Computers, Materials & Continua
DOI:10.32604/cmc.2021.018145
Article

Tech Science Press

Investigating and Modelling of Task Offloading Latency in Edge-Cloud Environment

Jaber Almutairi[1] and Mohammad Aldossary[2,*]

[1]Department of Computer Science, College of Computer Science and Engineering, Taibah University, Al-Madinah, Saudi Arabia
[2]Department of Computer Science, College of Arts and Science, Prince Sattam bin Abdulaziz University, Al-Kharj, Saudi Arabia
[*]Corresponding Author: Mohammad Aldossary. Email: mm.aldossary@psau.edu.sa
Received: 22 February 2021; Accepted: 24 March 2021

Abstract: Recently, the number of Internet of Things (IoT) devices connected to the Internet has increased dramatically as well as the data produced by these devices. This would require offloading IoT tasks to release heavy computation and storage to the resource-rich nodes such as Edge Computing and Cloud Computing. However, different service architecture and offloading strategies have a different impact on the service time performance of IoT applications. Therefore, this paper presents an Edge-Cloud system architecture that supports scheduling offloading tasks of IoT applications in order to minimize the enormous amount of transmitting data in the network. Also, it introduces the offloading latency models to investigate the delay of different offloading scenarios/schemes and explores the effect of computational and communication demand on each one. A series of experiments conducted on an EdgeCloudSim show that different offloading decisions within the Edge-Cloud system can lead to various service times due to the computational resources and communications types. Finally, this paper presents a comprehensive review of the current state-of-the-art research on task offloading issues in the Edge-Cloud environment.

Keywords: Edge-cloud computing; resource management; latency models; scheduling; task offloading; internet of things

1 Introduction

Internet of Things (IoT) technology has quickly evolved in recent years, where the number of devices that are connected to the internet (IoT) has increased massively. Some studies predict that in the upcoming three years, more than 50 billion devices will be connected to the internet [1,2], which will produce a new set of applications such as Autonomous Vehicles, Augmented Reality (AR), online video games and Smart CCTV.

Thus, Edge Computing has been proposed to deal with the huge change in the area of the distributed system. For enhancing customer experience and accelerating job execution, IoT

task offloading enables mobile end devices to release heavy computation and storage to the resource-rich nodes in collaborative Edges or Clouds. Nevertheless, resource management at the Edge-Cloud environment is challenging because it deals with several complex factors (e.g., different characteristics of IoT applications and heterogeneity of resources).

Additionally, how different service architecture and offloading strategies quantitatively impact the end-to-end service time performance of IoT applications is still far from known particularly given a dynamic and unpredictable assortment of interconnected virtual and physical devices. Also, latency-sensitive applications have various changing characteristics, such as computational demand and communication demand. Consequently, the latency depends on the scheduling policy of applications offloading tasks as well as where the jobs will be placed. Therefore, Edge-Cloud resource management should consider these characteristics in order to meet the requirements of latency-sensitive applications.

In this regard, it is essential to conduct in-depth research to investigate the latency within the Edge-Cloud system, the impact of computation and communication demands and resource heterogeneity to provide a better understanding of the problem and facilitate the development of an approach that aims to improve both applications' QoS and Edge-Cloud system performance. Also, efficient resource management will play an essential role in providing real-time or near real-time use for IoT applications.

The aim of this research is to investigate and model the delay for latency-sensitive applications within the Edge-Cloud environment as well as provide a detailed analysis of the main factors of service latency, considering both applications characteristics and the Edge-Cloud resources. The proposed approach is used to minimize the overall service time of latency-sensitive applications and enhance resource utilization in the Edge-Cloud system. This paper's main contributions are summarized as follows:

- An Edge-Cloud system architecture that includes the required components to support scheduling offloading tasks of IoT applications.
- An Edge-Cloud latency models that show the impact of different tasks' offloading scenarios/schemes for time-sensitive applications in terms of end-to-end service times.
- An evaluation of the proposed offloading latency models that consider computation and communication as key parameters with respect to offloading to the local edge node, other edge nodes or the cloud.

The remainder of this paper is organized as follows: Section 2 presents the system architecture that supports scheduling offloading tasks of IoT applications, followed by the descriptions of the required components and their interactions within the proposed architecture. Section 3 latency-sensitive applications. Section 4 presents the latency models, followed by experiments and evaluation in Section 5. A thorough discussion of the related work is presented in Section 6. Finally, Section 7 concludes this paper and discusses the future work

2 Proposed System Architecture

As illustrated in Fig. 1, the Edge-Cloud system from bottom to the top consists of three layers/tiers: IoT devices (end-user devices), multiple Edge Computing nodes and the Cloud (service provider). The IoT level is composed of a group of connected devices (e.g., smartphones, self-driving cars, smart CCTV); these devices have different applications where each application has several tasks (e.g., smart CCTV [3] application consists of movement dedication and face recognition). These services can be deployed and executed in different computing resources (connected

Edge node, other Edge nodes or Cloud), where the infrastructure manager and service providers have to decide where to run these services.

