IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 14, 2020, accepted October 5, 2020, date of publication October 8, 2020, date of current version October 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3029649

A Survey and Taxonomy on Task Offloading for Edge-Cloud Computing

BO WANG[1], CHANGHAI WANG[1], WANWEI HUANG[1], YING SONG[2,4], AND XIAOYUN QIN[3]

[1]Software Engineering College, Zhengzhou University of Light Industry, Zhengzhou 450002, China
[2]Beijing Advanced Innovation Center for Materials Genome Engineering, Beijing Information Science and Technology University, Beijing 100101, China
[3]Beijing Key Laboratory of Internet Culture and Digital Dissemination, Beijing Information Science and Technology University, Beijing 100101, China
[4]Department of Material and Chemical Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China

Corresponding author: Xiaoyun Qin (2017039@zzuli.edu.cn)

The research was supported in part by the National Natural Science Foundation of China (Grant No. 61872043, 61975187, 21904120), the Key Science and Technology Program of Henan Province (Grant No. 202102210174, 202102210383, 202102210149), the Key Scientific Research Projects of Henan Higher School (Grant No. 19A520043, 20B520039, 21A520050), Qin Xin Talents Cultivation Program, Beijing Information Science and Technology University (No. QXTCP B201904), the fund of the Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (Grant No. ICDDXN004), the Foundation Training Program for Young Key Teachers of Zhengzhou University of Light Industry, and the Research Fund for the Doctoral Program of Zhengzhou University of Light Industry.

ABSTRACT Edge-cloud computing, combining the benefits of both edge computing and cloud computing, is one of the most promising ways to address the resource insufficiency of smart devices. Task offloading is an important challenge must be addressed for edge-cloud computing in practice, which decides the place and the time for performing each task. Even though there is existing research focusing on the task offloading in edge-cloud computing, a lot of problems should be solved before the application of these offloading technologies. Thus, in this article, we first propose a taxonomy of task offloading in edge-cloud environments to investigate and classify related research articles, and then summarize several challenges which have not been addressed for future research directions on this area to promote the development of edge-cloud market.

INDEX TERMS Cloud computing, edge computing, edge-cloud, mobile computing, task offloading.

I. INTRODUCTION

Smart devices, e.g., smartphones, wearable devices, and Internet of Thing (IoT) devices, are increasingly popular as the development of information technology and the growing need for improving the quality of human life. As reported in CISCO Annual Internet Report (AIR) report released in 2020 [1], the number of networked devices will be 29.3 billion in the Globe by 2023. Juniper Research has found that the number of IoT devices is going to be 50 billion in 2022 [2]. With the rapid development of artificial intelligence (AI) algorithms (e.g., deep learning), computer network and telecommunication (e.g., 5G technology), in recent years, Internet services provided by/for smart devices has undergone rapid growth in both variety and complexity, e.g., mobile applications and IoT services. AIR report forecasts that connected home applications will have nearly half of Machine-To-Machine share by 2023 and connected car applications will grow the fastest at 30 percent CAGR over 2018-2023 [1].

Nowadays, a lot of smart devices have hardware performance almost equivalent to personal computers. 98% of Android devices have at least 4 cores and some mobile devices are even equipped with GPU [3]. Even so, many applications are hard to perform on a smart device due to its limited battery capacity, compute resources and wireless bandwidth because of its limited size [4], [5]. To address the problem, some work have proposed to employ cloud computing for extending resources of user devices [6], [7] by offloading some of users' tasks to a cloud to improve processing time and computing energy for users devices as the cloud has abundant processing resources. Task offloading increases the transmission data amount of devices, resulting in a longer transmission time and a more transmission energy due to the unstable performance of wireless networks [8], [9], which may cause a shorter device battery life and a poorer application performance.

The above problem can be alleviated by edge computing [10] which pushes computing resources (i.e., edge/fog servers[1]) to the edge of user devices for reducing communication distance, and thus latency. However, the scale of edge

The associate editor coordinating the review of this manuscript and approving it for publication was Chakchai So-In.

[1] In this paper, edge and fog are interchangeable as they both represent the technology of placing some server resources close to user devices to reduce the network distance between devices and clouds in published works.

servers is much smaller than cloud computing as an edge computing center is equipped with only a few servers due to the limitation of space with limited cooling capacity [11], and thus, the edge computing is likely to provide insufficient computing resources for satisfying all requirements of users' tasks.

By combining the benefits of cloud computing and edge computing, edge-cloud computing[2] is one of the most promising ways to address all of above problems for improving the battery lifetime and application performance for user devices. Edge-cloud computing [12] performs each task on a user device, an edge[3] or a cloud, which can provide better computing performance and transmission performance compared with edge computing or cloud computing in overall. While the task offloading is one of the most challenge problems must be addressed for improving the resource utilization efficiency in edge-clouds [11].

Task offloading in an edge-cloud is to decide which tasks are offloaded from user devices, which edge or cloud is an offloaded task assigned to, and further which server each offloaded task performs on in what order. These decisions are hard to make as an optimal offloading solution must understand heterogeneous resources, user requirements, complex networks, user mobilities, task dependences, and so on. For edge-clouds, the task offloading problem is much more complex than that for clouds or edges as it not only has all of challenges for task offloading on both mobile clouds and edges, but also introduces new ones, such as the heterogeneity between edges and clouds in terms of various resources, a more complex network, the decision of which edge or cloud each offloaded task assigned to, and so on.

Therefore, in this paper, we survey published articles about task offloading in cooperative edge and cloud computing, to sum up problems need to be solved for future research. We first present a comprehensive taxonomy of task offloading in edge-cloud environments, and based on the taxonomy, investigate related research works in detail. Then we discuss challenges which have not been addressed, and suggest several promising directions for future research. We hope our review work is helpful for academia and industry concerning service provisioning in edge-cloud computing.

The rest of this paper is organized as follows. Section 2 presents the background about edge-cloud computing, which is helpful to understand the remainder of the paper. Section 3 introduces in detail the comprehensive taxonomy of workload scheduling on task offloading in edge-cloud computing and Section 4 reviews related works in detail. Section 5 summarizes challenges and opportunities for future work. And finally, Section 6 concludes the paper.

II. BACKGROUND

In this section, we first provide a simple overview of edge-cloud computing environment, and present several representative cases of task offloading in edge-clouds, which are helpful to understand the remainder of the paper. Then, we present the previous work surveying articles related to task offloading, and the search method for the related literatures reviewed in this paper.

A. EDGE-CLOUD COMPUTING ENVIRONMENT

As shown in Fig. 1, there are three tiers, device tier, edge tier, and cloud tier, in edge-cloud computing. In the device tier, each of various user devices performs its tasks locally, and offloads some of its tasks to edge servers or cloud servers when its local resources are insufficient for finishing all of its tasks. The decision of which tasks being offloaded is made by either the user or the service provider, which will be illustrated in Section III-B. User devices have network connections with several edge-cloud servers by various network access points (AP), e.g. a micro base station (MiBS), a router, in different scenarios, for the data transmission required by offloaded tasks. User devices include intelligent furnitures in the scene of smart home [13], signal lights, cameras and vehicles in intelligent transportations [14], smartphones and tablets in mobile computing [15], and so on.

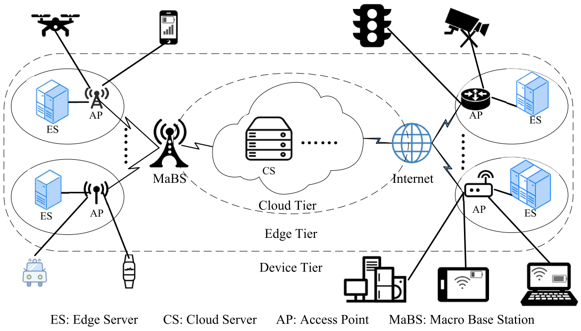

FIGURE 1. Edge-cloud computing scenario.

In the edge tier, there are one or more edge centers (short for edges), each of which is composed of one or more edge servers communicating with some user devices for performing offloaded tasks by corresponding APs. An edge server has a connection with some other edge servers [11], [16] or the cloud [16] tier for "borrowing" resources when it cannot complete all tasks offloaded to it. Due to the limitation of spaces and auxiliary equipments, an edge usually has only a few servers, and thus the cloud tier is needed for serving users as edge resources are not enough sometimes when user loads are high.

The cloud tier provides resources for completing offloaded tasks by private clouds, public clouds, or hybrid clouds [17] when edge resources are insufficient. The cloud tier has abundant computing resources while a poor network performance

[2] In this paper, we use the terminology of edge-cloud computing or edge-cloud to represent the platform consisted of user devices, edge(s) and cloud(s).

[3] In the paper, edge is short for edge computing when not coursing confusions.

for data transmission of offloaded tasks because of the sharing of many other cloud users and the long transmission distance of its connection with other tiers.

B. SERVICE CASES

Now, we illustrate several cases for users offloading their tasks in the edge-cloud environments, to help readers to understand the task offloading problem. As shown in Fig. 2, we consider a simple computing environment composed of two edges and one cloud, and the following four cases which represent, we believe, basic components of the vast majority of real scenarios.

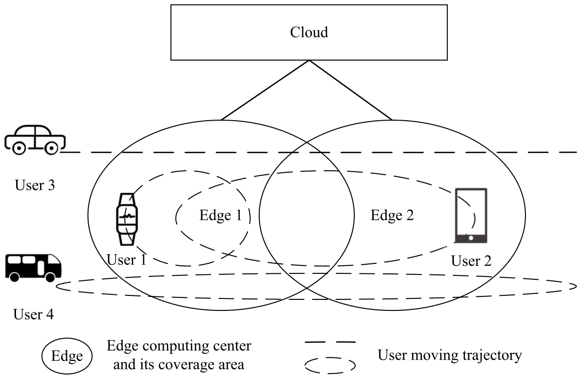

FIGURE 2. Some cases of users with various mobilities.

In the considered environment, users can communicate directly with an edge only if they are in the coverage area of the edge, and thus their tasks can be offloaded directly to the edge. The coverage area between two edges may have a overlap. These two edges have a connection, and one edge can offload some of its tasks to another when it has insufficient resources. In addition, each user or each edge can offload its tasks to the cloud if necessary.

1) CASE 1: USER 1 MOVING WITHIN ONE EDGE

There are a lot of real world scenes where users move in a small area, such as within the coverage area of an edge. For example, in a hospital, some patients have acute diseases that may be occur at any time. Nurses and doctors have to pay close attention to their status information collected by some devices (e.g., user 1 in Fig. 2) and analysed by some smart health predicting methods [18] continuously, to avoid the onset of these acute diseases or prompt first aid treatment. Another example is elderly people monitored with smart devices giving the health information to employees in a smart nursing home [19]. These elderly people take activities within the nursing home almost all the time. In this case, there are three options for processing tasks of user 1: (i) the user device if it has (enough) computing resources, (ii) edge 1 when it provides a better performance than the device, and (iii) the cloud when the device and the edge both have no enough resources and it can satisfy user requirements.

2) CASE 2: USER 2 MOVING ACROSS EDGES

The second case is that, see user 2 shown in Fig. 2, the user has a wider activity space than users in case 1, where the coverage area of an edge cannot cover the activity space. In addition, the user is always in the coverage area of at least one edge during a period of time. A representative example is a traveller visiting scenic spots of a city in a day, who takes videos and pictures frequently using its smartphone and shares them with its net friends via e.g., twitter and facebook, after processing by various algorithms [20]. The traveller may across coverage areas of two or more edges. Another example is the survivor search and rescue, requiring strictly for task execution delay, after medium- and large-scale disasters whose scopes cannot be coverred by only one edge, e.g., Sichuan earthquake [21], where mobile devices, including robots, UVAs, and so on, are dedicated to searching and locating survivors based on the collection and analysis of various data [22], [23]. In this case, the user moves from one edge (edge 1) to another (edge 2), where edge 1 may be executing some tasks offloaded by the user. In this situation, there are three options to handle these offloaded tasks: (i) finishing these tasks in edge 1 and returning the result to the user through edge 2 or the cloud, (ii) migrating these tasks to edge 2 or the cloud, which incurs migration overheads [23], (iii) restarting these tasks in the user device, edge 2 or the cloud, which is same as case 1.

3) CASE 3: USER 3 MOVING OUTSIDE EDGES

In real world, there are some user devices being high mobility, e.g., vehicles, and they cannot be covered by any edge sometimes. For example, a vehicle is traveling in a city (see user 3 in Fig. 2), running various tasks, such as real-time route planning [24], surrounding environmental information collecting [25], and moves into some areas having no direct connection with any edge. In this circumstance, tasks can be offloaded only to the cloud or other user devices as illustrated in case 4, when the vehicle equipped with insufficient resources. There is also a possible situation that the vehicle is moving from an area uncovered by no edge to the coverage area of an edge. Then for the vehicle, some tasks offloaded to the cloud can be assigned to the edge for execution in advance, or their results can be transmitted to the edge ahead, to improve their delay.

4) CASE 4: USER 3 HAVING NEIGHBOUR USER 4

There are two or more users whose positions are close enough to establish a stable network connection, such as within the coverage area of an edge, during some periods and thus they can execute tasks cooperatively, i.e., one user device can process some tasks of another when it has redundant resources [26]. For example, user 3 and user 4 in Fig. 2 have similar move pattern and are always quit nearly, such as less than a diameter of an edge, during a period. Then if user 3 has some tasks need to be offloaded, user 4 can accept its offloading request even when user 3 is not in any coverage area of edges.

C. RELATED SURVEY WORKS

As edge computing has become one of the popular solutions to improve the performance of user tasks for smart devices, and task offloading is a key function for the task performance and resource usage optimizations, there are several articles focusing on the survey of offloading technologies in edge computing. Heidari *et al.* [27] focused on the offloading for IoT in various computing environments, and reviewed 18 offloading approaches achieved by their article selection process in three aspects of computing environment, offloading scheme, and decision making process characteristic. Only 4 offloading works concerned edge-cloud computing environments in this review. Aazam *et al.* [28] reviewed 12 articles related to the task offloading in detail, of which only 4 concerned both edges and clouds for offloaded task execution. Wang *et al.* [29] reviewed 17 offloading methods for edge computing environments in five dimensions of offloading destination, load balance of edge servers, device mobility, application partitioning and partition granularity. Shakarami *et al.* [30] reviewed the computation offloading approaches based on game-theoretic for mobile edge computing. Salaht *et al.* [31] surveyed works addressing the service placement problem in edge computing. Yang and Rahmani [32] reviewed 15 articles designing heuristic or meta-heuristic task scheduling mechanisms in fog computing. Mach and Becvar [33] surveyed various edge computing architectures and reviewed several research works focusing on offloading decision (22 articles), resource allocation (10 articles), and user mobility management (16 articles) in mobile edge computing environments. These above review works did not concern whether the related works exploiting cloud resources for processing offloaded tasks.

