in Edge Intelligence Era

With the rapid development of AI techniques, DNNs have been applied in many applications. In IoT, DNNs have also to be deployed in the MEC architecture for supporting intelligent applications. Edge intelligence, as a promising technique, has attracted increasingly attention from both academic and industry. How to evaluate and optimize the QoS in edge intelligence is one of the most challenging tasks to be addressed. Follows, we list some emerging edge intelligence techniques, and discuss the QoS evaluation and optimization issue in them.

1. Federated learning

With the explosive growth of data volume of various IoT devices, uploading all data to cloud server for centralized processing will be challenging because of the limitation of network bandwidth, storage resources, and data privacy issues. For this reason, FL has emerged as a novel distributed learning framework for edge intelligence. Instead of uploading original data to the cloud for training, edge devices just use their local data to train the local deep learning (DL) models and upload the model parameters to the FL server at the cloud. Then, the FL server will aggregate all the local models and obtain the global model using some weighted averaging algorithms, such as federated average (FedAvg) [64]. Compared to the traditional cloud computing paradigm, FL enhances data privacy and reduces the workload at the core network. However, it still faces several challenges in QoS provisioning, as follows.

- End-to-end delay: In FL, the end-to-end delay consists of the following four parts. 1) The training time of the local models on the IoT devices or edge devices should be taken into account in QoS evaluation and optimization. 2) Model parameters (or its related updating information) need to be transmitted through communication channels (usually through wireless communications) between the servers and devices, and thus the communication delay has to be fully considered. 3) At the service site, the data needs to be processed and the aggregated model is calculated, resulting in the aggregation time. 4) The models should be trained iteratively until their convergence, and hence the convergence time should be carefully considered in the calculation and optimization of the end-to-end delay.
- Resource management: The implementation of FL relies mainly on parallel training on IoT devices and the aggregation of parameters on global server. However, the computational resources are usually limited in IoT devices. Improper or unbalanced resource allocation may lead to significant delays in synchronizing parameter aggregation on the server resulting in a severe degradation of QoS. Also, the resource management scheme in the cloud site as well as the allocation mechanism of communication resources have impact on the QoS of FL. Therefore, how to design an optimal resource management scheme is critical for QoS optimization in FL.
- Pricing and incentive mechanism: In FL, mobile devices need to dedicate their computational and storage resources to data training, and thus it is critical to recruit high-quality participants to improve model accuracy and reduce training time. However, participants are usually individually rational and selfish, and may be reluctant to participate in FL because of their limited resources and supererogatory cost. In addition, an adversary can learn the participant's data through the generative adversarial network, so the participants face the risk of privacy disclosure, adding further concerns for their involvement. Pricing strategies can be introduced to incentivize participants by offering monetary compensation, discounts on computational resources, or other benefits based on their level of contribution in FL. Additionally, leveraging game theory and economics-based approaches in designing incentive mechanisms has proven effective in the context of FL. Therefore, designing effective incentive mechanisms in FL is a promising research direction to solve the above problems.

2. Edge inference

With the increasing computational capabilities of edge devices, there is an inevitable trend to embed machine learning related computational tasks on the edge for execution. How to efficiently deploy and run DL models on edge nodes or IoT devices where both computing capabilities and energy consumption are constrained is challenging in edge intelligence scenarios [65]. The existing neural network partitioning schemes including both horizontal and vertical partitioning make it possible to deploy neural networks distributively on different edge/IoT nodes. However, the study on its optimization from the QoS provisioning aspect still remains largely unexplored. As follows, we list some of the latest techniques of edge inference and provide some insights from the QoS viewpoint.

- Model segmentation: DL models usually consist of multi-layer neuron networks. Model segmentation technique can partition a DL model into several parts and offload them to multiple edge servers or neighboring mobile devices. Benefiting from the collaboration between edge nodes and IoT devices, the QoS of edge inference can be enhanced. However, the selection of model partitioning points has to be carefully studied considering the available computational and network bandwidth resources of different edge and IoT nodes [66]. In addition, the connection between the layers after model segmentation may cause additional communication overhead between nodes. The general process of determining the segmentation points can be divided into three steps [67]: 1) measuring and modeling the resource cost of different DNN layers and the size of intermediate data between layers; 2) predicting the total cost based on the specific layer configuration and network bandwidth; 3) selecting the best one from the candidate segmentation points based on latency, energy requirements, etc.
- Early exit: To accelerate deep model inference, the model early exit technique saves runtime through processing the output results of the more advanced network layers to terminate model inference early and obtain the output results. Teerapittayanon *et al.* [68] presented a deep network architecture BranchyNeta which adds side-branching classifiers. This architecture allows some of the test sample's prediction results to exit the