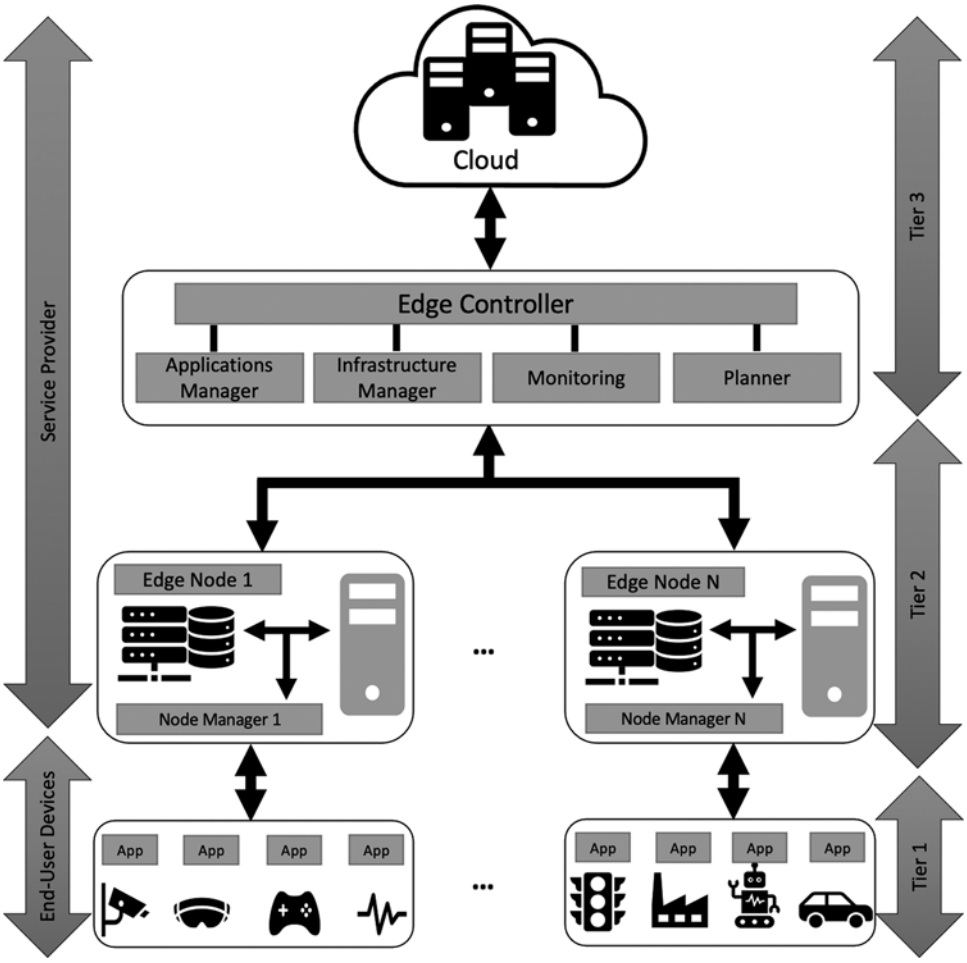

Figure 1: An overview of edge-cloud system

In this proposed system, at the Edge level, each Edge Computing node is a micro datacenter with a virtualized environment. It has been placed close to the connected IoT devices at the base station or Wi-Fi access point. These edge nodes have been distributed geographically and could be owned by the same Cloud provider or other brokers [4]. Note that, it has limited computational resources compared to the resources in the cloud. Each edge node has a node manager that can manage computational resources and application services that run on. All the edge nodes have connected to the *Edge Controller*.

The offloading tasks can be achieved when the IoT devices decide to process the task remotely in Edge-Cloud environments. Applications running on IoT devices can send their offloadable tasks to be processed by the Edge-Cloud system through their associated Edge node. We assume that each IoT application is deployed in a Virtual Machine (VM) in the edge node and the cloud. IoT devices offload tasks which belong to a predefined set of applications, these tasks are varied in term of the computational requirement (task length) and communication demand (amount of

transferred data). It is assumed that tasks are already offloaded from the IoT devices, and each task is independent; thus, the dependency between the tasks is not addressed in this paper. The locations of IoT devices are important for the service time performance because it is assumed that each location is covered by a dedicated wireless access point with an Edge node and the IoT devices connect to the related WLAN when they move to the covered location.

The associated Edge can process IoT tasks and also can be processed collaboratively with other edge nodes or the cloud, based on Edge orchestrator decisions. For example, if an IoT application is located in an edge node faraway from its connected edge, its data traffic has to be routed to it via a longer path in the Edge-Cloud system. At the cloud level, a massive amount of resources that enable IoT applications' tasks to be processed and stored.

The proposed architecture is just a possible implementation of other architectures in the literature such as [2,5,6]. The main difference in the proposed architecture is the introduced layer between the edge nodes and the cloud. This layer responsible for managing and assign offloading tasks to the edge nodes. More details about the required components and their interactions within the proposed architecture are follow.