There are some works surveying various technologies employed for implementing or optimizing the operation of edge/fog computing, where task offloading is one kind. Cong *et al.* [26] presented a hierarchical survey on the energy optimization methods for mobile devices, from hardware technologies to applications. This article mainly surveyed methods of offloading tasks from a user device to another device, edge servers, and remote clouds, while it reviewed only 4 works concerning both edge and cloud resources when make task offloading decision. Duc *et al.* [34] reviewed various technologies can be applied for reliable resource provisioning in edge-cloud computing, where computation offloading was a very small part of their works, including only 3 literatures. Yousefpour *et al.* [35] comprehensively surveyed research topics in fog computing, where task offloading/scheduling is 1 of 17 topics. This work tried to provide a complete view of fog computing, and thus did not detailedly review any related work. Rahimi *et al.* [36] presented a systematic literature review method for fog-based smart homes, and reviewed selected 22 related resource and service management approaches, including 7 task scheduling/offloading methods for smart home-fog-cloud environments.

Different from existed survey works, this paper mainly dedicates to achieving exhaustive and a clear overview of the task offloading technology with the collaboration of edges and clouds. We hope that our work is helpful for both research and business in edge and cloud operations.

D. REVIEWED LITERATURE SEARCH

The literatures we reviewed in this paper include the followings:

(1) the relevant literatures obtained by querying the Engineering Village Compendex database[4] and the Web of Science Core Collection[5] with the searching conditions (in the form of the query statement in Engineering Village Compendex database), (''edge cloud'' OR ''fog cloud'') AND (''task scheduling'' OR ''task offloading'' OR ''workload scheduling'' OR ''workload offloading'' OR ''workflow scheduling'' OR ''workflow offloading'' OR ''computing offloading'' OR ''computation offloading''), to cover as many high quality research articles as possible;

(2) and the relevant literatures citing the literatures obtained in (1) and (2), which were achieved by Google Scholar[6] (a recursive procedure), to cover as many newly published works as possible.

After searching and selecting related works by above two steps filtering out literatures not concerning e.g., the coordination of edge and cloud computing, the task offloading strategy, etc., we achieve 71 related published literatures. The number of publications in each year is shown in Fig. 3. As shown in the figure, the related article number grew at a compounded annual rate of 134% at last 4 years, which illustrates that more and more researcher taking an interest in the task offloading problem at cooperative edge and cloud computing environments, making our work meaningful.

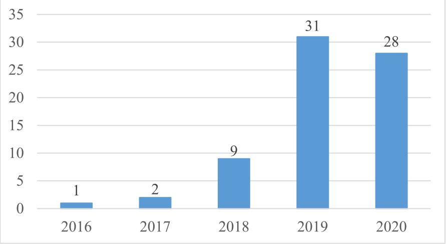

FIGURE 3. The number of related publications in each year.

III. TAXONOMY

In this section, we present a taxonomy to classify task offloading methods, based on characteristics of the problems they focused on, for helping us to summarize the challenges not addressed by existing researches. We classify related works

[4] Engineering Village. https://www.engineeringvillage.com/, accessed Sep 2020.

[5] Web of Science. http://apps.webofknowledge.com/, accessed Sep 2020.

[6] Google Scholar. https://scholar.google.com/, accessed Sep 2020.

in six ways according to the properties of task offloading approaches they proposed, as shown in Fig. 4. These classifications can help us to review related works in detail and summarize them for leading out challenges and opportunities of optimizing the resource usage in edge-clouds. Each category of every related work are listed in Table 1. The taxonomy is detailed as followings.

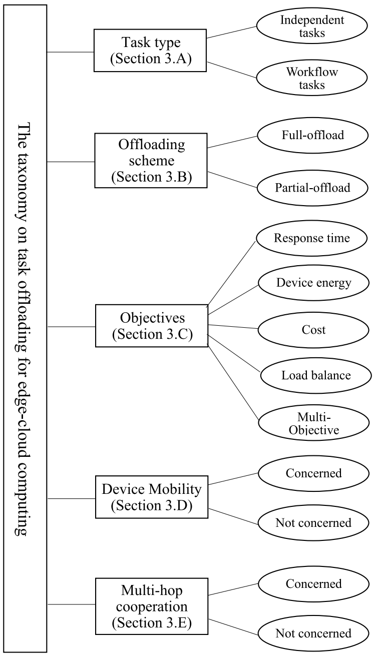

FIGURE 4. The taxonomy on task offloading for edge-cloud computing.

A. TASK TYPE

There are various tasks for offloading from user devices, and tasks with distinct types have requirements of resources with various characteristics. Thus tasks with different types should be repectively managed by different methods to achieve the best result [107]. The task type is a useful differentiating factor for related literatures to understand the applicability of an offloading method to a task type.

In general, tasks concerned by existed offloading methods can be classified into two categories, `independent`[7] tasks and `workflow` composed of multiple tasks with data or logic dependences. This classification of related works corresponds to 2nd column in Table 1.

There are plenty of independent tasks various devices perform, such as environmental monitoring of independent sensor networks [108], finger or face recognition of smartphones [109]–[111]. As not concerning data dependence for tasks, the offloading problem is much easier for independent tasks, compared with that for workflows, and thus, the number of researchers focusing on the offloading of independent tasks on edge-clouds is more than that for workflow, as shown in Table 1 and Section 4.1.

While, the world is not always simple, as a lot of applications of smart devices are performing multiple tasks with logic or data dependency relationships, where a task can be started only after its dependent tasks are all finished. For example, in the scenario of autonomous vehicles, a vehicle makes a decision only when it receives all needed information from other vehicles [7], traffic information collector [112], etc. It is difficult for offloading workflows in an edge-cloud, as achieving an optimal solution not only understands the requirement of each task, the complex access network and the user mobility, but also concerns dependences of tasks. Therefore, there are only several literatures studying on the task offloading on edge-cloud computing for workflows (see Table 1 and Section 4.2).

B. OFFLOADING SCHEME

The first step of task offloading is to decide which tasks to be offloaded for each user device. According to the way offloading methods employ, they can be classified into two categories, full-offload (`full`) vs. partial-offload (`partial`). The classification of related works corresponds to 3rd column in Table 1.

Task offloading methods with full-offload assumes that all tasks are offloaded to edges or clouds or the user decides which tasks it offloads, which means that these methods are not concern the offloading decision when conducting task offloading in edge-cloud computing. These methods are suitable for the task offloading in the scene of tasks requested by devices without processing power, e.g., sampling sensors [113], or for users with expert experiences.

Closer to the reality, the offloading decision is made by the service provider concerning user's quality of experience (QoE) or quality of service (QoS), as users hardly have expert experiences in the decision and a majority of smart devices have processing abilities, e.g., smartphones, wearable devices. Thus a portion of related works propose their task offloading methods with an offloading decision. We classified these works into partial-offload in this paper.

C. OPTIMIZATION OBJECTIVE

For task offloading, there are various requirements of both users and the service provider in edge-cloud computing. The users' satisfaction is one of the most important criteria for evaluating the quality of task offloading methods, as it greatly affects the profit of the service provider due to the penalty when there is any violation of service level agreement (SLA) [17], and it largely determines users' willingness to pay for services from the provider in the future [114], [115]. Thus, user requirements must be concerned and satisfied when conducting task offloading. In related works, there are mainly two metrics expressing user requirements, the task response time (`performance`) and the device battery life (equivalent to the `energy` consumed by the device), respectively illustrated in Section III-C1 and III-C2, both of which effect on

[7] We use the typewriter font to represent possible values for categories of related works in table 1.

TABLE 1. The category of each related work for each classification.

Related Work	Task Type	Offloading Scheme	Objectives	Device Mobility	Multi-hop Cooperation
Wu et al. [37]	independent	full	performance		
Yousefpour et al. [38]	independent	full	performance		
Han et al. [39], [40]	independent	full	performance		
Apat et al. [41]	independent	full	performance		
Benblidia et al. [42]	independent	full	performance		
Aburukba et al. [43]	independent	full	performance		
Murtaza et al. [44]	independent	full	performance		
Ren et al. [45]	independent	full	performance		
Ning et al. [46]	independent	full	performance		
Li et al. [47]	independent	full	performance		hop-e
Fan et al. [48]	independent	full	energy		
Gao et al. [49]	independent	full	cost		
Chen et al. [50]	independent	full	cost		
Chen et al. [51]	independent	full	profit		
Yuan et al. [52]	independent	full	profit		
Lin et al. [53]	independent	full	performance, energy		
Du et al. [54]	independent	full	performance, energy		
Duan et al. [55]	independent	full	performance, energy		
Mahmud et al. [56]	independent	full	performance, profit		
Li et al. [57]	independent	full	Performance, cost		
Sun et al. [58]	independent	full	performance, cost		
Adhikari et al. [59]	independent	full	performance, utilization		
Ma et al. [60]	independent	full	QoE, cost		
Miao et al. [61]	independent	partial	performance		
Kai et al. [62]	independent	partial	performance		
Guo et al. [63]	independent	partial	performance		
Meng et al. [64], [65]	independent	partial	performance		hop-e
Cui et al. [66], [67]	independent	partial	performance		hop-d, hop-e
Sarkar et al. [68]	independent	partial	performance		hop-e
Ouyang et al. [69]	independent	partial	performance	Y	
Cheng et al. [70]	independent	partial	energy		
Xia et al. [71]	independent	partial	energy		
Zhang et al. [72]	independent	partial	cost		
Chabbouh et al. [73]	independent	partial	performance, balance	Y	
Wang et al. [74]	independent	partial	performance, cost		
Zhao et al. [75]	independent	partial	performance, cost		
Khayyat et al. [76]	independent	partial	performance, energy		
Alshahrani et al. [77]	independent	partial	performance, energy		
Chen et al. [78]	independent	partial	performance, cost, energy		
Hong et al. [16]	independent	partial	performance, energy		hop-d
Sun et al. [79]	independent	partial	performance, energy		
Long et al. [80]	independent	partial	performance, energy		
Nguyen et al. [81]	independent	partial	performance, energy		
Wang et al. [82]	independent	partial	performance, energy	Y	hop-e
Alkhalaileh et al. [83]	independent	partial	performance, energy		
Wang et al. [84]	independent	partial	performance, energy		
Wu et al. [85]	independent	partial	performance, energy		
Yang et al. [86]	independent	partial	performance, energy		
Shah-Mansouri and Wong [87], [88]	independent	partial	performance, energy		
Adhikari and Gianey [89]	independent	partial	performance, energy		
Peng et al. [90]	independent	partial	performance, energy, utilization		
Du et al. [91]	workflow	full	performance		
Liu et al. [92]	workflow	full	performance		
Haja et al. [93]	workflow	full	performance		hop-e
Meng et al. [94]	workflow	full	cost		
Meng et al. [95]	workflow	full	cost	Y	
Xie et al. [96]	workflow	full	performance, cost		
Gad-Elrab and Noaman [97]	workflow	full	performance, energy		
Sun et al. [98]	workflow	partial	performance	Y	hop-d
Lakhan and Li [99]	workflow	partial	performance		
Lakhan et al. [100]–[102]	workflow	partial	energy		
Zhu et al. [103]	workflow	partial	energy		
Sun et al. [104]	workflow	partial	performance, energy		
Wu et al. [105]	workflow	partial	performance, energy		
Maio and Kimovski [106]	workflow	partial	performance, reliability, cost		

user's QoE [116]. A very few works concern the `QoE` as a metric directly.

For the service provider in edge-cloud computing, the resource usage cost (`cost`) or the profit (`profit`) for executing offloaded tasks is concerned in most of related works, and only a few works consider improving the load balance (`balance`) of servers in each edge or cloud [73], as shown in Table 1. While, other important factors affecting user QoE and provider profits, e.g. reliability [117], security [118], are not taken into account, which is a challenge must be addressed in future researches. The 4th column in Table 1 shows the main factors concerned by related works for user and provider requirements.

1) RESPONSE TIME

The response time is the time between submitting the request and receiving the result for a task, which is usually considered as a QoS metric in the design of offloading methods. The response time is also expressed as the finish time or the delay in some cases of related works shown in Section IV. The response time of a task is influenced by many factors as followings,

(1) the communication delay between the device and an edge when the task is offloaded to the edge,
(2) the communication delay between the device and a cloud when the task is offloaded to the cloud,
(3) the communication delay between two devices, when the task is offloaded from one device to another device applying device multi-hop cooperation as illustrated in Section III-E, or when two dependent tasks are executed on two devices respectively, e.g., unmanned aerial vehicle cooperation [119],
(4) the communication delay between two edges, when the task is offloaded from one edge to another edge applying edge multi-hop cooperation as illustrated in Section III-E, or when two dependent tasks are offloaded to two edges respectively,
(5) the communication delay between two clouds, when two dependent tasks are offloaded to two clouds respectively applying multi-cloud [17], [120],
(6) the processing delay on a device, when the task is executed on the device
(7) the processing delay on an edge, when the task is offloaded to the edge, and
(8) the processing delay on a cloud, when the task is offloaded to the cloud.

The complexity of the task response time makes it difficult to achieve an optimal solution of task offloading, and thus related works only concern part of the above factors in various cases of edge-cloud computing for simplification, detailed in Section IV.

There are three ways to take the response time optimization into account for task offloading. They can be achieved by optimizing

(1) the average response time of all tasks [98], which probably leads to some tasks with long response time, and thus results in a long tail of response time [121],
(2) the long-term average response time of each task [78], which fails to handle short term fluctuation cases, and thus may result in some SLA violations, and
(3) the response time of each task to ensure the requirement is satisfied for every user.