network early when the samples satisfy high-confidence inferences. Although the early model exit technique can effectively reduce the resource consumption, it also reduces the model accuracy. Therefore, the trade-off between QoS and model accuracy has to be carefully balanced when designing model exit scheme or choosing the optimal exit points.

• Model compression: Model complexity can be reduced by compressing DL models in a way that better enables low-latency and low-energy model inference on resource-constrained edge devices. However, model compression will reduce the model accuracy, so how to trade-off QoS and accuracy is still a challenge in this area. Liu *et al.* [69] presented a usage-driven selection framework AdaDeep, which can adaptively select various compression techniques based on the currently available resources to form a compression model for optimal model inference.

• Model selection: There are usually multiple candidate models that can implement the same DL function. However, for the same input, different models have different resource consumption and model accuracy. Therefore, how to select the optimal model dynamically and adaptively for the input, so as to collaboratively optimize the resource consumption and model accuracy, is an interesting research topic. Taylor *et al.* [70] proposed a machine learning-based adaptive model selection algorithm that automatically selects DNN models according to input and accuracy requirements.

• Edge caching: In edge intelligence, edge caching refers to caching data, models, or computation outputs of DNN at the edge of the network so that such cached contents can be returned directly upon user requests. The purpose of edge caching is to reduce the workload of the centralized services in the cloud, decrease the average response time of user requests, and enhance the service availability and reliability. However, comparing to the cloud, the computational capability and storage capacity of the edge nodes are limited, and hence it is impossible to cache all the contents that users request. Therefore, how to adaptively adjust the cache contents according to the dynamic environments and varying user preferences is one of the most important challenges to be addressed. Moreover, the DNN model consistency and computational security at the edge are also critical factors affecting the QoS and user experience, which should be taken into consideration in QoS evaluation and optimization of MEC with edge caching techniques.

3. Digital twin

A digital twin is a virtual representation of an intended or actual real-world physical product, system, or process. It leverages IoT devices for real-time data collection and edge computing for simulation and decision making. As an emerging technique, several aspects in digital twins still remain unexplored.

• Data synchronisation and accuracy: Digital twins rely on real-time data synchronisation to ensure that the digital twin is aligned with the state of the actual physical system. In MEC environment, data synchronisation can become more complex due to issues such as network latency, bandwidth limitations, and data loss. Therefore, ensuring data accuracy and real-time availability of digital twins is a challenge. Zhou *et al.* [71] addressed the key challenges of digital twin construction and digital twin-assisted resource scheduling such as low accuracy, significant iteration delays, and security threats.

• Computing capability and resource constraints: Edge devices typically have limited computing and storage resources, while creating and maintaining complex digital twin models may require significant computing resources, conflicting with the resource constraints of edge devices. Therefore, it is a challenge to efficiently build and update digital twin models with limited resources. Li *et al.* [72] researched the digital twin driven vehicular edge computing networks for adaptively computing resource management.

• Real-time requirements: Certain edge applications require real-time response, such as industrial automation and autonomous driving systems. In these applications, the digital twin's model needs to be updated in real time to reflect changes in the state of the actual system. Achieving real-time requirements can be challenged by network latency and computational limitations [71].

• Security and privacy protection: Digital twins contain detailed information about the physical system, so security and privacy protection is an important consideration. In edge intelligence environments, data transmission may pass through insecure networks, so appropriate encryption and security measures are needed to protect the confidentiality and integrity of digital twin data. Zhou *et al.* [73] analyzed potential security issues during the migration of digital twin models in the Internet of vehicles (IoV) environment, and proposed corresponding defense methods against these network attacks.

• Fault tolerance: Edge environments are often more susceptible to external interference and faults. Therefore, digital twins need to have a certain degree of fault tolerance in the face of network interruptions, device failures and other abnormal situations in edge environment to maintain system stability and reliability [71].