2.1 Edge Controller

Edge Controller (EC) is designed similar to [7–9], some studies called *Edge Orchestrator*, which is a centralized component responsible for planning, deploying and managing application services in the Edge-Cloud system. EC communicates with other components in the architecture to know the status of resources in the system (e.g., available and used), the number of IoT devices, their applications' tasks and where IoT tasks have been allocated (e.g., Edge or Cloud). EC consists of the following components: *Application Manager*, *Infrastructure Manager*, *Monitoring* and *Planner*. The location of the *Edge Controller* can be deployed in any layer between Edge and Cloud. For example, in [10], EC act as an independent entity in the edge layer that manages all the edge nodes in its control. It is also responsible for scheduling the offloading tasks in order to satisfy applications' users and Edge-Cloud System requirements. The EC is synchronizing its data with the centralized Cloud because if there is any failure, other edge nodes can take EC responsibility from the cloud [11,12].

2.2 Application Manager

The application manager is responsible for managing applications running in the Edge-Cloud system. This includes requirements of application tasks, such as the amount of data to be transferred, the amount of computational requirement (e.g., required CPU) and the latency constraints. Besides, the number of application users for each edge node.

2.3 Infrastructure Manager

The role of the infrastructure manager is to be in charge of the physical resources in the entire Edge-Cloud system. For instance, processors, networking and the connected IoT devices for all edge nodes. As mentioned earlier, Edge-Cloud is a virtualized environment; thus, this component responsible for the VMs as well. In the context of this research, this component provides the EC with the utilization level of the VMs.

2.4 Monitoring

The main responsibility of this component is to monitoring application tasks (e.g., computational delay and communication delay) and computational resources (e.g., CPU utilization) during

the execution of applications' tasks in the Edge-Cloud system. Furthermore, detecting the tasks' failures due to network issues or the shortage of computational resources.

2.5 Planner

The main role of this component is to propose the scheduling policy of the offloading tasks in the Edge-Cloud system and the location where they will be placed (e.g., local edge, other edges or the cloud). In the context of this research, the proposed approach for offloading tasks works on this component and passes its results to EC for execution.

3 Latency-sensitive Applications

Latency-sensitive applications can be defined as applications that have high sensitivity to any delays accrue in communication or computation during the interaction with the Edge-Cloud system.

For instance, the IoT device sends data to the point that processing is complete at the edge node or the cloud in the back end of the network, and the subsequent communications are produced by the network in response to receive the results. There are many examples of latency-sensitive applications and the acceptable service time varies depending on the application type, which affected by the amount of transferred data and the required computation volume [13]. For example, self-driving cars consist of several services, the work presented in [14] classified these services in categories based on their latency-sensitivity, quality constraints and workload profile (required communication and computation). First, critical applications, which must be processed in the car's computational resources, for instance, autonomous driving and road safety applications. Second, high-priority applications, which can be offloaded but with minimum latency, such as image aided navigation, parking navigation system and traffic control. Third, low-priority applications, which can be offloaded and not vital as high-priority applications (e.g., infotainment, multimedia, and speech processing). Tab. 1 presents more examples of latency-sensitive applications in different technology sectors [13].

Table 1: Latency-sensitive applications

Industry	Applications
Industrial automation	Industrial Control
	Robot Control
	Process Control
Healthcare Industry	Remote Diagnosis
	Emergency Response
	Remote Surgery
Entertainment Industry	Immersive Entertainment
	Online Gaming
Transport Industry	Driver Assistance Applications
	Autonomous Driving
	Traffic Management
Manufacturing Industry	Motion Control
	Remote Control
	AR and VR Applications

4 Latency Models

Investigating and modelling the various offloading decisions for IoT tasks that can increase the Quality of Service (QoS) has attracted the attention of many researchers in the field. With the increasing number of IoT devices, the amount of produced data, the need for an autonomous system that requires a real-time interaction as well as the lake of support from the central Cloud due to network issues; service time has been considered as one of the most important factors to be handled in Edge Computing [15–17].

One of the main characteristics of Edge Computing is to reduce the latency level. Additionally, it has been proved through literature that using Edge Computing will enhance application performance in terms of overall service time comparing to the traditional Cloud system [18–20]. However, different offloading decisions within the Edge-Cloud system can lead to various service time due to the computational resources and communications types. The current real-world applications measure the latency between the telecommunication service provider and the cloud services [21]. Also, a few existing works compare the latency between offloading to the edge or the cloud. Yet, the latency between multiple edge nodes that work collectively to process the offloading tasks are not considered. Consequently, investigating the latency of the Edge-Cloud system is an essential step towards developing an effective scheduling policy due to the following reasons. Firstly, task allocation in the Edge-Cloud system is not only two choices (e.g., either at IoT device or in the cloud), but could be on any edge nodes. Moreover, edge nodes connected in a loosely coupled way on heterogeneous wireless networks (i.e., WLAN, MAN and WAN), making the process of resource management and the offloading decision more sophisticated. Secondly, given that task processing is allocated among multiple edge nodes working collectively and the cloud, it is challenging to make an optimal offloading decision.

Therefore, this paper introduces the latency models to investigate the delay of different offloading scenarios/schemes. Also, it explores the effect of computational and communication for each offloading scenario/scheme, which are: (1) offloading to the local edge, (2) offloading to the local edge with the cloud and (3) offloading to the local edge, other available edge nodes and the cloud. The list of the latency models' parameters and their notations are shown in Tab. 2.