2) DEVICE ENERGY

Due to the limitation of battery capacity for various smart devices, the energy consumed by the device finishing tasks is one of the most concern for the user. For a device finishing a task, the energy are mostly consumed by data processing and data transmission. For both kinds of consumed energy, all of existed related works utilize simple linear models where the consumed energy is linearly increased with the amount processed data or transmitted data. There are also three ways to tackle the consumed energy in task offloading, similar to the response time. When designing task offloading methods concerning the device energy, one must study on the trade-off among response time, process energy, and transmission energy, as the offloading of a task can improve its response time while decreases the process energy by reduce the processed data size but increases the transmission energy by increasing the transmit data size.

3) COST

For the service provider of an edge-cloud computing, the profit is of the most concern, which depends on the cost of edge and cloud resource usages and sometimes the penalty cost due to SLA violations [17]. There are mainly two kinds of resources, private resources and public resources, used for finishing offloaded tasks, for the service provider. The private resource is the provider's owned resource which can be private edge resources and private clouds, while the public resource is the resources rented from public clouds. The cost of private resources is their operation cost, while the public resources are charged usually on the basis of unit time, e.g., hour. [122], [123]. The different cost models of these two kinds of resources should be considered if they are both used for executing offloaded tasks.

4) LOAD BALANCE

In an edge or a cloud, there are multiple servers (or virtual servers) for processing offloaded tasks. The imbalance of loads in these servers leads to various response times for offloaded tasks, and thus may result in a long tail. Therefore, there are existing works concerning the load balance in each edge and each cloud for task offloading on edge-cloud computing [73].

5) MULTIPLE OBJECTIVES

As there are various requirements for task offloading, it is not true in practice to consider only one objective. Thus, multi-objective optimization must be studied for the application of task offloading technologies. There are three ways for studying multiple objectives for task offloading, (1) transforming multiple objectives to one by some approaches, e.g., weighted-adding, which must decide the importance (weight) of each objective, (2) selecting one (the most important) of these objectives as the optimization one and others as constraints with lower/upper bounds, (3) designing solving method for providing various Pareto-optimal solutions [124] of these objectives for providers' choices according to practical conditions.

D. DEVICE MOBILITY

In many cases, the location of user devices are dynamic, e.g., smartphones of travellers, unmanned vehicles. The device location often decides the edge tasks of the device are offloaded to, because a task is usually offloaded to the closest edge for a low network latency. Thus, the location of user devices, i.e., device mobility, must be concerned when performing task offloading [73], [98]. The concern of device mobility for related works corresponds to the 5th column in Table 1, where '`Y`' represents the corresponding related work 'concerns' the device mobility, and a blank cell means 'not concerns'.

E. MULTI-HOP COOPERATIVE OFFLOADING

Usually, edges are deployed on various areas with relatively more users for a low overall network latency. While, the user requirement of resource amount is changed with time in an area. At a time, the resource requirements are various for different areas. These may lead to the frequently occurrence of load imbalance between edges. Then the edge with overload can assign some of its offloaded tasks to another edge with light load or no-load to achieve a lower latency than to a cloud [28], which is called multi-hop cooperative offloading [125].

Multi-hop cooperative offloading can be applied in two scenarios of connected devices (`hop-d`) [98] and connected edges (`hop-e`) [28]. In the first scenario, a task of device A is offloaded to another device B instead of an edge or a cloud when B provides a better performance for the task [16], [98]. The concern of multi-hop cooperation may improve the performance of a task offloading while increase much more complexity for an optimal solution, and thus there are only a few related works concerning it. The 6th column at Table 1 shows which multi-hop cooperation related works concern, where a blank cell means 'not concern'.

IV. LITERATURE REVIEW

In this section, we review each related work in detail, organizing them with a three hierarchical classification, task type, offloading scheme, and their optimization objectives.

A. INDEPENDENT TASKS

1) ALL OFFLOADING

a: RESPONSE TIME OPTIMIZATION

Wu *et al.* [37] study on the optimization of the overall response time for all user requests in an edge-cloud. They formulate the problem as a integer linear programming model, and design a hybrid heuristic method by combining the powerful global optimizing ability of genetic algorithm and the relatively reasonable selection strategy of simulated annealing to solve the problem. This work assumes that cloud resources are so abundant that the cloud provides no waiting time for offloaded tasks, and does not concern the resource provisioning in the cloud.

For optimizing the delay of offloaded tasks, Yousefpour *et al.* [38] propose an easy online task offloading. With the proposed method, a fog server accepts the new offloading request if the waiting time is below a set threshold, and otherwise, sends the request to a neighbour fog server if the count of sending the request is smaller than a given value, and to the cloud if not. There are two main parameters, the threshold for deciding the accept of a request on a fog server and the count for deciding whether offloading a request to the cloud, must be set, where their settings require expertise, which limits this work's application.

Apat *et al.* [41] present a task assignment method for improving the response time to users by maximizing the number of tasks assigned to edge servers. The method iteratively assigns the task with minimal slack time to the edge server with minimum length to the user, and assigns remaining tasks cannot finished in edges to the cloud. The resource provisioning in the cloud and the scheduling in a server is not concerned in this work.

Han *et al.* [39], [40] propose an online task dispatching and scheduling algorithm, OnDisc, for improving the total weighted response time (WRT) of all jobs. For each new offloaded task, OnDisc heuristically dispatches if to the server providing the shortest additional WRT, where the cloud is seen as a server. In every server, OnDisc executes the task with the minimum ratio between its weight and processing time first, where the weight indicates how sensitive the task is to the delay, and considers that the new task can preempt the executing task if it has a lower ratio.

Benblidia *et al.* [42] present a ranking based task scheduling method using linguistic and fuzzy quantified proposition, to rank each fog node (or the cloud) for a user in its preference of the distance to the user device, the service price, the latency, the bandwidth and the reliability. They always assign offloaded tasks to the first ranked node. In this work, the satisfaction of user requirements, e.g., deadline constraints, is not concerned, and thus not guaranteed.

Aburukba *et al.* [43] study on the optimization of weighted sum of all request delays with deadline constraints in IoT-edge-cloud environments to allocate an edge-cloud resource (processor) for each task's execution. They formulate the optimization problem as a Mixed Integer Linear Programming (MILP) problem which is NP-Hard, and propose a genetic algorithm to solve the MILP problem, where each gene represents a map between a task and an edge-cloud resource.

Murtaza *et al.* [44] present an adaptive approach based on the idea of rule-based learning and case-based reasoning for improving the processing time of tasks in a fog-cloud environment. They use 5% tasks randomly offloaded to fog nodes and the cloud to training the parameters of task execution, such as the propagation time. Based on learned results, they offload each remaining task to the nearest fog node or the cloud when the nearest fog node is busy and the task has a short processing time in the cloud.

Ren *et al.* [45] study on optimizing the overall delay of all tasks in collaborative cloud and edge computing, with assumptions that each user is associatied with an edge server, and that all tasks have the same type and arrive simultaneously. They formulate an optimization problem for minimizing weighted-sum delay of all tasks, and decompose the problem into two subproblems based on the independence of communication and computation resource allocations. The first one is to minimize the weighted transmission delay between devices and edge nodes, which can be directly solved by mathematical derivation. Another subproblem is minimizing the weighted computing delay of edge nodes and the cloud server, which can be transformed into a convex optimization problem can be solved by KKT conditions.

Ning *et al.* [46] present a task offloading method for minimizing the total delay for all users in edge-clouds. They first formulate the offloading problem for one user and multiple users as MILP problems, named SCOP and MCOP respectively, which both are NP-hard. Then, to solve MCOP, they first adopt the branch and bound method to obtain the initial offloading decision, which is applicable as the very limited number of offloaded tasks for a user. Then to tackle the edge resource conflicts among user requirements, they iteratively modify the offloading position of the task from an edge server with resource conflicts to the cloud such that the increase of the total execution delay is minimal. This work assumes each edge server only processes one task, and the cloud has a fixed process rate for all tasks, which narrows its application range in the real world.

Li *et al.* [47] try to reduce the service delay for IoT tasks offloaded to fog-cloud servers, considering a fog server can offload its tasks to its neighbouring fog servers. They establish an optimization problem for minimizing the long-term average delay in the IoT-fog-cloud system, and apply Lyapunov drift-plus-penalty method to solve the problem. The optimization of the long-term average delay may lead to the performance imbalance among tasks, regions, or time slots, and thus incurs a poor performance for some tasks sometimes.

b: COST/PROFIT OPTIMIZATION

Gao *et al.* [49] transform the task offloading problem as periodical auctions by seeing servers and users as the resource sellers and buyers, respectively, where the cloud is seen as a server, and formulate it as a constrained total profit optimization problem. To solve the problem, they model is as a weighted bipartite graph matching problem with capacity and deadline constraints, and adopt the heuristic method which iteratively selects the edge with the best profits from the bipartite graph. This work only concerns the input data transmission delay for quantifying the performance.

Chen *et al.* [50] formulate the task offloading problem as a long-term average cost optimization with the bound constraint of long-term average queue length in the edge tier. To solve the problem, they exploit Lyapunov optimization techniques simplify the stochastic problem into a set of deterministic optimization subproblems for each time slots. Each subproblem is to optimize the weighted queue length and cost by deciding the number of tasks offloaded to the edge in the time slot, which can be directly addressed due to only one variable need be solved.

Chen *et al.* [51] present a task scheduling method (RCTSPO) for processing offloaded tasks in an edge-cloud providing resources in the form of VMs to optimize the profit, where the value of a task is proportional to the resource amounts and the time it takes. RCTSPO first employs K-means to classify tasks into several classes, where the position of a task is its required resource amounts in computing power, memory capacity, and network bandwidth, and classifies all VMs as one class. Then, RCTSPO calculates the profit for each task in the task class closest to the VM class, and uses Kuhn-Munkres method to solve the optimal matching of tasks in the task class and VMs with profit maximization. This work requires various resources needed by each tasks are known and constants in amount, which does not match reality. The resources are both considered as VMs for offloaded task processing in this work which ignores the heterogeneity between edge and cloud resources, may lead to resource inefficiency [126].

Yuan and Zhou [52] work on the optimization of the profit for edge-cloud providers. They formulate the profit optimization with considerations of the maximum response time constraint for all tasks and the load balance for edge nodes, where the revenue and the penalty cost for each task is specified by SLAs and the cost includes the execution costs of tasks offloaded to edge nodes and the energy cost of task executions in cloud servers. To address the profit maximization problem, authors design a migrating birds optimization (MBO) based method, and adopt simulated-annealing update mechanism to alleviate the local optimal convergence problem of MBO.

c: MULTI-OBJECTIVE OPTIMIZATION

Lin *et al.* [53] employ Hungarian algorithm [127] for selecting the edge-cloud server for each user offloaded task, to optimize the weighted sum of **latency** and consumed **energy** for all user tasks. To improve the fairness of users, they execute the task of low-throughput user first on the edge-cloud. As they offload all tasks to the edge-cloud, the computing energy of user devices is ignored. This work does not concern the resource allocation of edge-cloud servers, and thus is not concerned the heterogeneity between edges and clouds either.

To guarantee the fairness and the performance requirements for users, Du *et al.* [54] study on the task offloading decision of mobile devices and the resource allocation of the edge and cloud tiers, to optimize the minimum weighted sum of **delay** and consumed **energy** for each user. This work first models the problem as a mixed integer non-linear programming problem including two sub-problems of task offloading decision and resource allocation. The work transforms the offloading decision sub-problem to a non-convex quadratically constrained quadratic programming (QCQP) problem by variable substitution and employs semidefinite relaxation (SDR) to convert the QCQP to a standard convex problem, which can be solved using existing convex optimization toolbox. Fractional programming and Lagrange dual decomposition are used to solve the resource allocation sub-problem. This work concerns a simple scenario of an edge-cloud composed of one edge server and one cloud server, and doesn't consider the waiting delay of tasks executing on the edge or the cloud. For all users, the weight values of summing delay and energy are identical for an application of this work, which is not suitable for the various user requirements.

Duan *et al.* [55] focuse on the edge-cloud environment consisting of an edge cloud, and a remot cloud for processing tasks offloaded by multiple FANETs, where FANET is flying ad hoc network constituted by multiple UAVs [128]. For optimizing the time-averaged energy consumption of edge-cloud servers with the constraint of all the tasks can be executed within a finite time delay by joint task scheduling and resource allocation, they first establish a stochastic optimization problem, and then exploit Lyapunov optimization method to transform the problem as multiple convex optimization problems in all of time slots. At last, they solve each convex optimization problem by exploiting the decreasing gradient direction, iteratively optimally solving task scheduling and resource allocation sub-problems. This work considers edges as an edge cloud, ignoring the complex network introduced by geographically distributed edges, and neglects the heterogeneity between edge and cloud resources. These limit the application of their proposed method.

Above work only concern the benefit of users without concerning the resource usage cost optimization. To improve the resource cost and service delivery latency in edge-cloud computing, Mahmud *et al.* [56] formulate a constrained integer linear programming model that maximizes the total profit merit of offloaded applications, where the profit merit is defined as the ratio of the profit and the slack time for each application processing. To solve the problem with a polynomial time, they exploit the best fit method iteratively assigning the offloaded application to the first computational instance such that all constraints are satisfied and the profit merit is minimum, where cloud-based instances are sorted behind edge-based instances. As done by Duan *et al.* [55], the complex network and the resource heterogeneity of the edge-cloud are neglected in this work.

Li *et al.* [57] study on the optimization problem of the finish time and the cloud resource usage cost by task offloading. Their proposed method first decides the place (an edge or the cloud) for the execution of each offloaded task, adopting the artificial fish swarm algorithm. To avoid falling into local optimal solution, they use a modified conventional process by using simulated annealing method to calculate the probability of updating bulletin. Then their method greedily assigns a task to the node with the minimum utilization in an edge or the cloud. In the cloud, for improving the resource usage cost, the method rents the VM with minimum resource amount when the overall load is high, and releases the VM with maximum resource amount when it is low. This work focuses on the task offloading for media delivery applications, and thus considers that a task can be divided into multiple same-sized subtasks for parallel process.

Sun *et al.* [58] propose a method to decide the number of active servers and the request dispatches in the edge-cloud consisting of multiple edges and/or datacenters buying power from the market, taking into account the trade-off between **power cost** and **request latency**. To achieve the active server number for each edge/datacenter in a large time slot, they establish a integer non-linear programming problem with the optimization objective of weighted sum of power cost and request latency, based on the queuing theory assuming requests and servers are respectively homogeneous, and solve the problem applying continuity relaxation and sequential convex programming. According to the solved active server numbers, they employ Lyapunov optimization to formulate the request dispatch problem as a quadratic programming problem which can be solved by existed algorithms easily.