VII. Conclusion

With the growing popularity of IoT, QoS has become one of the most important concerns in many IoT applications. QoS evaluation and optimization of MEC for IoT have attracted increasing attention from both academia and industry. This paper conducts a comprehensive survey on multi-dimensional QoS evaluation and optimization of MEC for IoT. Multi-dimensional QoS metrics are summarized and their formal definitions are presented. QoS evaluation approaches are introduced, which are classified into three types including measurement-based, simulation-based, and analysis-based approaches. Then, QoS optimization problems in real MEC systems are summarized, and optimization models as well as their corresponding solutions are provided. Finally, the emerging edge intelligence is introduced, and the corresponding challenges of QoS provisioning are discussed. This paper is expected to provide a comprehensive review and some insights of QoS research in MEC and IoT areas.

Besides edge intelligence, there are still many future research directions to be explored in this field. Firstly, the IoT environment is highly dynamic, which poses serious challenges to the ability and efficiency of service offering and its QoS provisioning. It is always valuable to develop more effective and adaptive modeling and optimization approaches for IoT scenarios. Secondly, as the scale of IoT systems grows exponentially fast, there are several tough challenges to be addressed, such as state-space explosion in model-based QoS evaluation and search-space explosion in QoS optimization. The efficiency of the QoS evaluation and optimization approaches should be further improved to attack such challenges. Thirdly, emerging IoT scenarios and applications may require new models and approaches, which may promote the theoretical and technical research in QoS provisioning and its related fields.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (Grant Nos. 2022YFD2001000 and 2022YFD2001002).

References

[1] A. Avizienis, J. C. Laprie, B. Randell, *et al.*, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.

[2] C. Feng, P. C. Han, X. Zhang, *et al.*, "Computation offloading in mobile edge computing networks: A survey," *Journal of Network and Computer Applications*, vol. 202, article no. 103366, 2022.

[3] Y. F. Zhan, J. Zhang, Z. C. Hong, *et al.*, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 2, pp. 1035–1044, 2022.

[4] H. M. Qiu, K. Zhu, N. C. Luong, *et al.*, "Applications of auction and mechanism design in edge computing: A survey," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 1034–1058, 2022.

[5] P. J. Cong, J. L. Zhou, L. Y. Li, *et al.*, "A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud," *ACM Computing Surveys*, vol. 53, no. 2, article no. 38, 2021.

[6] P. Ranaweera, A. Jurcut, and M. Liyanage, "MEC-enabled 5G use cases: A survey on security vulnerabilities and countermeasures," *ACM Computing Surveys*, vol. 54, no. 9, article no. 186, 2022.

[7] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, "Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research," *Internet of Things*, vol. 12, article no. 100273, 2020.

[8] S. Bagchi, M. B. Siddiqui, P. Wood, *et al.*, "Dependability in edge computing," *Communications of the ACM*, vol. 63, no. 1, pp. 58–66, 2020.

[9] K. Wang, Y. Shao, L. Xie, *et al.*, "Adaptive and fault-tolerant data processing in healthcare IoT based on fog computing," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 263–273, 2020.

[10] A. Aral and I. Brandić, "Learning spatiotemporal failure dependencies for resilient edge computing services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 7, pp. 1578–1590, 2021.

[11] A. Avizienis, J. C. Laprie, B. Randell, *et al.*, "Fundamental concepts of dependability," *Newcastle University Report*, no. CSTR-739, pp. 1–21, 2001.

[12] E. Ahvar, A. C. Orgerie, and A. Lebre, "Estimating energy consumption of cloud, fog, and edge computing infrastructures," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 2, pp. 277–288, 2022.

[13] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*, 4th ed., Morgan Kaufmann, Boston, USA, 2007.

[14] J. W. Huang, C. Lin, X. Z. Kong, *et al.*, "Modeling and analysis of dependability attributes for services computing systems," *IEEE Transactions on Services Computing*, vol. 7, no. 4, pp. 599–613, 2014.

[15] M. S. Elbamby, C. Perfecto, C. F. Liu, *et al.*, "Wireless edge computing with latency and reliability guarantees," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1717–1737, 2019.

[16] N. Kherraf, S. Sharafeddine, C. M. Assi, *et al.*, "Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1435–1449, 2019.