Table 2: Summary of notations

Symbol	Meaning
t_{te_up}	Transmission Time between the IoT to the Edge node for uploading
t_{te_down}	Transmission Time between the IoT to the Edge node for Downloading
t_{ce}	Computation time in the Edge node
t_{tc_up}	Transmission Time between the Edge node to the Cloud for uploading
t_{tc_down}	Transmission Time between the Edge node to the Cloud for Downloading
t_{cc}	Computation time in the Cloud
t_{teo_up}	Transmission Time between the Edge node to other nearby Edge nodes for uploading
t_{teo_down}	Transmission Time between the Edge node to other nearby Edge nodes for Downloading
t_{ceo}	Computation time in the other nearby Edge node

4.1 Latency to Local Edge

This is known as a one-level offloading system, which is basically offloading to "Cloudlet" or "Local Edge". It aims to provide a micro-data center that supports IoT devices within a specific area such as a coffee shop, mall center and airport [22,23]. Thus, IoT devices can offload their tasks to be processed on the edge or cloud, as an example. This offloading scenario/scheme provides ultra-low latency due to the avoidance of network backhaul delays.

To clarify, IoT devices send their offloading tasks through the wireless network, and then the tasks will be processed by the edge node and finally send the results to IoT devices, as shown in Fig. 2. The end-to-end service time composed of two delays, network delay and computational delay. The network delay consists of the time of sending the data to the edge and the time to receive the output from the edge to the IoT device. The computation time is the time from arriving the task to the edge node until the processing has completed. Therefore, the end-to-end service time latency is the sum of communication delay and computational delay [24], which can be calculated as follows:

$$L_{Local\_edge} = t_{te\_up} + t_{ce} + t_{te\_down} \tag{1}$$

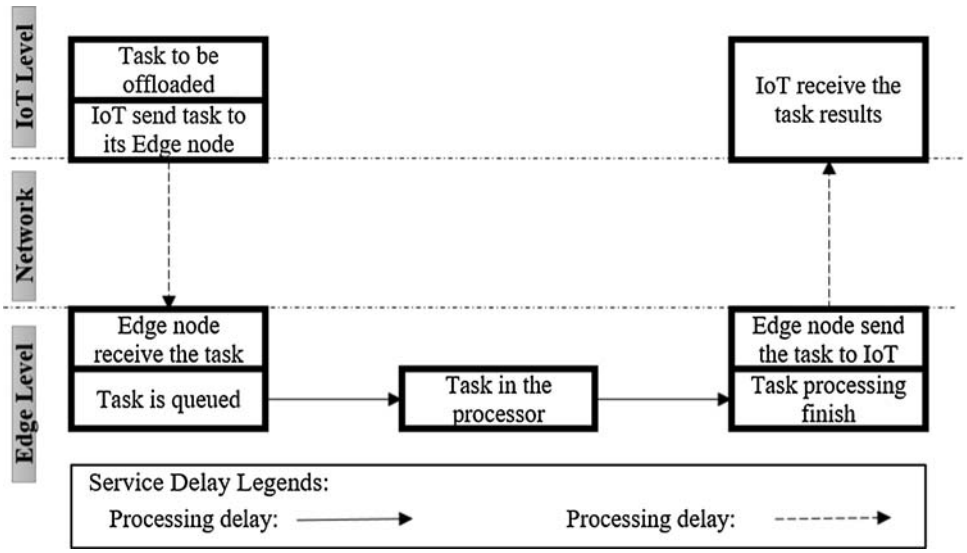

Figure 2: Latency to local edge

4.2 Latency to Local Edge with the Cloud

In this offloading scenario/scheme, rather than relying on only one Edge node, the IoT tasks can be processed collaboratively between the connected Edge node and the cloud servers. This will combine the benefits of both Cloud and Edge Computing, where the cloud has a massive amount of computation resources, and the edge has lower communication time [25]. In this scenario/scheme, the edge can do part of the processing such as pre-processing, and the rest of the tasks will be processed in the cloud.

As illustrated in Fig. 3, IoT sends the computation tasks to the connected edge and then part of these tasks forwarded to the cloud. Once the cloud finishes the computation, it will send the

result to the edge, and the edge will send it to the IoT devices. This scenario/scheme consists of communication time (e.g., the time between the IoT device to the edge node and the time between edge nodes to the cloud) and computation time (e.g., processing time in the edge and processing time in the cloud). Thus, the end-to-end service time can be calculated as follows:

$$L_{L\_C} = t_{te\_up} + t_{ce} + t_{tc\_up} + t_{cc} + t_{tc\_down} + t_{te\_down} \tag{2}$$