Adhikari *et al.* [59] focus on optimizing the **resource utilization** and the execution **latency** for offloaded task execution in an fog-cloud environment. They formulate the problem as a multiple objective optimization problem, and apply PSO to make offloading decisions for optimizing the weighted sum of the objectives.

To balance the benefits obtained by users, edges and service providers, Ma *et al.* [60] present a non-linear integer programming model for minimizing the overall resource utility of all requests, measured by the weighted sum of **user QoE** and **resource costs**. Then, to solve the model, they consider a pure-strategy game [129] among users, edges, and service providers, to obtain the mapping between user requests and resources of service providers and edges. This work does not concern the task scheduling for an edge or a service provider.

2) PARTIAL OFFLOADING

a: RESPONSE TIME OPTIMIZATION

Miao *et al.* [61] present a task offloading for optimizing the delay for each user task. They propose to use Long Short-Term Memory (LSTM) method for predicting the data size for each task. Based on predicted values, they formulate an optimization problem minimizing the delay to decide the the numbers of data processed locally and in an edge-cloud node for each task, assuming each task can be divided into two subtasks with any data size. To further optimize the delay for each task, they propose to migrate some of its subtasks from allocated edge-cloud node to another node. They make above two decisions for tasks separately without considering the conflict of tasks' resource requirements in the edge-cloud.

Kai *et al.* [62] propose a task offloading method for minimizing the total processing time of all tasks by collaborative user device, an edge server and the cloud. They formulate the optimization problem into a non-convex problem, and exploit successive convex approximation to transform it into a convex optimization problem solved by the iterative method efficiently. In this work, authors assume each task can be divided into three parts in any proportions, respectively processed locally, in the edge server, and in the cloud, which limits the application scope of their work.

Guo *et al.* [63] study on optimizing the average response time of all user requests by address the offloading decision and bandwidth allocation problem in an edge-cloud. They first formulate the problem, and then decompose it into multiple convex subproblems each of which can be solved based on the binary search method and Newton's method. For simplicity, they only consider a cloud server and an edge server, which makes their work being applicable for only a few scenarios.

Meng *et al.* [64], [65] propose an online method, Dedas, trying to maximize the number of tasks that meet the deadlines and minimize the average completion time (ACT) of the tasks, by jointly scheduling of networking and computing resources. In an edge server or the cloud, Dedas inserts the new task in a position or replaces an existing task if there is a deadline violation due to adding the new task, to generate a feasible schedule with the minimum ATC. Based on the schedule method, Dedas dispatches the new task to the edge server such that the number of completed tasks is maximized and ACT is minimized. A task is dispatched to the cloud only if edge resources can not satisfy its requirements. In this work, authors regard a multi-core server as multiple single-core servers, which may result in a load imbalance among cores for a server and thus a low resource efficiency [130]. And they always select the shortest network path which is statically configured for data communications, which may lead to imbalance of network resources.

Cui *et al.* [66], [67] try to optimize the processing delay and communication load in the marine fog-cloud computing environments, where each unmanned surface vehicle can be task generator (TaV), fog node (CoV) for task processing or both, and the remote cloud is considered as a common fog node. They formulate the problem as a Multi-Armed Bandit (MAB) problem where a TaV and each candidate CoV are respectively the player and the action, and solve the MAB problem based on the upper confidence bound algorithm. This work makes the offloading decision for TaVs separately, which avoids the bottleneck of centralized decider due to the distributed manner, but may lead to resource conflicts when multiple TaVs request resources from one CoV in a same time.

Sarkar *et al.* [68] intend to reduce the task processing delay for each user by deciding the position (the device, the nearest fog node, a neighbor fog node, or the cloud server) for each task' processing. They formulate the total delay minimization problem as an integer programming (IP) problem. To address the problem, they first transform the IP problem in to a quadratically constraint quadratic program (QCQP) problem, and then apply semidefinite relaxation method to transform the QCQP problem into a semidefinite programming problem which can be solved efficiently by standard optimization toolbox. This work only concerns the offloading decision, doesn't consider the task scheduling in a fog node or the cloud server.

To address the challenge of user mobility, Ouyang *et al.* [69] concern the performance optimization of the service placement for each user during the operation of an edge-cloud. This work first formulates the problem optimizing the weight of computing delay, communication delay, and the cross-edge service migration delay due to the user mobility for each user. And then they transform the offline service placement problem into the shortest-path problem for the constructed graph with the node of the possible service placement for each time slot and the edge of the communication with the weighted delay, which is solved by the dynamic programming approach. For the online problem, they transform the problem into a contextual multi-armed bandit problem (MAB) by regarding the computation nodes as arms, and solve it by contextual Thompson sampling learning scheme. This work is conducted for one mobile user, which is not match with the real world where an edge-cloud provides services for multiple users.

b: ENERGY OPTIMIZATION

Cheng *et al.* [70] focus on the total energy optimization with each task performance guarantee in an edge-cloud. They first formulate the problem into a binary linear programming which is proofed as a NP-complete problem, where the decision variables represent whether the user is serviced in the device, the edge, and the cloud, respectively. To solve the problem, they first relax the binary constraints and solve the relax linear programming problem by the interior points method. Then based on the solution, they assign 0 and 1 to decision variables with the basic idea of greedily cancel the task with highest resource occupation when the capacity constraint is violated. In this work, authors concern optimizing the total data transmission energy in the whole system, which can lead to imbalance of energy consumption among devices, edges, and the cloud, and ignore the computing energy. They handle the energy consumed by mobile devices, edges, and cloud, equally, which is not practical as the energy is much scarcer for mobile devices [131].

Xia *et al.* [71] study on optimizing the average device energy with delay requirements and energy budgets for task execution in a 5G multi-cell mobile-edge-cloud. They formulate this problem as an ILP problem, and apply the following three steps to solve the ILP problem. First, they relax the integral constraints, and solve the relaxed ILP, which provides candidate positions (the device, some access points, some edges, and some data centers) for each task. Then, they filter the candidate positions with higher energy costs, and at last, they iteratively assign the task incurring the smallest energy cost to the candidate position with the smallest energy cost. Even though a lot of mobile devices are equipped with multiple CPUs nowaday, this work assumes that each device processes only one task simultaneously.

Fan *et al.* [48] propose a method for optimizing the device energy consumption for task execution with deadline constraints in an edge-cloud environment considering content cache in edge servers for some tasks. The proposed method first offloads tasks with cached contents to the corresponding edge servers, and for other tasks, respectively assigns them to the cloud or joint device-edge incurring lower device energy. When a task is decided to be executed in joint device-edge, they consider the task can be portioned into two parts for local execution and offloading to the edge in any proportion, where the proportion decision problem is a linear programming problem which can be easily solved. This work try to minimize the average consumed energy of all devices, which may lead to a large energy usage for some devices, and thus decrease the battery life for these devices.

c: COST OPTIMIZATION

To improve the cost of edge resources for service providers, Zhang *et al.* [72] propose a decentralized task offloading method (DMRA) for edge-clouds. The basis idea of DMAR is to iteratively execute the following procedures: mobile devices sends a task offloading request to their respective closest edge server satisfying all requirements; an edge server accepts the offloading request where the task consumes resources with the minimum amount. An offloaded task is assigned to the cloud when its requirements can not be satisfied by both the device and edges. When an offloaded task is assigned to the cloud, this work doesn't consider to idle some resources of the device or an edge by offloading some non-latency-sensitive tasks to the cloud for finishing the task if the task fails finish because of the poor cloud network performance. The task scheduling problem in an edge server or the cloud is not concerned by this work.

d: MULTI-OBJECTIVE OPTIMIZATION

Chabbouh *et al.* [73] propose an offloading decision method for an edge-cloud and a task scheduling method for the edge tier (Cloud-RRH). The decision method offloads a task from the mobile device when the following conditions are satisfied: the device velocity is lower a predefined threshold; the device has insufficient resources, or the edge tier provides a lower latency and consumed energy compared with the device; the communication channel is in a good state. The designed scheduling method is designed by solving an established mixed-integer problem as a linear program for optimizing the **load balance** in edge and cloud tiers, data **transmission delay**, and task **migration delay** for all offloaded tasks. This work considers containers as the provided resources by edges and clouds, while it doesn't concern the deployment and configuration of used containers, and ignores the heterogeneous of resource provisioning between edges and clouds.

Wang *et al.* [74] study on the tradeoff between delays and costs for vehicular applications exploiting the edge-cloud resources. They first employ a game-theoretic online algorithm to make task offloading decisions, where applications are the players trying to obtain both fewer service delays and smaller rent fees. Then they apply first fit algorithm for allocating edge-cloud resources to offloaded tasks with small workload and large workload, separately. This work assumes each vehicle chooses to communicate with only one edge server, even though vehicles are usually too mobile to be covered by only edge network coverage. In addition, all applications have a same requirement in the performance in this work not exploiting the application heterogeneity, such as allocating less resource for non-time-sensitive tasks for cost saving.

For improving the energy consumed by IoT devices and task delays, Cheng *et al.* [78] design an offloading decision method for an edge-cloud and a task scheduling approach for a edge server. The offloading method first models the offloading decision problem as a Markov decision process, and then adapts the policy gradient method [132] to resolve the problem for optimizing the weighted sum of **task delay**, **device energy** and **server usage cost**. The task scheduling approach models the optimization of overall delay for all tasks on the server as a mixed integer programming, and applies a heuristic method to solve it. This work views the cloud tier as a whole, and does not concern the task scheduling in the cloud. In this work, parameters are same for all user devices, and edge servers are homogeneous. This work boots a virtual machine (VM) for each type of tasks, which may result in too many VMs running on an edge servers, leading to a resource inefficiency due to the serious VM coexistence interference [122].

Sun *et al.* [79] propose a Mixed-Integer Non-linear Programming (MINLP) problem for optimizing the weighted sum of total **task latencies** and total device **energy consumption** for all tasks in a device-edge-cloud with a single edge server and one cloud. As the problem is NP-hard, they first decompose it into two simpler subproblems, offloading decision (0-1 knapsack problem) and resource allocation (convex nonlinear optimal problem), and then iteratively solves these two subproblems as follow. For solving the offloading decision problem, they offload the task with large computing load and transmission rate of the device to the edge, and when offloaded tasks is too many, they iteratively offload the task with smallest processed data size from the edge to the cloud, given a solution of resource allocation. And given the offloading decisions, they adopt Cauchy-Schwards Inequality to resolve the resource allocated to each offloaded task in the edge.

All of these above work assume that every user device has a direct network connection with an edge or a cloud, while it is not alway true in reak work. To address the problem, Hong *et al.* [16] model the task offloading problem as a game, where plays are IoT devices each of which has the objective of minimizing a weighted sum of its **consumed energy** and its **task finish time**, considering that a task of one device can be offloaded to another device besides edge and cloud. To achieve an available solution, they propose a distributed method with the idea of iteratively change its executing place to improve its objective value for each task until the edge-cloud reaches a Nash equilibrium. For simplification, they assume that one device only processes one task, and that the resources are allocated to offloaded tasks evenly for each edge server. The costs for using edge or cloud resources are not concerned by this work. Their proposed distributed method requires user devices broadcasting their respective status frequently which consumes extra energy, leading to shorter battery lives.

Long *et al.* [80] study on the optimization of the overall **energy consumption** and task execution **delay** in a IoT-edge-cloud environment where sensing devices generate tasks to be executed by vehicles, MEC servers and cloud center. They first formulate the problem as a binary linear programming, minimizing the accumulated weight sum of the overall energy consumption and delay for all tasks, and then employ asynchronous advantage actor-critic (A3C), a novel deep reinforcement technique, to solve the problem. In this work, they consider the energy consumption of all computations and transmissions, ignoring the scarcity differences among various devices/servers, such as, the transmission energy of sensing devices is much scarcer than the computation energy of cloud servers. In addition, the optimization of accumulated sum may lead to an imbalance performance among tasks, such as the long tail latency [133].

Nguyen *et al.* [81] concern the optimization of **device energy** and **service delay** for all users in the fog-cloud system composed of one fog server and one cloud server, applying compressing data to reduce the consumed energy and delay for transmitting data of offloaded tasks from user devices to the fog/cloud server. They first establish a min-max optimization problem minimizing the maximum weighted energy and service delay of all users, and transform it into a mixed-integer non-linear programming (MINLP) problem. Then, for solving the problem, they employ the bisection search method to classify users into two sets, the set of users offloading their tasks and the set of remaining users, based on their proved result of users incurring higher weighted energy and service delay by locally executing tasks should have higher priorities for offloading. Thereafter, they we can apply the interior point method to find the minimum fog/cloud resources for offloaded tasks. This work is the first attempt to introduce data compression into task offloading for fog-cloud systems, while it applies data compression for all offloaded tasks without considering the trade-off between incurred computing cost and saved delay cost by data compression.

Wang *et al.* [82] try to optimize the cost, i.e., the weighted sum of task execution **latency** and **energy** consumption for user devices (UEs), considering the device mobility. They formulate the optimization problem as a MINLP problem, and respectively exploit a Gini coefficient-based method (GCFSA) and a genetic based method (ROAGA) to solve the offloading decision and resource allocation problems. GCFSA first locally executes tasks whose average sojourn times is shorter than the transmission time or whose cost cannot be reduce by offloading, and then calculates the income of each UE in each fog node, where the income is calculated based on the cost reduction by offloading. At last, GCFSA iteratively offloads each task to the fog node providing the maximum revenue until the assigned task number reaches the maximum constrained by the channel number for each fog node, where the revenue is the cost reduction achieved by offloading. ROAGA exploits the genetic algorithm to allocate resources to offloaded tasks in each fog node with the objective of revenue maximization. This work uses the cloud to transmit execution results from one fog node to another fog node if a user device moves out the coverage of the first fog node after its task is offloaded to the node, and thus doesn't exploit the benefit of the cloud, e.g., abundant computation resources. This work assumes that the sojourn time of each UE in different fog nodes follows the Gaussian i.i.d, which narrows its application scope.