[17] B. Soret, L. D. Nguyen, J. Seeger, *et al.*, "Learning, computing, and trustworthiness in intelligent IoT environments: Performance-energy tradeoffs," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 629–644, 2022.

[18] K. Ergun, R. Ayoub, P. Mercati, *et al.*, "Energy and QoS-aware dynamic reliability management of IoT edge computing systems," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, Tokyo, Japan, pp. 561–567, 2021.

[19] O. Said, "Design and performance evaluation of QoE/QoS-oriented scheme for reliable data transmission in internet of things environments," *Computer Communications*, vol. 189 pp. 158–174, 2022.

[20] H. L. Truong and M. Karan, "Analytics of performance and data quality for mobile edge cloud applications, " in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, San Francisco, CA, USA, pp. 660–667, 2018.

[21] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of edge computing systems," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 11, article no. e3493, 2018.

[22] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, *et al.*, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017.

[23] J. McChesney, N. Wang, A. Tanwer, *et al.*, "DeFog: Fog computing benchmarks," in *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, Virginia, VA, USA, pp. 47–58, 2019.

[24] S. Sharma and H. Saini, "A novel four-tier architecture for delay aware scheduling and load balancing in fog environment," *Sustainable Computing: Informatics and Systems*, vol. 24, article no. 100355, 2019.

[25] H. Benadji, L. Zitoune, and V. Vèque, "Performances evaluation and congestion analysis in IoT network: Simulation and emulation approach," in *2022 IEEE International Mediter-*

ranean Conference on Communications and Networking (MeditCom), Athens, Greece, pp. 232–237, 2022.

[26] M. Ashouri, F. Lorig, P. Davidsson, *et al.*, "Edge computing simulators for IoT system design: An analysis of qualities and metrics," *Future Internet*, vol. 11, no. 11, article no. 235, 2019.

[27] X. D. Zhao, J. W. Huang, L. Liu, *et al.*, "Automated performance evaluation for multi-tier cloud service systems subject to mixed workloads," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA, pp. 2630–2631, 2017.

[28] J. Y. Liang, B. W. Ma, Z. H. Feng, *et al.*, "Reliability-aware task processing and offloading for data-intensive applications in edge computing," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4668–4680, 2023.

[29] A. Yousefpour, G. Ishigaki, R. Gour, *et al.*, "On reducing IoT service delay via fog offloading," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 998–1010, 2018.

[30] J. W. Huang, H. Gao, S. H. Wan, *et al.*, "AoI-aware energy control and computation offloading for industrial IoT," *Future Generation Computer Systems*, vol. 139, pp. 29–37, 2023.

[31] M. Y. Mei, M. W. Yao, Q. H. Yang, *et al.*, "Delay analysis of mobile edge computing using Poisson cluster process modeling: A stochastic network calculus perspective," *IEEE Transactions on Communications*, vol. 70, no. 4, pp. 2532–2546, 2022.

[32] Y. Narimani, E. Zeinali, and A. Mirzaei, "QoS-aware resource allocation and fault tolerant operation in hybrid SDN using stochastic network calculus," *Physical Communication*, vol. 53, article no. 101709, 2022.

[33] X. K. Wang, X. Chen, Z. Li, *et al.*, "Access delay analysis and optimization of NB-IoT based on stochastic network calculus," in *2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*, Xi'an, China, pp. 23–28, 2018.

[34] W. M. Zuberek, "Performance evaluation using unbounded timed Petri nets," in *Proceedings of the 3rd International Workshop on Petri Nets and Performance Models*, Kyoto, Japan, pp. 180–186, 1989.

[35] D. Carvalho, L. Rodrigues, P. T. Endo, *et al.*, "Mobile edge computing performance evaluation using stochastic petri nets," in *2020 IEEE Symposium on Computers and Communications (ISCC)*, Rennes, France, pp. 1–6, 2020.

[36] J. W. Huang, M. Wang, Y. Wu, *et al.*, "Distributed offloading in overlapping areas of mobile-edge computing for internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13837–13847, 2022.

[37] Q. Li, S. G. Wang, A. Zhou, *et al.*, "QoS driven task offloading with statistical guarantee in mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 1, pp. 278–290, 2022.