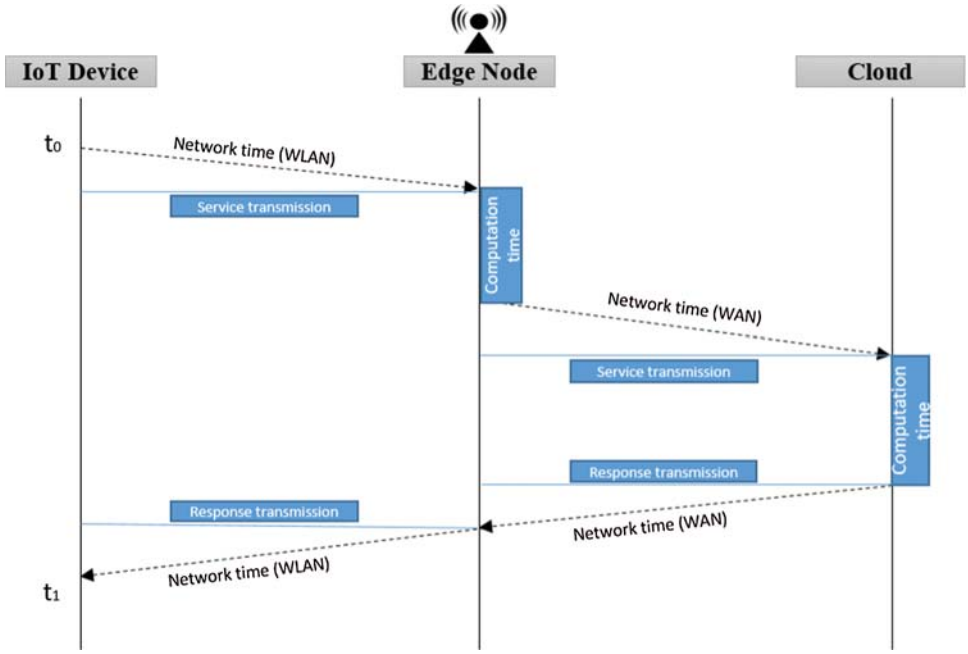

Figure 3: Latency to local edge with the cloud

4.3 Latency to Multiple Edge Nodes with the Cloud

This is known as a three-level offloading scenario/scheme [26], that aims to utilize more resources at the edge layer and support the IoT devices in order to reduce the overall service time. It adds another level by considering other available computation resources in the edge layer. Basically, it distributes IoT tasks over three levels: *connected edge*, *other available edge nodes* and *the cloud*. The *edge controller* (edge orchestrator) controllers all edge servers by Wireless Local Area Network (WLAN) or Metropolitan Area Network (MAN), which have low latency compared to Wild Area Network (WAN).

As illustrated in Fig. 4, IoT sends the computation tasks to the connected edge and then part of these tasks transferred to other available resources in the edge level through the edge controller and the rest to the cloud. This will help to decrease the dependency of cloud processing as well as increase the utilization of computing resources at the edge [20]. This scenario/scheme consists of communication time (e.g., the time between the IoT device to the edge node, the time between edge node to other collaborative edge node and the time between edge nodes to the cloud) and computation time (e.g., processing time in the edge, processing time in other collaborative edge

node and processing time in the cloud). Thus, the end-to-end service time can be calculated as follows:

$$L_{three-off} = [t_{te_{up}} + t_{ce} + t_{teo_{up}} + t_{ceo} + t_{tc_{up}} + t_{cc} + t_{tc_{down}} + t_{teo_{down}} + t_{te_{down}}] \quad (3)$$