Zhao *et al.* [75] focus on the optimization of the utility in a vehicular edge-cloud system, where the utility is quantified based on the task **latency** and the edge-cloud resource **cost**. They first formulate the optimization problem into mixed-integer programming problem, which is maximum cardinality bin packing problem and thus NP-hard. Then they decompose the problem into two subproblems of offloading decisions and resource allocations, and solve the problem by iteratively solving one subproblem given the solution of another subproblem, where offloading decisions are solved by game theory with players of vehicles, and resource allocation subproblem is convex and solved by KKT conditions.

Khayyat *et al.* [76] concern the optimization of the vehicular **energy** and the task **execution time** in a vehicular edge-cloud system, assuming that each edge server are equally shared among all connected vehicles and that all edge servers have equal resources. They first formulate the problem into a binary linear programming (BLP) for minimizing the weighted sum of the energy and the execution time for all tasks. To solve the BLP, they exploit distributed deep Q learning method, where the state space and the action space represent tasks' requirements and the binary offloading decision respectively, and use the optimization objective of BLP to be represented as the reward. This work assumes that there is no departure of vehicles in the system even though vehicles are highly dynamic, which limit its application as it would affect the overall performance when a vehicle leaves the system with uncompleted tasks [134].

As done by Khayyat *et al.* [76], Alshahrani *et al.* [77] formulate the task offloading problem into a BLP for minimizing the weighted sum of the **energy** and the **execution time** for VR game tasks. They use the brach and bound method to solve the problem, which can provide the optimal solution but is not applicable for large- or medium-scale problems.

Alkhalaileh *et al.* [83] pay attention to minimize the device energy times the edge-cloud resource usage cost for finishing all tasks before their respective deadlines. They formulate the problem as a mixed integer non-linear programming (MINLP), which is an NP-hard optimisation problem. Due to the convexity of the MINLP problem, the optimisation problem can be transformed into a MILP problem, and they apply the Branch and Bound algorithm (BB) to solve the MILP problem optimally, while BB has a complexity increasing exponentially with the problem size, leading to their method is only applicable for small and medium-sized problem.

Wang *et al.* [84] minimize the total energy-efficiency cost summing weighted execution **latency** and consumed **energy** for all tasks in the a computing environment composed of some mobile phones, an edge with multiple access points, and a central cloud. They address the problem by the following iterative steps for each task: (i) assigning the task to the position (the device, the edge or the cloud) incurring minimal cost; (ii) if the task is executed locally, optimizing the CPU clock frequency of the device; (iii) if the task is offloaded, optimizing the overall transmission power and queue delay for all channels.

Wu *et al.* [85] propose a distributed deep learning method for optimizing the weighted sum of total execution time and total energy consumption of all tasks in a collaborate edge and cloud computing environment. They employ parallel deep neural networks (DNN) with the input of task workloads and the output of offloading decisions, and update labelled data with new generated data to update DNN parameters. This work concerns consumed energies of both user devices and edge-cloud servers, ignoring the different scarcity between them.

Yang *et al.* [86] present an offloading solution for maritime task executions in a ship-edge-cloud computing environment to minimize the weighted latency and energy consumption (the overhead). For each task, they execute it locally if it induces a more overhead to offload it, and otherwise, offload the task to the edge server or the cloud who incurs a less overhead.

Shah-Mansouri and Wong [87], [88] study on the task offloading problem in edge-clouds to improve the task **execution time** and the device **energy** consumption for users. They use a strategic game approach to achieve the Nash equilibrium among users, where each player represents a user with the objective of minimizing the amount of cost reduction achieved by offloading the task, where the cost is quantified by the weighted sum of the execution time and the consumed energy. For simplicity, this work assumes an edge server allocate its resources to tasks offloaded on it evenly.

As done in [87], [88], Adhikari and Gianey [89] focus on improve the execution time and the energy consumption, too. They employ a biologically inspired algorithm, firefly algorithm, to solve the offloading problem optimizing the weighted sum of their focused two objectives, with constraints of power consumption, CO_2 emission rate and temperature emission for each computational node.

Peng *et al.* [90] present a optimization problem with three objectives, minimizing the total execution time, minimizing the total energy consumption of user devices, and maximizing the average resource utilization of edge servers, for task offloading in edge-cloud computing. They first formulate the problem, and then apply SPEA2 [135], an improved strength Pareto evolutionary algorithm, to solve the problem.

B. WORKFLOW

1) ALL OFFLOADING

a: RESPONSE TIME OPTIMIZATION

Du *et al.* [91] try to optimize the execution time of a workflow in the edge-cloud for partitioning all tasks into two sets respectively offloaded to the edge and the cloud. To address the problem, they construct a weighted graph, where the node represents a task with the weight of computing delay, and an edge represents the communication between two corresponding tasks with the weight of the data transmission delay. Between two nodes, there are at most four edges as each task has two possible. Then they distribute the computing delay of each node to edges started from it evenly. After that, they greedily select the edge with the minimum weight, and finally achieve the mapping between offloaded tasks and edge/cloud tiers.

Liu *et al.* [92] iteratively assign each task to the edge server providing minimum completion time, and improve the transferred data amount between servers by the task redundant execution, for improving the finish time of a workflow. The redundant execution is helpful for application response time reduction, while increases the used resource amount. The tradeoff between the response time reduction and the resource usage cost must be studied when applying this work.

Haja *et al.* [93] propose a scheduling solution for optimizing the delay between map and reduce tasks in a edge-cloud environment. The proposed solution first assigns each map task to compute resources based on the data locality: assigning the map task to a server containing its input data; if no such server, assigning the map task to the server in the same rack with the input data; if there is no server satisfying either of these two constraints, calculating the delays from servers containing the data to the least utilized servers of each cluster (edge or data center), and assigning the map task to the one with the smallest delay. Then the scheduling solution assigns each reduce task as close to map tasks as possible regarding network delay, by first calculating the maximum delay between the least utilized server from each cluster and servers executed map tasks, and then assigning the reduce task to the server with the minimum calculated delay. This work only concerns the delay optimization between map and reduce tasks, which may not result to a solution of low response time of the MapReduce job. In addition, this work conducts the scheduling for only one job, and thus doesn't consider the resource efficiency improvement by exploiting the complementarity among various applications.

These above works only focus on the offloading decision, without concerning the task assignment and scheduling in a tier or a server. And they focus on the task offloading for only one workflow application, not taking advantage of resource sharing among various user applications for resource efficiency improvements. These lead to non-global optimal solutions achieved by these works.

b: COST OPTIMIZATION

To address the challenge of the user mobility which can lead to the frequent change of the edge server servicing a user, Meng *et al.* [95] study on the cost optimization of edge and cloud computing resources by designing a task scheduling method for finishing one workflow on an edge-cloud. The scheduling method first applies random waypoint model [136] for predicting mobilities of user devices, and employ the partial critical path (PCP) strategy [137] to assign deadline to each task. Then, the method dynamically allocates resources for each unscheduled tasks by VCG auction mechanism [129] where each task auctions its required resources to edges and public clouds, and adjusts allocated resources for tasks according to the dynamic status of resources and task execution for fault tolerance. This work applies one charge model for both edge resources and cloud resources, which is differ from the real world where the resource cost is the operational cost of edge servers while the rent cost of cloud resources for edge service providers.

Meng *et al.* [94] present a task scheduling method on hierarchical edge-clouds for minimizing the cost with deadline constraints, based on PSO improved by filtering infeasible initialized solution and updating parameters according to the iteration number. Meta-heuristic methods, e.g., GA, PSO, ACO, may achieve a better solution, but they are usually too cost or too slow convergence rate to be applicable for solving large scale problems, because of their pursuit of the global optimal solution. Thus, existed works exploiting meta-heuristics to address task scheduling problem evaluate their proposed methods on only small or middle scale systems, e.g., for only tens to thousands tasks, due to these methods' very poor performance for large scale systems.

c: MULTI-OBJECTIVE OPTIMIZATION

Xie *et al.* [96] design an improved particle swarm optimization (PSO) algorithm for scheduling a workflow on an edge-cloud to provide service providers a tradeoff between the **finish time** and the execution **cost** by optimizing their weighted sum. While for simplification, this work homogenizes the network resource by assuming all network bandwidths between two nodes are identical.

Gad-Elrab and Noaman [97] study on optimizing the **finish time** of the workflow, the **monetary cost** and the **energy consumption** of edge-cloud servers for task execution. They formulate the problem as a MILP problem minimizing the weight sum of their concerned three optimization objectives, and propose a heuristic method to address the problem. The proposed method first classifies tasks into several levels according to the task dependences, and then for each top level, iteratively assigns tasks to servers such that each server is assigned by one task and the optimization objective value is maximized. By employing their method, tasks are assigned to edge-cloud servers almost evenly, which may lead to load imbalance of servers due to the heterogeneities of servers and tasks.

2) PARTIAL OFFLOADING

a: RESPONSE TIME OPTIMIZATION

Li *et al.* [138] study on the task offloading problem optimizing the finish time for a workflow job on an edge-cloud. They first formulate the problem as a mixed integer non-linear programming model and linearize the model using big-M method [139] for transforming the model to a MILP model. Then, they design an method by integrating the Logic-Based Benders Decomposition (LBBD) principle [140] with the MILP to solve the model. For simplification, this work assumes an edge or cloud server performs only one task. And the no consideration of energy consumption may lead to a much higher consumed energy by reduce the finish time. As shown in their experiment results, the proposed method consumes hundreds of seconds in the scenario of 5 computing nodes performing 20 tasks, and thus is not available for large scale systems.

Nowadays, connected and autonomous vehicles have equipped with more and more computing, storage, and networking resources. To efficiently utilize these resources, Sun *et al.* [98] design a task offloading method based on genetic algorithm for offloading some tasks from an edge-cloud to multiple connected vehicles, for improving the average response time of all requests. They first exploit Hyper-Erlang distribution [141] to model the dwell time of each vehicle in a cell, and then formulate the task offloading as an optimization problem. The formulated problem is solved by improving genetic algorithm with the integer encoding and the restriction of unavailable offspring generation. For all random variables, e.g. task response time, this work uses their respective average value for decision, leading to no guarantee of requirement satisfaction for all requests.

Lakhan and Li [99] present a method, CATSA, for optimizing the manespan of mobile workflow applications. CATSA first partitions the deadline of each application into tasks' sub-deadlines based on the data amounts processed by tasks and tasks' dependency relationships. Then CATSA tries existed task order method, e.g., Earliest Due Date (EDD), Smallest Process First (SPF), and Smallest Slack Time First (SSTF), and selects the result with the best performance for task order. Thereafter, the method uses existed pair-wise decision methods, TOPSIS [142] and AHP [143], to decide the position for each task execution, and applies a local search method exploiting random searching for the edge/cloud. This work ignores the resource heterogeneity of user devices, the edge, and the cloud, such as the device energy is not concerned when making offloading decisions.

b: ENERGY OPTIMIZATION

Lakhan *et al.* [100], Lakhan and Xiaoping [101], and Abdullah and Xiaoping [102] propose a task offloading strategy for optimizing the energy consumed by the mobile device and the edge-cloud resources (i.e., VMs) for a user. The proposed strategy first employs the min-cut algorithm [144] to partition the workflow application into two kinds of tasks, i.e., local execution and remote execution, for minimize the mobile energy consumption. And to optimize energy consumption in the edge-cloud, the strategy iteratively assigns a task to the VM with minimal power consumption and completing the task before its deadline. This work assumes that power consumption of each VM and the wireless network transfer rate are both constants, which is inconsistent with the real world. In addition, this work doesn't concern the network energy consumption of the device and the VM provisioning in the edge-cloud, leading to a suboptimal solution of task execution in mobile-edge-cloud computing environments.

Zhu *et al.* [103] exploit the edge-cloud to improve the energy consumed by the user device for processing an application with deadline constraint. They formulate the energy consumption minimization problem a convex function ignoring the discreteness of task offloading decision, and employ the Newton's method to solve the problem. Their conducted experiments results show that edge-cloud has a better performance than edge or only device in energy consumption optimization for task processing, while they don't verify the advantage of their method by comparing with other (state-of-the-art) offloading methods.

c: MULTI-OBJECTIVE OPTIMIZATION

Sun *et al.* [104] design a task offloading method, ETCORA, to improve the **energy consumption** and the **finish time**. At first, ETCORA compares executing overheads on the user device (OL) and on the edge tier (OE) for a task, and assigns the task to the edge if OL is greater than OE. Otherwise, ETCORA compares OL with the executing overhead on the cloud, and assigns the task to the tier with a smaller overhead. The executing overhead is quantified by the weighted sum of the finish time and the energy consumed by the device and the data transmission. For tasks are offloaded to edge or cloud, they exploit Newton iteration and sub-gradient methods to solve the transmission power allocation problem. For simplification, the work ignores the difference between the edge tier and the cloud tier in task scheduling or resource provisioning, and doesn't concern the queue order of tasks. In addition, this work assumes a task can be executed immediately ignoring the waiting time in a computing node.

For optimizing the finish time and consumed energy of edge servers, Wu *et al.* [105] first define a bi-objective minimization problem, and then present a hybrid evolutionary task offloading method for edge-clouds to address it. The method greedily assigns tasks to the tier providing minimum weighted sum of finish time and energy, applies probability-based evolutionary algorithm to decide the task execution order respectively in the edge or cloud tier, and adopts the earliest finish time first scheme to conduct the task assignment in a tier. This work assumes that only one task can be processed by a device/server at a time, which leads to inefficient use of resources [145], due to there are multiple cores for a device/server in the real world [3].

All of above works concerning multiple objectives transform the problem as a single-objective optimization problem by weighting concerned objectives, which includes a much complicated step requiring users/providers to define the relative importances of objectives, resulting in a narrowed application scope. To address the problem, Maio and Kimovski [106] study on the Pareto-front in optimizing the response time, the reliability, and the cost for task execution in a edge-cloud environment, by formulating the problem as a multi-objective optimization problem which is NP-hard, and addressing the problem applying Non-dominated Sorting Genetic Algorithm II (NSGA-II) [146] with simulated binary crossover and polynomial mutation for providing solutions in the Pareto front. In this work, they assume each task instead of the workflow has its own requirements, such as the deadline, which simplifies the problem by without considering the division of some requirements into each task for the workflow, while is not consistent with the reality. In addition, this paper doesn't consider the waiting time of tasks in a fog/cloud node by assuming each task can be immediately executed after its predecessors are all finished, which is true when all kinds of resources are quite sufficiently, while it is hard to realize.