[38] S. J. Shao, Y. Li, S. Y. Guo, *et al.*, "Delay and energy consumption oriented UAV inspection business collaboration computing mechanism in edge computing based electric power IoT," *Chinese Journal of Electronics*, vol. 32, no. 1, pp. 13–25, 2023.

[39] X. M. An, R. F. Fan, H. Hu, *et al.*, "Joint task offloading and resource allocation for IoT edge computing with sequential task dependency," *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 16546–16561, 2022.

[40] Y. Liu, Q. He, D. Q. Zheng, *et al.*, "Data caching optimization in the edge computing environment," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 2074–2085, 2022.

[41] G. L. Zhang, S. Zhang, W. Q. Zhang, *et al.*, "Joint service caching, computation offloading and resource allocation in mobile edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 5288–5300, 2021.

[42] Y. Chen, Y. J. Sun, C. Y. Wang, *et al.*, "Dynamic task allocation and service migration in edge-cloud IoT system based on deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16742–16757, 2022.

[43] F. Z. Liu, H. Yu, J. W. Huang, *et al.*, "Joint service migration and resource allocation in edge IoT system based on deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 11, no. 7, pp. 11341–11352, 2024.

[44] Q. L. Peng, Y. N. Xia, Z. Feng, *et al.*, "Mobility-aware and migration-enabled online edge user allocation in mobile edge computing," in *2019 IEEE International Conference on Web Services (ICWS)*, Milan, Italy, pp. 91–98, 2019.

[45] F. Z. Liu, B. F. Lv, J. W. Huang, *et al.*, "Edge user allocation in overlap areas for mobile edge computing," *Mobile Networks and Applications*, vol. 26, no. 6, pp. 2423–2433, 2021.

[46] F. Z. Liu, J. W. Huang, and X. B. Wang, "Joint task offloading and resource allocation for device-edge-cloud collaboration with subtask dependencies," *IEEE Transactions on Cloud Computing*, vol. 11, no. 3, pp. 3027–3039, 2023.

[47] S. Y. Li, J. W. Huang, and B. Cheng, "A price-incentive resource auction mechanism balancing the interests between users and cloud service provider," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2030–2045, 2021.

[48] S. Y. Li, J. W. Huang, and B. Cheng, "Resource pricing and demand allocation for revenue maximization in IaaS clouds: A market-oriented approach," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3460–3475, 2021.

[49] W. W. Lu, S. L. Gong, and Y. H. Zhu, "Timely data delivery for energy-harvesting IoT devices," *Chinese Journal of Electronics*, vol. 31, no. 2, pp. 322–336, 2022.

[50] F. J. Zhao, Y. Chen, Y. C. Zhang, *et al.*, "Dynamic offloading and resource scheduling for mobile-edge computing with energy harvesting devices," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2154–2165, 2021.

[51] S. Sundar and B. Liang, "Offloading dependent tasks with communication delay and deadline constraint," in *IEEE INFOCOM 2018 IEEE Conference on Computer Communications*, Honolulu, HI, USA, pp. 37–45, 2018.

[52] Z. Z. Liang, Y. Liu, T. M. Lok, *et al.*, "Multi-cell mobile edge computing: Joint service migration and resource allocation," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 5898–5912, 2021.

[53] Y. Chen, J. Zhao, Y. Wu, *et al.*, "QoE-aware decentralized task offloading and resource allocation for end-edge-cloud systems: A game-theoretical approach," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 769–784, 2024.

[54] J. W. Huang, B. W. Ma, M. Wang, *et al.*, "Incentive mechanism design of federated learning for recommendation systems in MEC," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 2596–2607, 2024.

[55] Y. Chen, H. Xing, S. Chen, *et al.*, "Game-based channel selection for UAV services in mobile edge computing," *Security and Communication Networks*, vol. 2022, article no. 4827956, 2022.

[56] Y. F. Tan, S. Ali, H. T. Wang, *et al.*, "An OO-based approach of computing offloading and resource allocation for large-scale mobile edge computing systems," in *17th EAI International Conference on Collaborative Computing: Networking, Applications and Worksharing*, Virtual Event, pp. 65–83, 2021.

[57] J. W. Huang, Y. H. Lan, and M. F. Xu, "A simulation-based approach of QoS-aware service selection in mobile edge computing," *Wireless Communications and Mobile Computing*, vol. 2018, article no. 5485461, 2018.