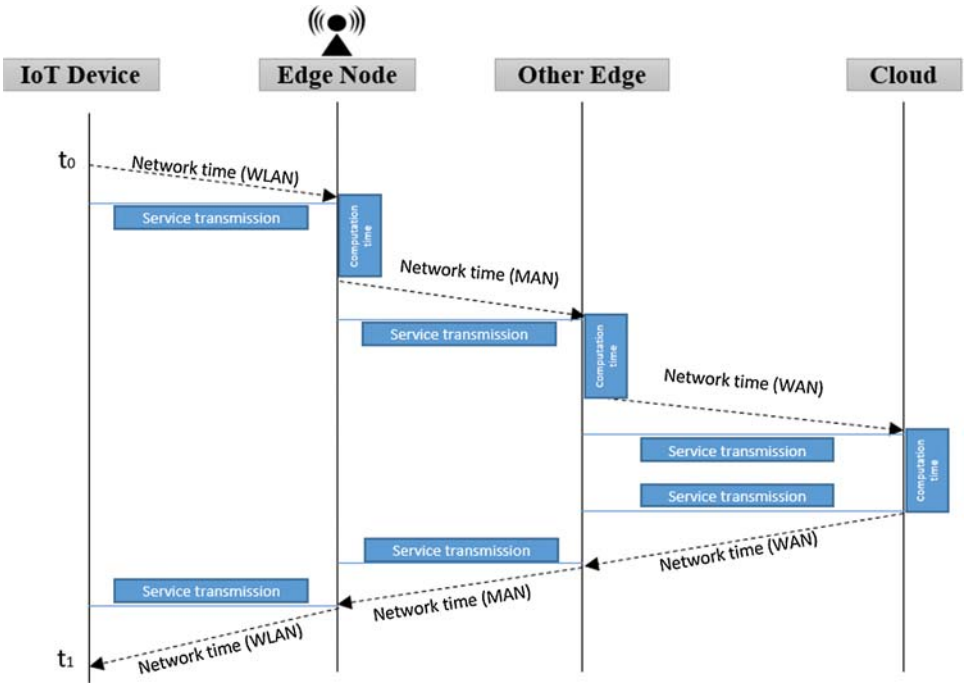

Figure 4: Latency to multiple edge nodes with the cloud

5 Experiments and Evaluation

5.1 Design of Experiments

A number of simulation experiments have been conducted on EdgeCloudSim in order to obtain the results of different offloading scenarios/schemes, and their influence on overall service time. EdgeCloudSim provides sufficient models to represent some specific situations. For example, the service time model is designed to represent the several kinds of delay taking place in the WLAN, MAN, and WAN as well as mobile devices and even the delay of processing in the CPUs of VMs. Thus, experiments in this paper are practically conducted within this simulation to investigate and evaluate the performance of IoT applications over the three different offloading scenarios/schemes. All the experiments are repeated five times, and the statistical analysis is performed to consider the mean values of the results to avoid any anomalies from the simulation results. We assume that we have three edge nodes connected to the cloud. Each edge node has two servers, and each of them has four VMs with a similar configuration. The number of edge nodes does not matter in the context of this research as long it more than two, because one of our aims to investigate the latency between two edge nodes. The cloud contains an unlimited number of computational resources. We got inspiration from other related works such as [24,27] to design the experiments and its parameters (e.g., number of IoT devices, edge nodes and the amount of transferred data for each offloading tasks).

Tab. 3 represents the key parameters of the simulation environment. The warm-up period is used to allow the system to evolve to a condition more representative of a steady state before getting the simulation output. A number of iterations are used to avoid any anomalies from the simulation results.

Table 3: Key parameters of the simulation environment

Key parameters	Values
Simulation Time	30 min
Warm-up Period	3 min
Number of Iterations	5
Number of IoT Devices	100–1000
Number of Edge Nodes	3
Number of VM per Edge Server	8
Number of VM in the Cloud	not limited
Average Data Size for Upload/Download (KB)	500/500

5.2 Results and Discussion

The conducted experiments show the results of three different offloading scenarios/schemes, offloading to the local edge (i.e., cloudlet), offloading to the local edge with the cloud and offloading to multiple edge nodes with the cloud. The aim of these experiments is to investigate and evaluate the processing delays, network delays and end-to-end service delays of the three offloading scenarios/schemes. This will increase our understanding of the offloading decision in the Edge-Cloud system in order to design an efficient Edge-Cloud resource management.

Fig. 5 presented the overall service time of the three offloading scenarios/schemes. Offloading to one-level is has the lowest service time. This result is consistent with work in [24,28], this is because of the avoidance of major latency between the end device and the cloud. Two-offloading levels have lower service time performance than the three-offloading. This shows the overall service time will never be truly minimized unless the network time is considered in the offloading process. However, these results may be somewhat limited by the number of IoT devices and the system load.

Also, the results presented in Fig. 6 have shown a significant difference in network time between one level offloading and the others (two-level and three-levels). As mentioned earlier, this is due to the avoidance of WAN and MAN delays. Two offloading levels lower than the three offloading levels because of the further communications between edge nodes.

In terms of processing time, offloading to the edge and the cloud has the lowest service time comparing to others, as depicted in Fig. 7. The reason is that the local edge has limited computational resources; thus, if the number of IoT devices increases, the processing delays will increase due to limited capacity. On the other hand, offloading to multiple edge nodes with the cloud has the highest processing time. However, the result of processing time was not very encouraging; thus, more investigation will be held on the impact of the parameter of processing time (computational demand), as a part of future work.