V. CHALLENGES AND DIRECTIONS

Until now, there are 71 published literatures studying on task offloading on edge-cloud computing with 134% CAGR, which results in a very promising research area. In this section, we discuss some of the challenges that need to be addressed in the future in order to enable optimal task offloading in edge-clouds to deliver high quality of services for extending the application range of edge-cloud computing.

A. JOINT OFFLOADING DECISION, ASSIGNMENT, AND SCHEDULING

For performing user tasks on an edge-cloud optimally, three optimization problems must be addressed jointly, (i) offloading decision deciding which tasks are offloaded for user devices, (ii) task assignment deciding which tier is each offloaded task performed on, which edge or cloud server is each offloaded task assigned to for its execution, or sometimes which devices are some tasks respectively assigned to in the case of connected user devices, and (iii) task scheduling deciding the execution order of all tasks for each device or server. Unfortunately, existing researches concern only one or two of these problems, as illustrated in previous sections, which leading to sub-optimal solutions for task offloading.

B. DECENTRALIZED OFFLOADING

As the scale of edge-cloud computing is becoming larger due to the number increase of connected devices [1], [2], the challenge of achieving optimal task offloading is more serious because the complexity of solving methods is increased with the system scale. One promising way to alleviate the challenge is to design a decentralized task offloading method by dividing the task offloading problem of a large edge-cloud system into several ones of multiple small subsystems, e.g., multiple edge systems and one cloud system. There is also a challenge for decentralized method design, how to design a decentralized task offloading method so that it has almost the same performance as centralized ones [147].

C. RESOURCE REQUIREMENT EVALUATION

In general, the performance requirements is in the form of QoS metrics, e.g., the response time. While, almost all related works use resource amount to express task requirements or employ the simple linear relation between task performance and resource amount, for simplifying the offloading problem in edge-cloud computing. Thus, existing works have to be employed with the ''real'' relationship between QoS values and resource amounts in edge-clouds, which is scarcely studied by researches. Therefore, it is necessary to establish the model mapping QoS requirements to various resources to address the problem, how many resources and which edge or cloud resources should be provided to satisfy the QoS requirements?

D. COST EVALUATION

More resources provide a better performance for tasks, while cost more. Thus, the service provider must provide resources carefully for offloaded tasks to optimize its cost with requirement satisfactions, which makes a cost evaluation of used resources necessary. While it is complex for building a cost model for edge-cloud resources, as (1) there is a much difference between cost models of operating private resources and renting public resources [17], [148], (2) it is impossible to accurately evaluate the operation cost for private resources due to its too many affecting factors including the electricity costs for power, hardware/software maintenance costs, and so on [149], (3) there are various price models for the public resource usage, e.g, on-demand, spot, and reserved, and the price of a type of resources may vary with time [150].

E. MOBILITY

Many smart devices are moving with time, e.g., unmanned aerial vehicles, smartphones, wearable devices, which affects the performance of user tasks offloaded on an edge, as the network performance usually becomes better along with the decrease of the distance between the user device and the edge. While, the device mobility is remarkably diverse in practice [151], which raises a new challenge for the task offloading in edge-cloud computing. It is also necessary to evaluate the concentration of users based on device mobility as it is helpful for designing the edge server deployment (see Section V-K) and balancing loads of edges.

F. DATA CACHING

Data transmission delay is a main impact factor on the performance of task execution when it is offloaded to an edge or a cloud. One of the most effective ways to address the problem is caching data the offloaded task needed in advance. While it is difficult to design a good data caching strategy with a high access hit ratio [152], because user data access patterns are hard to be predicted due to the high diversity and mobility of users in edge-cloud computing.

G. SECURITY

Most of time, security is one of the most important considerations for users, especially enterprise users, deciding whether offloading their tasks to others' resources to prevent their private data from being stolen or illegally attack [153]. While, no research has concerned the security when studying on the task offloading in edge-cloud computing. From the perspective of security, there are three kinds of tasks, (i) private tasks only can be processed by user owned resources, (ii) tasks with some private requirements, which can be offloaded to others' resources with the help of protection technologies usually with performance overheads, (iii) tasks without private requirement, which can be executed in any place. When conducting task offloading concerning, there is a tradeoff between the overheads of using protection technologies and of migrating executing tasks for idling some private resources. These make the task offloading more challenging in edge-cloud computing.

H. USAGE OF MULTI-CLOUD

In the cloud market, there are many public clouds can be used as the cloud tier for edge-cloud computing. The diversity of public cloud resources bring various benefits to public cloud users (i.e. service providers in edge-cloud computing) [154], [155], such as cost improvement by renting resources with the best cost-performance ratio every time and avoiding vendor lock-in, while it is not exploited by existed researches for task offloading. The usage of multiple public clouds increases the complexity as it introduces several resource heterogeneities [17], [154]. The service provider should carefully rent and allocate public resources to offloading tasks when employing multiple clouds for workflows, as the introduction of several public clouds may degrade the performance due to the low network performance between each two public clouds. Especially in the era of IoT big data, there are plenty of data analysis applications whose performance is largely limited by the network performance.

I. RESOURCE PROVISIONING DELAY

For an edge or a cloud, there is a delay for resource provisioning in practice, such as, a cloud consumes seconds or minutes for starting a VM instance [156]. While, no existed take the resource provisioning delay into account, which may result in violate some QoS requirements, e.g., the response time. Thus resource provisioning delay aware task offloading is essential to edge-cloud computing, and the evaluation of the provisioning delay is needed. There are many variables for accurately evaluating the provisioning delay must considered [17], [157], e.g., the resource heterogeneity, the software required by offloaded tasks, the network configuration, the time of the day, the edge or cloud location, etc., leading to a big challenge in the evaluation.

J. INFORMATION MONITORING

In edge-cloud computing, different task offloading methods require various system information, and the global system information is needed for making optimal task offloading decisions [158]. While, the edge-cloud system scales become more and more large, which increases the challenge in the system information monitoring. In large scale system, fine granularity information helps to improve the performance while remarkably increase the monitoring overheads, and thus, the tradeoff between the granularity and the overhead must be studied.

K. EDGE SERVER DEPLOYMENT

Existed researches on edge-cloud computing, even on edge computing, to our best of knowledge, only can be applied in the scenario that all edge servers have been deployed. While, before providing services by an edge, the provider must build the edge, deploy edge servers, which complements task offloading technologies, for resource efficiency. However, there is no available research on the edge server deployment addressing the problem, which server is selected and deployed on an edge center using the historical service data, to achieve a specific goal, e.g., optimal investment cost [122], [123] with user satisfactions when the edge is operates. It is not easy to address the problem, as the user requirements during the edge operation must be predicted accurately, the requirements must be considered at a time series instead of a time, edge servers in market are more and more diversity which provides more opportunities for providers improving their investment costs while increases the complexity of solving the problem, and so on.

VI. CONCLUSION

In this paper, we present a taxonomy on task offloading in edge-cloud computing for to classifying related works in the perspective of task type, offloading scheme, objective, and mobility. And then we investigate published related researches in detail. Even thought there are 71 published researches focusing on task offloading in edge-cloud computing, a lot of problems still be required research efforts before adopting edge-cloud computing to provide services. Thus we summarized several of these problems as well as research directions, which includes jointing offloading decision, task assignment, and task scheduling, decentralized task offloading, resource requirement evaluation for tasks, cost evaluation for resources, user device mobility, data caching, security guarantee, employing multi-cloud in the cloud tier, the concern of resource provisioning delay, information monitoring in large scale edge-clouds, as well as edge server deployment. We believe our survey work is helpful for industrial circles and academic interested in edge-clouds.

REFERENCES

[1] CISCO. (2020). *Cisco Annual Internet Report (2018–2023)*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html

[2] S. Sorrel. (2018). *The Internet of Things: Consumer Industrial & Public Services 2018–2023*. Sunnyvale, CA, USA. [Online]. Available: https://www.juniperresearch.com/press/press-releases/iot-connections-to-grow-140-to-hit-50-billion

[3] C.-J. Wu *et al.*, "Machine learning at Facebook: Understanding inference at the edge," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2019, pp. 331–344.

[4] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.

[5] Y. Ni, S. Zhou, Q. Wang, Y. Zhou, and H. Zhu, "Auction game based Phone-to-Phone electricity trading via wireless energy transfer," in *Proc. IEEE 19th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2019, pp. 1213–1219.

[6] A. Boukerche, S. Guan, and R. E. D. Grande, "Sustainable offloading in mobile cloud computing: Algorithmic design and implementation," *ACM Comput. Surv.*, vol. 52, no. 1, Feb. 2019.

[7] Q. Qi, J. Wang, Z. Ma, H. Sun, Y. Cao, L. Zhang, and J. Liao, "Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4192–4203, May 2019.

[8] S. Guo, D. Zeng, L. Gu, and J. Luo, "When green energy meets cloud radio access network: Joint optimization towards brown energy minimization," *Mobile Netw. Appl.*, vol. 24, no. 3, pp. 962–970, Jun. 2019.

[9] H.-N. Dai, R. C.-W. Wong, H. Wang, Z. Zheng, and A. V. Vasilakos, "Big data analytics for large-scale wireless networks: Challenges and opportunities," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 1–36, Sep. 2019.

[10] C.-H. Hong and B. Varghese, "Resource management in fog/edge computing: A survey on architectures, infrastructure, and algorithms," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 1–37 Sep. 2019.

[11] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao, "Task offloading with network function requirements in a mobile edge-cloud network," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2672–2685, Nov. 2019.

[12] J. Ren, D. Zhang, S. He, Y. Zhang, and T. Li, "A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet," *ACM Comput. Surv.*, vol. 52, no. 6, Oct. 2019.

[13] D. Marikyan, S. Papagiannidis, and E. Alamanos, "A systematic review of the smart home literature: A user perspective," *Technol. Forecasting Social Change*, vol. 138, pp. 139–154, Jan. 2019.

[14] A. Ferdowsi, U. Challita, and W. Saad, "Deep learning for reliable mobile edge analytics in intelligent transportation systems: An overview," *IEEE Veh. Technol. Mag.*, vol. 14, no. 1, pp. 62–70, Mar. 2019.

[15] A. Kamilaris and A. Pitsillides, "Mobile phone computing and the Internet of Things: A survey," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 885–898, Dec. 2016.

[16] Z. Hong, W. Chen, H. Huang, S. Guo, and Z. Zheng, "Multi-hop cooperative computation offloading for industrial IoT-edge-cloud computing environments," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2759–2774, Dec. 2019.

[17] B. Wang, C. Wang, Y. Song, J. Cao, X. Cui, and L. Zhang, "A survey and taxonomy on workload scheduling and resource provisioning in hybrid clouds," in *Cluster Computing*. Springer, Feb. 2020, pp. 1–26.

[18] A. Golande, P. Sorte, V. Suryawanshi, U. Yermalkar, and S. Satpute, "Smart hospital for heart disease prediction using IoT," *Int. J. Informat. Vis.*, vol. 3, nos. 2–2, pp. 198–202, 2019.

[19] M. U. Rehman, A. E. Andargoli, and H. Pousti, "Healthcare 4. 0: Trends, challenges and benefits," in *Proc. Australas. Conf. Inf. Syst.*, 2019, pp. 556–564.

[20] B. Akay and D. Karaboga, "A survey on the applications of artificial bee colony in signal, image, and video processing," *Signal, Image Video Process.*, vol. 9, no. 4, pp. 967–990, 2015.

[21] Y. Liang and X. Wang, "Developing a new perspective to study the health of survivors of Sichuan earthquakes in China: A study on the effect of post-earthquake rescue policies on survivors' health-related quality of life," *Health Res. Policy Syst.*, vol. 11, no. 1, p. 41, 2013.

[22] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "LSAR: Multi-UAV collaboration for search and rescue missions," *IEEE Access*, vol. 7, pp. 55817–55832, 2019.

[23] X. Zhang and S. Debroy, "Migration-driven resilient disaster response edge-cloud deployments," in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl. (NCA)*, Sep. 2019, pp. 1–8.

[24] A. Ahmad, S. Din, A. Paul, G. Jeon, M. Aloqaily, and M. Ahmad, "Real-time route planning and data dissemination for urban scenarios using the Internet of Things," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 50–55, Dec. 2019.

[25] X. Zhang, M. Zhou, H. Liu, and A. Hussain, "A cognitively inspired system architecture for the mengshi cognitive vehicle," *Cognit. Comput.*, vol. 12, no. 1, pp. 140–149, Jan. 2020.

[26] P. Cong, J. Zhou, L. Li, K. Cao, T. Wei, and K. Li, "A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud," *ACM Comput. Surv.*, vol. 53, no. 2, pp. 1–44, Apr. 2020.

[27] A. Heidari, M. A. J. Jamali, N. J. Navimipour, and S. Akbarpour, "Internet of Things offloading: Ongoing issues, opportunities, and future challenges," *Int. J. Commun. Syst.*, vol. 33, no. 14, p. e4474, Sep. 2020.

[28] M. Aazam, S. Zeadally, and K. A. Harras, "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities," *Future Gener. Comput. Syst.*, vol. 87, pp. 278–289, Oct. 2018.

[29] J. Wang, J. Pan, F. Esposito, P. Calyam, Z. Yang, and P. Mohapatra, "Edge cloud offloading algorithms: Issues, methods, and perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–23, Feb. 2019.

[30] A. Shakarami, A. Shahidinejad, and M. Ghobaei-Arani, "A review on the computation offloading approaches in mobile edge computing: A game-theoretic perspective," *Softw. Pract. Exper.*, vol. 50, no. 9, pp. 1719–1759, 2020.

[31] F. A. Salaht, F. Desprez, and A. Lebre, "An overview of service placement problem in fog and edge computing," *ACM Comput. Surveys*, vol. 53, no. 3, pp. 1–35, Jul. 2020.

[32] X. Yang and N. Rahmani, "Task scheduling mechanisms in fog computing: Review, trends, and perspectives," in *Kybernetes*. Bingley, U.K.: Emerald Publishing Limited, Mar. 2020, pp. 1–17, doi: 10.1108/K-10-2019-0666.

[33] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[34] T. L. Duc, R. G. Leiva, P. Casari, and P.-O. Östberg, "Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 1–39, Sep. 2019.

[35] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Archit.*, vol. 98, pp. 289–330, Sep. 2019.