[58] S. Q. Wang, R. Urgaonkar, M. Zafer, *et al.*, "Dynamic ser-

vice migration in mobile edge computing based on Markov decision process," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 1272–1288, 2019.

[59] X. F. Zhang, J. Zhang, Z. T. Liu, *et al.*, "MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3296–3309, 2020.

[60] F. Huang, G. X. Li, H. C. Wang, *et al.*, "Navigation for UAV pair-supported relaying in unknown IoT systems with deep reinforcement learning," *Chinese Journal of Electronics*, vol. 31, no. 3, pp. 416–429, 2022.

[61] H. C. Ke, J. Wang, L. Y. Deng, *et al.*, "Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7916–7929, 2020.

[62] Y. Zhou, H. Ge, B. W. Ma, *et al.*, "Collaborative task offloading and resource allocation with hybrid energy supply for UAV-assisted multi-clouds," *Journal of Cloud Computing*, vol. 11, no. 1, article no. 42, 2022.

[63] J. W. Huang, B. F. Lv, Y. Wu, *et al.*, "Dynamic admission control and resource allocation for mobile edge computing enabled small cell network," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1964–1973, 2022.

[64] B. McMahan, E. Moore, D. Ramage, *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *20th International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, FL, USA, pp. 1273–1282, 2017.

[65] X. F. Wang, Y. W. Han, V. C. M. Leung, *et al.*, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[66] L. Z. Li, K. Ota, and M. X. Dong, "Deep learning for smart industry: Efficient manufacture inspection system with fog computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.

[67] Y. P. Kang, J. Hauswald, C. Gao, *et al.*, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 615–629, 2017.

[68] S. Teerapittayanon, B. McDanel, and H. T. Kung, "BranchyNet: Fast inference via early exiting from deep neural networks," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Cancun, Mexico, pp. 2464–2469, 2016.

[69] S. C. Liu, Y. Y. Lin, Z. M. Zhou, *et al.*, "On-demand deep model compression for mobile devices: A usage-driven model selection framework," in *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, Munich, Germany, pp. 389–400, 2018.

[70] B. Taylor, V. S. Marco, W. Wolff, *et al.*, "Adaptive deep learning model selection on embedded systems," *ACM SIGPLAN Notices*, vol. 53, no. 6, pp. 31–43, 2018.

[71] Z. Y. Zhou, Z. H. Jia, H. J. Liao, *et al.*, "Secure and latency-aware digital twin assisted resource scheduling for 5G edge computing-empowered distribution grids," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4933–4943, 2022.

[72] B. Li, W. C. Xie, Y. H. Ye, *et al.*, "FlexEdge: Digital twin-enabled task offloading for UAV-aided vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 8, pp. 11086–11091, 2023.

[73] Y. Zhou, J. Wu, X. Lin, *et al.*, "Secure digital twin migration in edge-based autonomous driving system," *IEEE Consumer Electronics Magazine*, vol. 12, no. 6, pp. 56–65, 2023.

Jiwei HUANG was born in 1987. He received the B.E. and Ph.D. degrees in computer science and technology from Tsinghua University, Beijing, China, in 2009 and 2014, respectively. He is currently a Professor and the Vice Dean of the College of Information Science and Engineering / College of Artificial Intelligence, China University of Petroleum, Beijing, China, and the Director of the Beijing Key Laboratory of Petroleum Data Mining. His research interests include Internet of things, edge computing, and services computing.
(Email: huangjw@cup.edu.cn)

Fangzheng LIU was born in 1991. She received the M.S. degree in computer science and technology from North China University of Technology, Beijing, China, in 2017. She is currently pursuing the Ph.D. degree with the College of Information Science and Engineering, China University of Petroleum, Beijing, China. Her current research interests include edge/cloud/services computing as well as performance modeling and optimization.
(Email: 2019310704@student.cup.edu.cn)

Jianbing ZHANG was born in 1974. He received the Ph.D. degree from the Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, China in 2006. He is now an Assistant Professor with the Department of Computer Science and Technology, China University of Petroleum, Beijing, China. His current research interests include web services and geographic information system services.
(Email: zhangjb@cup.edu.cn)