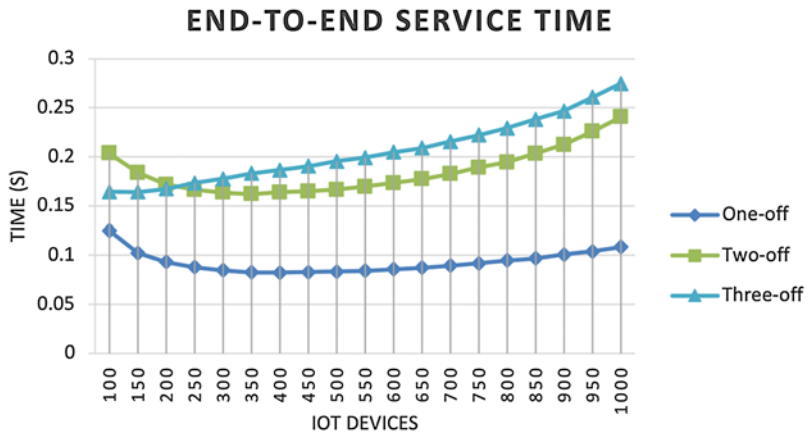

Figure 5: End-to-end service time for three offloading scenarios/schemes

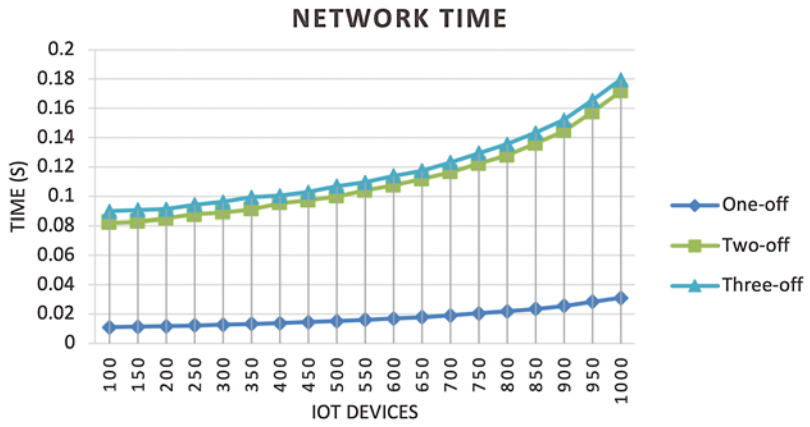

Figure 6: Network time for three offloading scenarios/schemes

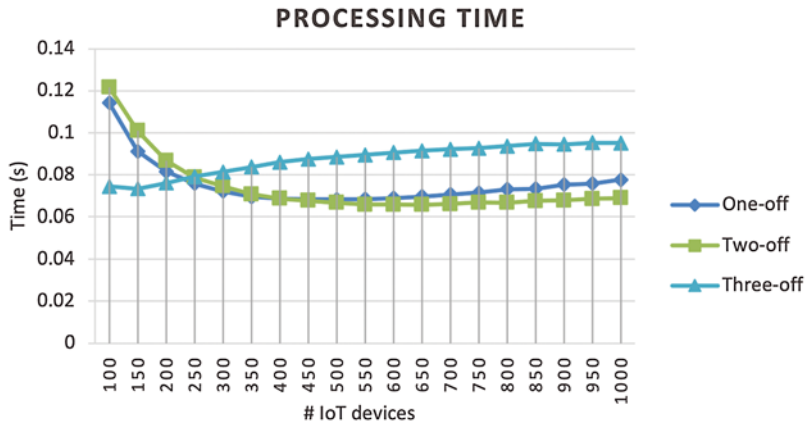

Figure 7: Processing time for three offloading scenarios/schemes

6 Offloading Approaches in Edge-cloud Environment: state-of-the-art

Computation offloading is not a new paradigm; it is widely used in the area of Cloud Computing. Offloading transfers computations from the resource-limited mobile device to resource-rich

Cloud nodes in order to improve the execution performance of mobile applications and the holistic power efficiency. Users devices are evenly located at the edge of the network. They could offload computation to Edge and Cloud nodes via WLAN network or 4G/5G networks. Generally, if a single edge node is insufficient to deal with the surging workloads, other edge nodes or cloud nodes are ready for assisting such an application. This is a practical solution to support IoT applications by transferring heavy computation tasks to powerful servers in the Edge-Cloud system. Also, it is used to overcome the limitations of IoT devices in terms of computation power (e.g., CPU and memory) and insufficient battery. It is one of the most important enabling techniques of IoT, because it allows performing a sophisticated computational more than their capacity [29]. Thus, the decisions of computational offloading in the context of IoT can be summarized as follows:

- First, whether the IoT device decides to offload a computational task or not. In this case, several factors could be considered, such as the required computational power and transferred data.
- Second, if there is a need for offloading, does partial offloading or full offloading. *Partial offloading* refers to the part of the tasks that will be processed locally at the IoT device and other parts in the Edge-Cloud servers. Also, factors such as task dependency and task priority can be considered in this case. *Full offloading* means, the whole application will be processed remotely in the Edge-Cloud servers [30].

In terms of the objectives of computation offloading in the context of Edge Computing, it can be classified into two categories; objectives that focus on *application characteristics* and objectives that focus on *Edge-Cloud resources*. Several studies [7,31–33] had aim to minimize service latency, energy consumption and mandatory cost, as well as maximize total revenue and resource utilization. In fact, scheduling offloading tasks is a challenging issue in the Edge-Cloud Computing paradigm, since it considers several trade-offs form application requirements (e.g., reduce latency) and system requirements (e.g., maximize resource utilization). Thus, developing an efficient resource management technique, that meets the requirements of both application and system, is an active area of research.

In the following subsections, some of the studies conducted on task offloading in Edge-Cloud environments to reduce the latency and maximize resource utilization, are reviewed and discussed, as illustrated in Fig. 8.

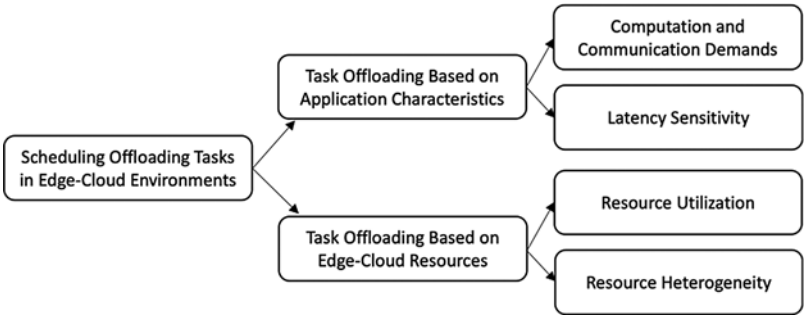

Figure 8: Classification of the topics reviewed

6.1 Task Offloading Based on Application Characteristics

As stated in [34–36], scheduling offloaded tasks that focused on application characteristics is considered significantly important, especially, with the increase of IoT applications. Therefore, this subsection presents the conducted studies on task offloading, which mainly focuses on application characteristics including (computation and communication demands, and latency-sensitivity).