[36] M. Rahimi, M. Songhorabadi, and M. H. Kashani, "Fog-based smart homes: A systematic review," *J. Netw. Comput. Appl.*, vol. 153, Mar. 2020, Art. no. 102531.

[37] H. Wu, S. Deng, W. Li, S. U. Khan, J. Yin, and A. Y. Zomaya, "Request dispatching for minimizing service response time in edge cloud systems," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2018, pp. 1–9.

[38] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 998–1010, Apr. 2018.

[39] H. Tan, Z. Han, X. Li, and F. C. M. Lau, "Online job dispatching and scheduling in edge-clouds," in *Proc. INFOCOM IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.

[40] Z. Han, H. Tan, X.-Y. Li, S. H.-C. Jiang, Y. Li, and F. C. M. Lau, "OnDisc: Online latency-sensitive job dispatching and scheduling in heterogeneous edge-clouds," *IEEE/ACM Trans. Netw.*, vol. 27, no. 6, pp. 2472–2485, Dec. 2019.

[41] H. K. Apat, B. S. Compt, K. Bhaisare, and P. Maiti, "An optimal task scheduling towards minimized cost and response time in fog computing infrastructure," in *Proc. Int. Conf. Inf. Technol. (ICIT)*, Dec. 2019, pp. 160–165.

[42] M. A. Benblidia, B. Brik, L. Merghem-Boulahia, and M. Esseghir, "Ranking fog nodes for tasks scheduling in fog-cloud environments: A fuzzy logic approach," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2019, pp. 1451–1457.

[43] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid fog–cloud computing," *Future Gener. Comput. Syst.*, vol. 111, pp. 539–551, Oct. 2020.

[44] F. Murtaza, A. Akhunzada, S. U. Islam, J. Boudjadar, and R. Buyya, "QoS-aware service provisioning in fog computing," *J. Netw. Comput. Appl.*, vol. 165, Sep. 2020, Art. no. 102674.

[45] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.

[46] Z. Ning, P. Dong, X. Kong, and F. Xia, "A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4804–4814, Jun. 2019.

[47] L. Li, M. Guo, L. Ma, H. Mao, and Q. Guan, "Online workload allocation via fog-fog-cloud cooperation to reduce IoT task service delay," *Sensors*, vol. 19, no. 18, p. 3830, Sep. 2019.

[48] X. Fan, H. Zheng, R. Jiang, and J. Zhang, "Optimal design of hierarchical cloud-fog&edge computing networks with caching," *Sensors*, vol. 20, no. 6, p. 1582, Mar. 2020.

[49] G. Gao, M. Xiao, J. Wu, H. Huang, S. Wang, and G. Chen, "Auction-based VM allocation for deadline-sensitive tasks in distributed edge cloud," *IEEE Trans. Services Comput.*, early access, Mar. 4, 2019, doi: 10.1109/TSC.2019.2902549.

[50] Y. Chen, N. Zhang, Y. Zhang, and X. Chen, "Dynamic computation offloading in edge computing for Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4242–4251, Jun. 2019.

[51] L. Chen, K. Guo, G. Fan, C. Wang, and S. Song, "Resource constrained profit optimization method for task scheduling in edge cloud," *IEEE Access*, vol. 8, pp. 118638–118652, 2020.

[52] H. Yuan and M. Zhou, "Profit-maximized collaborative computation offloading and resource allocation in distributed cloud and edge computing systems," *IEEE Trans. Autom. Sci. Eng.*, early access, Jul. 14, 2020, doi: 10.1109/TASE.2020.3000946.

[53] X. Lin, H. Zhang, H. Ji, and V. C. M. Leung, "Joint computation and communication resource allocation in mobile-edge cloud computing networks," in *Proc. IEEE Int. Conf. Netw. Infrastruct. Digit. Content (IC-NIDC)*, Sep. 2016, pp. 166–171.

[54] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1594–1608, Apr. 2018.

[55] R. Duan, J. Wang, C. Jiang, Y. Ren, and L. Hanzo, "The transmit-energy vs computation-delay trade-off in gateway-selection for heterogenous cloud aided multi-UAV systems," *IEEE Trans. Commun.*, vol. 67, no. 4, pp. 3026–3039, Apr. 2019.

[56] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, "Profit-aware application placement for integrated fog–cloud computing environments," *J. Parallel Distrib. Comput.*, vol. 135, pp. 177–190, Jan. 2020.

[57] C. Li, C. Wang, and Y. Luo, "An efficient scheduling optimization strategy for improving consistency maintenance in edge cloud environment," *J. Supercomput.*, vol. 76, pp. 6941–6968, Sep. 2020, doi: 10.1007/s11227-019-03133-9.

[58] C. Sun, X. Wen, Z. Lu, W. Jing, and M. Zorzi, "Eco-friendly powering and delay-aware task scheduling in geo-distributed edge-cloud system: A two-timescale framework," *IEEE Access*, vol. 8, pp. 96468–96486, 2020.

[59] M. Adhikari, S. N. Srirama, and T. Amgoth, "Application offloading strategy for hierarchical fog environment through swarm optimization," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4317–4328, May 2020.

[60] S. Ma, S. Guo, K. Wang, W. Jia, and M. Guo, "A cyclic game for joint cooperation and competition of edge resource allocation," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 503–513.

[61] Y. Miao, G. Wu, M. Li, A. Ghoneim, M. Al-Rakhami, and M. S. Hossain, "Intelligent task prediction and computation offloading based on mobile-edge cloud computing," *Future Gener. Comput. Syst.*, vol. 102, pp. 925–931, Jan. 2020.

[62] C. Kai, H. Zhou, Y. Yi, and W. Huang, "Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability," *IEEE Trans. Cognit. Commun. Netw.*, early access, Aug. 20, 2020, doi: 10.1109/TCCN.2020.3018159.

[63] K. Guo, M. Yang, Y. Zhang, and J. Cao, "Joint computation offloading and bandwidth assignment in cloud-assisted edge computing," *IEEE Trans. Cloud Comput.*, early access, Oct. 30, 2019, doi: 10.1109/TCC.2019.2950395.

[64] J. Meng, H. Tan, C. Xu, W. Cao, L. Liu, and B. Li, "Dedas: Online task dispatching and scheduling with bandwidth constraint in edge computing," in *Proc. INFOCOM IEEE Conf. Comput. Commun.*, May 2019, pp. 2287–2295.

[65] J. Meng, H. Tan, X.-Y. Li, Z. Han, and B. Li, "Online deadline-aware task dispatching and scheduling in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1270–1286, Jun. 2020.

[66] K. Cui, W. Sun, and W. Sun, "Joint computation offloading and resource management for USVs cluster of fog-cloud computing architecture," in *Proc. IEEE Int. Conf. Smart Internet Things (SmartIoT)*, Aug. 2019, pp. 92–99.

[67] K. Cui, B. Lin, W. Sun, and W. Sun, "Learning-based task offloading for marine fog-cloud computing networks of USV cluster," *Electronics*, vol. 8, no. 11, p. 1287, Nov. 2019.

[68] I. Sarkar, S. Kumar, and M. Mukherjee, "An optimized task placement in computational offloading for fog-cloud computing networks," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Dec. 2019, pp. 1–5.

[69] T. Ouyang, R. Li, X. Chen, Z. Zhou, and X. Tang, "Adaptive user-managed service placement for mobile edge computing: An online learning approach," in *Proc. INFOCOM IEEE Conf. Comput. Commun.*, Apr. 2019, pp. 1468–1476.

[70] S. Cheng, Z. Chen, J. Li, and H. Gao, "Task assignment algorithms in data shared mobile edge computing systems," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 997–1006.

[71] Q. Xia, Z. Lou, W. Xu, and Z. Xu, "Near-optimal and learning-driven task offloading in a 5G multi-cell mobile edge cloud," *Comput. Netw.*, vol. 176, Jul. 2020, Art. no. 107276.

[72] C. Zhang, H. Du, Q. Ye, C. Liu, and H. Yuan, "DMRA: A decentralized resource allocation scheme for multi-SP mobile edge computing," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 390–398.

[73] O. Chabbouh, S. B. Rejeb, Z. Choukair, and N. Agoulmine, "A strategy for joint service offloading and scheduling in heterogeneous cloud radio access networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 196, Nov. 2017.

[74] Z. Wang, S. Zheng, Q. Ge, and K. Li, "Online offloading scheduling and resource allocation algorithms for vehicular edge computing system," *IEEE Access*, vol. 8, pp. 52428–52442, 2020.

[75] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[76] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, "Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks," *IEEE Access*, vol. 8, pp. 137052–137062, 2020.

[77] A. Alshahrani, I. A. Elgendy, A. Muthanna, A. M. Alghamdi, and A. Alshamrani, "Efficient multi-player computation offloading for VR edge-cloud computing systems," *Appl. Sci.*, vol. 10, no. 16, p. 5515, Aug. 2020.

[78] X. Cheng, F. Lyu, W. Quan, C. Zhou, H. He, W. Shi, and X. Shen, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[79] C. Sun, L. Hui, X. Li, J. We, Q. Xiongl, X. Wang, and V. C. M. Leun, "Task offloading for end-edge-cloud orchestrated computing in mobile networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, May 2020, pp. 1–6.

[80] J. Long, Y. Luo, X. Zhu, E. Luo, and M. Huang, "Computation offloading through mobile vehicles in IoT-edge-cloud network," *EURASIP J. Wireless Commun. Netw.*, pp. 1–21, Sep. 2020, doi: 10.21203/rs.3.rs-41747/v2.

[81] T. Ti Nguyen, V. Nguyen Ha, L. Bao Le, and R. Schober, "Joint data compression and computation offloading in hierarchical fog-cloud systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 293–309, Jan. 2020.

[82] D. Wang, Z. Liu, X. Wang, and Y. Lan, "Mobility-aware task offloading and migration schemes in fog computing networks," *IEEE Access*, vol. 7, pp. 43356–43368, 2019.

[83] M. Alkhalaileh, R. N. Calheiros, Q. V. Nguyen, and B. Javadi, "Data-intensive application scheduling on mobile edge cloud computing," *J. Netw. Comput. Appl.*, vol. 167, Oct. 2020, Art. no. 102735.

[84] Q. Wang, S. Guo, J. Liu, and Y. Yang, "Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing," *Sustain. Comput. Informat. Syst.*, vol. 21, pp. 154–164, Mar. 2019.

[85] H. Wu, Z. Zhang, C. Guan, K. Wolter, and M. Xu, "Collaborate edge and cloud computing with distributed deep learning for smart city Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8099–8110, Sep. 2020.

[86] T. Yang, H. Feng, S. Gao, Z. Jiang, M. Qin, N. Cheng, and L. Bai, "Two-stage offloading optimization for energy–latency tradeoff with mobile edge computing in maritime Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5954–5963, Jul. 2020.

[87] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for IoT systems: A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, Aug. 2018.

[88] H. Shah-Mansouri and V. W. Wong, *Computation Offloading Game for Fog-Cloud Scenario*. Hoboken, NJ, USA: Wiley, 2020, ch. 3, pp. 61–82.

[89] M. Adhikari and H. Gianey, "Energy efficient offloading strategy in fog-cloud environment for IoT applications," *Internet Things*, vol. 6, Jan. 2019, Art. no. 100053.

[90] K. Peng, H. Huang, S. Wan, and V. C. M. Leung, "End-edge-cloud collaborative computation offloading for multiple mobile users in heterogeneous edge-server environment," in *Wireless Networks*. Springer, Jun. 2020, pp. 1–12.

[91] M. Du, Y. Wang, K. Ye, and C. Xu, "Algorithmics of cost-driven computation offloading in the edge-cloud environment," *IEEE Trans. Comput.*, vol. 69, no. 10, pp. 1519–1532, Oct. 2020.

[92] L. Liu, H. Tan, S. H.-C. Jiang, Z. Han, X.-Y. Li, and H. Huang, "Dependent task placement and scheduling with function configuration in edge computing," in *Proc. Int. Symp. Qual. Service (IWQoS)*, Jun. 2019, pp. 1–10.

[93] D. Haja, B. Vass, and L. Toka, "Towards making big data applications network-aware in edge-cloud systems," in *Proc. IEEE 8th Int. Conf. Cloud Netw. (CloudNet)*, Nov. 2019, pp. 1–6.

[94] S. Meng, W. Huang, X. Xu, Q. Li, W. Dou, and B. Liu, "A self-adaptive PSO-based dynamic scheduling method on hierarchical cloud computing," in *Cloud Computing, Smart Grid and Innovative Frontiers in Telecommunications* (CloudComp 2019, SmartGift 2019). Cham, Switzerland: Springer, 2019, pp. 89–100, doi: 10.1007/978-3-030-48513-9_7.

[95] S. Meng, Q. Li, T. Wu, W. Huang, J. Zhang, and W. Li, "A fault-tolerant dynamic scheduling method on hierarchical mobile edge cloud computing," *Comput. Intell.*, vol. 35, pp. 577–598, May 2019.

[96] Y. Xie, Y. Zhu, Y. Wang, Y. Cheng, R. Xu, A. S. Sani, D. Yuan, and Y. Yang, "A novel directional and non-local-convergent particle swarm optimization based workflow scheduling in cloud–edge environment," *Future Gener. Comput. Syst.*, vol. 97, pp. 351–378, Aug. 2019.

[97] A. A. A. Gad-Elrab and A. Y. Noaman, "Fuzzy clustering-based task allocation approach using bipartite graph in cloud-fog environment," in *Proc. 16th EAI Int. Conf. Mobile Ubiquitous Syst. Comput., Netw. Services (MobiQuitous)*, New York, NY, USA, Nov. 2019, p. 454–463.

[98] F. Sun, F. Hou, N. Cheng, M. Wang, H. Zhou, L. Gui, and X. Shen, "Cooperative task scheduling for computation offloading in vehicular cloud," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11049–11061, Nov. 2018.

[99] A. Lakhan and X. Li, "Content aware task scheduling framework for mobile workflow applications in heterogeneous Mobile-Edge-Cloud paradigms: CATSA framework," in *Proc. IEEE Int. Conf. Parallel Distrib. Process. Appl., Big Data Cloud Comput., Sustain. Comput. Commun., Social Comput. Netw. (ISPA/BDCloud/SocialCom/SustainCom)*, Dec. 2019, pp. 242–249.

[100] A. Lakhan, D. K. Sajnani, M. Tahir, M. Aamir, and R. Lodhi, "Delay sensitive application partitioning and task scheduling in mobile edge cloud prototyping," in *Proc. Int. Conf. 5G Ubiquitous Connectivity*. Cham, Switzerland: Springer, 2018, pp. 59–80.

[101] A. Lakhan and L. Xiaoping, "Energy aware dynamic workflow application partitioning and task scheduling in heterogeneous mobile cloud network," in *Proc. Int. Conf. Cloud Comput., Big Data Blockchain (ICCBB)*, Nov. 2018, pp. 1–8.

[102] A. Lakhan and X. Li, "Dynamic partitioning and task scheduling for complex workflow healthcare application in mobile edge cloud architecture," in *Proc. IEEE 4th Int. Conf. Comput. Commun. (ICCC)*, 2018, pp. 2532–2536.

[103] Y. Zhu, Z. Wang, Z. Han, N. Li, and S. Yang, "Multithread optimal offloading strategy based on cloud and edge collaboration," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, May 2020, pp. 1–5.

[104] H. Sun, H. Yu, G. Fan, and L. Chen, "Energy and time efficient task offloading and resource allocation on the generic IoT-fog-cloud architecture," *Peer-Peer Netw. Appl.*, vol. 13, no. 2, pp. 548–563, Mar. 2020.

[105] C. Wu, W. Li, L. Wang, and A. Zomaya, "Hybrid evolutionary scheduling for energy-efficient fog-enhanced Internet of Things," *IEEE Trans. Cloud Comput.*, early access, Dec. 24, 2018, doi: 10.1109/TCC.2018.2889482.

[106] V. De Maio and D. Kimovski, "Multi-objective scheduling of extreme data scientific workflows in fog," *Future Gener. Comput. Syst.*, vol. 106, pp. 171–184, May 2020.

[107] J. Choi, S. Kim, T. Adufu, S. Hwang, and Y. Kim, "A job dispatch optimization method on cluster and cloud for large-scale high-throughput computing service," in *Proc. Int. Conf. Cloud Autonomic Comput.*, Sep. 2015, pp. 283–290.

[108] A. M. Mielke, S. M. Brennan, M. C. Smith, D. C. Torney, A. B. Maccabe, and J. KarlinM, "Independent sensor networks," *IEEE Instrum. Meas. Mag.*, vol. 8, no. 2, pp. 33–37, Jun. 2005.

[109] C. S. Kim, N. S. Cho, and K. R. Park, "Deep residual network-based recognition of finger wrinkles using smartphone camera," *IEEE Access*, vol. 7, pp. 71270–71285, 2019.

[110] N. Muslim and S. Islam, "Face recognition in the edge cloud," in *Proc. Int. Conf. Imag., Signal Process. Commun.*, New York, NY, USA, 2017, pp. 5–9.

[111] Y. Song, Y. Peng, and L. Zhang, "Dynamic tasks assignment for face recognition in edge computing," in *Proc. 28th Wireless Opt. Commun. Conf. (WOCC)*, May 2019, pp. 1–5.

[112] B. Chen, Z. Yang, S. Huang, X. Du, Z. Cui, J. Bhimani, X. Xie, and N. Mi, "Cyber-physical system enabled nearby traffic flow modelling for autonomous vehicles," in *Proc. IEEE 36th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Dec. 2017, pp. 1–6.

[113] M. Z. A. Bhuiyan, J. Wu, G. Wang, T. Wang, and M. M. Hassan, "E-sampling: Event-sensitive autonomous adaptive sensing and low-cost monitoring in networked sensing systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 1, pp. 1–29, Mar. 2017.

[114] K. Papadakis-Vlachopapadopoulos, R. S. González, I. Dimolitsas, D. Dechouniotis, A. J. Ferrer, and S. Papavassiliou, "Collaborative SLA and reputation-based trust management in cloud federations," *Future Gener. Comput. Syst.*, vol. 100, pp. 498–512, Nov. 2019.

[115] M. Cinque, S. Russo, C. Esposito, K.-K.-R. Choo, F. Free-Nelson, and C. A. Kamhoua, "Cloud reliability: Possible sources of security and legal issues?" *IEEE Cloud Comput.*, vol. 5, no. 3, pp. 31–38, May 2018.

[116] D. Chemodanov, P. Calyam, S. Valluripally, H. Trinh, J. Patman, and K. Palaniappan, "On qoe-oriented cloud service orchestration for application providers," *IEEE Trans. Services Comput.*, early access, Aug. 23, 2018, doi: 10.1109/TSC.2018.2866851.

[117] J. Li, W. Liang, M. Huang, and X. Jia, "Providing reliability-aware virtualized network function services for mobile edge computing," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 732–741.

[118] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.

[119] Á. R. Castaño, F. Real, P. Ramón-Soria, J. Capitán, V. Vega, B. C. Arrue, A. Torres-González, and A. Ollero, "Al-robotics team: A cooperative multi-unmanned aerial vehicle approach for the mohamed bin zayed international robotic challenge," *J. Field Robot.*, vol. 36, no. 1, pp. 104–124, Jan. 2019.

[120] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *J. Netw. Comput. Appl.*, vol. 89, pp. 96–108, Jul. 2017.

[121] Q. Wang, S. Zhang, Y. Kanemasa, and C. Pu, "Mitigating tail response time of n-tier applications: The impact of asynchronous invocations," *ACM Trans. Internet Technol.*, vol. 19, no. 3, pp. 1–25, Jul. 2019.

[122] B. Wang, Y. Song, Y. Sun, and J. Liu, "Analysis model for server consolidation of virtualized heterogeneous data centers providing Internet services," *Cluster Comput.*, vol. 22, no. 3, pp. 911–928, Sep. 2019.

[123] B. Wang, Y. Song, X. Cui, and J. Cao, "Mathematical programming for server consolidation in cloud data centers," in *Proc. 4th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2017, pp. 678–683.

[124] J. Lin, "Multiple-objective problems: Pareto-optimal solutions by method of proper equality constraints," *IEEE Trans. Autom. Control*, vol. AC-21, no. 5, pp. 641–650, Oct. 1976.

[125] C. Funai, C. Tapparello, and W. Heinzelman, "Computational offloading for energy constrained devices in multi-hop cooperative networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 60–73, Jan. 2020.

[126] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *J. Netw. Comput. Appl.*, vol. 143, pp. 1–33, Oct. 2019.

[127] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[128] ł. Bekmezci, O. K. Sahingoz, and Ş. Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Netw.*, vol. 11, no. 3, pp. 1254–1270, May 2013.

[129] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[130] B. Wang, Y. Song, Y. Sun, and J. Liu, "Managing deadline-constrained bag-of-tasks jobs on hybrid clouds with closest deadline first scheduling," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 7, pp. 2952–2971, 2016.

[131] W. Kongsiriwattana and P. Gardner-Stephen, "Smart-phone battery-life short-fall in disaster response: Quantifying the gap," in *Proc. IEEE Global Humanitarian Technol. Conf. (GHTC)*, Oct. 2016, pp. 220–225.

[132] R. S. Sutton and F. Bach, *Reinforcement Learning—An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[133] A. Mirhosseini, B. L. West, G. W. Blake, and T. F. Wenisch, "Q-zilla: A scheduling framework and core microarchitecture for tail-tolerant microservices," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2020, pp. 207–219.

[134] Q. Wu, H. Ge, H. Liu, Q. Fan, Z. Li, and Z. Wang, "A task offloading scheme in vehicular fog and cloud computing system," *IEEE Access*, vol. 8, pp. 1173–1184, 2020.

[135] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," Swiss Federal Inst. Technol. (ETH), Zürich, Switzerland, TIK-Rep. 103, Sep. 2001.

[136] N. Kumar, S. Zeadally, N. Chilamkurti, and A. Vinel, "Performance analysis of Bayesian coalition game-based energy-aware virtual machine migration in vehicular mobile cloud," *IEEE Netw.*, vol. 29, no. 2, pp. 62–69, Mar. 2015.

[137] S. Abrishami, M. Naghibzadeh, and D. H. J. Epema, "Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds," *Future Gener. Comput. Syst.*, vol. 29, no. 1, pp. 158–169, Jan. 2013.

[138] S. Li, W. Chen, Y. Chen, C. Chen, and Z. Zheng, "Makespan-minimized computation offloading for smart toys in edge-cloud computing," *Electron. Commerce Res. Appl.*, vol. 37, Sep. 2019, Art. no. 100884.

[139] (2020). *Big M Method*. [Online]. Available: https://en.wikipedia.org/wiki/Big_M_method

[140] J. Hooker and G. Ottosson, "Logic-based benders decompositio," *Math. Program.*, vol. 96, no. 1, pp. 33–60, 2003.

[141] Y. Fang, "Hyper-erlang distribution model and its application in wireless mobile networks," *Wireless Netw.ork*, vol. 7, no. 3, pp. 211–219, May 2001.

[142] D. Liang and Z. Xu, "The new extension of TOPSIS method for multiple criteria decision making with hesitant pythagorean fuzzy sets," *Appl. Soft Comput.*, vol. 60, pp. 167–179, Nov. 2017.

[143] T. L. Saaty, "Decision making with the analytic hierarchy process," *Int. J. Services Sci.*, vol. 1, no. 1, pp. 83–98, 2008.

[144] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, "A context sensitive offloading scheme for mobile cloud computing service," in *Proc. IEEE 8th Int. Conf. Cloud Comput. (CLOUD)*, Los Alamitos, CA, USA, Jun. 2015, pp. 869–876.

[145] B. Wang, Y. Song, J. Cao, X. Cui, and L. Zhang, "Improving task scheduling with parallelism awareness in heterogeneous computational environments," *Future Gener. Comput. Syst.*, vol. 94, pp. 419–429, May 2019.

[146] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Cham, Switzerland: Springer, 2000, pp. 849–858.

[147] S. Jošilo and G. Dán, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 207–220, Jan. 2019.

[148] P. Dreher, D. Nair, E. Sills, and M. Vouk, "Cost analysis comparing HPC public versus private cloud computing," in *Proc. Int. Conf. Cloud Comput. Services Sci.* Cham, Switzerland: Springer, 2016, pp. 294–316, doi: 10.1007/978-3-319-62594-2_15.

[149] S. Saha, J. Sarkar, A. Dwivedi, N. Dwivedi, A. M. Narasimhamurthy, and R. Roy, "A novel revenue optimization model to address the operation and maintenance cost of a data center," *J. Cloud Comput.*, vol. 5, no. 1, pp. 266–278, Dec. 2016.

[150] N. C. Luong, P. Wang, D. Niyato, Y. Wen, and Z. Han, "Resource management in cloud networking using economic analysis and pricing models: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 954–1001, 2nd Quart., 2017.

[151] M. D. Soltani, A. A. Purwita, Z. Zeng, H. Haas, and M. Safari, "Modeling the random orientation of mobile devices: Measurement, analysis and LiFi use case," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2157–2172, Mar. 2019.

[152] E. E. Ugwuanyi, S. Ghosh, M. Iqbal, T. Dagiuklas, S. Mumtaz, and A. Al-Dulaimi, "Co-operative and hybrid replacement caching for multi-access mobile edge computing," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2019, pp. 394–399.

[153] J. Pan and J. McElhannon, "Future edge cloud and edge computing for Internet of Things applications," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 439–449, Feb. 2018.

[154] C. Qu, R. N. Calheiros, and R. Buyya, "Auto-scaling Web applications in clouds: A taxonomy and survey," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–33, 2018.

[155] A. C. Zhou, B. He, X. Cheng, and C. T. Lau, "A declarative optimization engine for resource provisioning of scientific workflows in geo-distributed clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 647–661, Mar. 2017.

[156] K. Razavi, G. Kolk, and T. Kielmann, "Prebaked μVMs: Scalable, instant VM startup for IAAS clouds," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2015, pp. 245–255.

[157] T. Nguyen and A. Lebre, "Virtual machine boot time model," in *Proc. 25th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, 2017, pp. 430–437.

[158] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 84–99, Feb. 2019.

BO WANG received the B.S. degree in computer science from Northeast Forest University (NEFU), Harbin, China, in 2010, and the Ph.D. degree in computer science from Xi'an Jiaotong University (XJTU), Xi'an, China, in 2017. He was a Guest Student with the State Key Laboratory of Computer Architecture, Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), from 2012 to 2016. He is currently a Lecturer with the Software Engineering College, Zhengzhou University of Light Industry (ZZULI). He has published more than ten research articles in these areas. His research interests include distributed systems, cloud computing, edge computing, resource management, and task scheduling. He served in various academic journals and conferences.

CHANGHAI WANG was born in Liaocheng, Shandong, China, in 1987. He received the M.S. degree in computer science and technology from Jiangsu University, in 2012, and the Ph.D. degree from Nankai University, in 2016. He has been with the Zhengzhou University of Light Industry (ZZULI) since 2017. He has published nine articles related to activity recognition. His research interests include activity recognition, wearable computing, and mobile computing. He received the Best Paper Award from the Tenth IEEE Asia-Pacific Services Computing Conference and the Excellent Paper Award from *Journal of Communication* in 2016.

WANWEI HUANG received the Ph.D. degree in computer science from Information Engineering University, Zhengzhou, China, in 2017. He is currently an Associate Professor with the Software Engineering College, Zhengzhou University of Light Industry (ZZULI). He has published more than ten publications in these areas. His research interests include wireless networks, artificial intelligence, and so on. He served in various academic journals and conferences.

YING SONG received the Ph.D. degree in computer engineering from the Institute of Computing Technology (ICT), Chinese Academy of Sciences. She is currently an Associate Professor with the Computer School, Beijing Information Science and Technology University. Her work has covered topics, such as performance modeling, resource management, cloud computing, and big data computing platform. She has been authored or coauthored more than 30 publications in these areas since 2007. Her main research interests include computer architecture, parallel and distributed computing, and virtualization technology. She served in various academic conferences.

XIAOYUN QIN received the Ph.D. degree in analytical chemistry from the Institute of Chemistry, Chinese Academy of Sciences, Beijing, China, in 2017. She is currently a Lecturer with the Department of Material and Chemical Engineering, Zhengzhou University of Light Industry (ZZULI). She has published more than ten publications. Her research interests include computational chemistry, biocomputing, and so on. She served in various academic journals and conferences.