- Computation and Communication Demands

There are many ongoing research projects focusing on the task computation and communication demands of IoT applications. For example, Wang et al. [37] proposed an online approximation algorithm that mainly objective to balance the load and minimizing resource utilization in order to enhance application performance. This work considers the attributes of computational and communications for homogenous resources, without considering the service latency. Rodrigues et al. [24], presented a hyper method for minimizing service latency and reduce power consumption. This method aims to reduce the communication and computational delays by migrating the VM to an unloaded server. The authors investigate the impact of tasks computational and communication demands. They evaluate their approach under realistic conditions by mathematical modelling. However, their method does not consider the application delay constraints as well as the offloading to the cloud. Deng et al. [16], proposed an approximate approach for minimizing network latency and power consumption by allocating workload between Fog and Cloud. However, their approach does not optimize the trade-off between all mentioned objectives (e.g., computational delay and resource utilization).

Zeng et al. [38] designed a strategy for task offloading that aims to minimize the completion time. In their work, both computation time and transmission time are considered. Also, the authors investigate the impact of other factors such as I/O interrupt requests and storage activities. However, delay-constraints applications and resource heterogeneity are not considered in their work. Fan et al. [39] designed an allocation scheme that aims to minimize service latency for IoT applications, by taking into account both computation and communication delays. Furthermore, the authors investigate the impact of the overloaded VM on processing time, and they evaluated their work with different types of applications. However, the proposed method does not show the effectiveness of the heterogeneity of the VMs in terms of service time and also does not consider the latency-sensitive application.

- Latency Sensitivity

In terms of application latency-sensitivity, a number of studies are conducted in order to enhance the overall service time in the Edge-Cloud environment. For instance, Mahmud et al. [34] proposed a latency-aware policy that aims to meet the required deadlines for offloading tasks. This approach considering task dependency as well as the computational and communication requirements. Also, the resource utilization at the edge level is considered. However, the issue of resource heterogeneity dose not addressed in their work. Azizi et al. [40] designed a priority-based service placement policy that prioritizes tasks with deadlines; thus, the nearest deadlines scheduled first. Further, their work considers both computational and communication demands. However, their evaluation does not address the issue when the system has multi IoT devices with different resource utilization. Sonmez et al. [27] presented an approach for task offloading that targets latency-sensitive applications. This approach is based on fuzzy logic, which focused on delay as a key factor along with computational and communication demands. Nevertheless, in this approach resource heterogeneity is not considered.

6.2 Task Offloading Based on Edge-cloud Resources

This subsection presents the literature of offloading tasks and mainly focused on resource utilization and resource heterogeneity as main objectives.

- Resource Utilization

Scheduling offloading tasks based on resource utilization or resource heterogeneity has received considerable critical attention from many researchers. For example, Nan et al. [41] developed an online optimization algorithm for offloading tasks that aim to minimize the cost of renting Cloud services by utilizing resources at the edge using the Lyapunov technique. Further, their algorithm guarantees the availability of edge resources and ensures processing the task within the required time. Yet, this algorithm does not consider the impact of computational and communication demands for latency-sensitive applications. Xu et al. [6] proposed a model for resource allocation that aims to maximize resource utilization and reduce task execution latency, as well as, reducing the dependability on the cloud in order to minimize Cloud cost. However, this work only considers resource utilization and does not refer to resource heterogeneity. Besides, application uploading and downloading data are not addressed in their work, which plays a significant role in overall service time. Li and Wang [42] introduced a placement approach that aims to reduce edge nodes' energy consumption and maximize resource utilization. They evaluated the proposed algorithm through applied numerical analysis based on the Shanghai Telecom dataset. However, their work does not provide any information regarding the application characteristics (e.g., computation, communication and delay-sensitivity).

- Resource Heterogeneity

Resource heterogeneity for the offloading decision plays a critical role to enhance the performance of service time in the Edge-Cloud environment. Thus, a number of studies have investigated the impact of resource heterogeneity on service time. For instance, Scoca et al. [43] proposed a scour-based algorithm for scheduling offloading tasks that considers both computation and communication parameters. Furthermore, their algorithm considers a heterogeneous VMs and sorts heavy tasks to be allocated to the most powerful VM. However, their algorithm does not consider server utilization as key parameters, which could affect the performance of service time. Roy et al. [44] proposed a strategy for task allocation that allocates different application tasks to an appropriate edge server by considering resource heterogeneity. This approach aims to reduce the execution latency as well as balancing the load between edge nodes. Yet, task communication time is not considered in this approach. Taneja et al. [45] proposed a resource-aware placement for IoT offloading tasks. Their approach ranks the resources at the edge with their capabilities and then assigns tasks to a suitable server based on the task's requirements (e.g., CPU, RAM and Bandwidth). However, this method focused on improving the performance of application service time, but without explicitly considering application latency-sensitivity.

Ultimately, with the dynamicity of IoT workload demands, Edge-Cloud service providers aimed to find a balance between utilizing Edge-Cloud resources efficiently and satisfying QoS objectives of IoT applications. Consequently, designing a new task offloading mechanism can contribute to enhancing resource utilization and supporting the latency-sensitive application requirements in the Edge-Cloud environment.

Section 6.1 has reviewed the related work on offloading tasks that are mainly focusing on application parameters such as computation demands, communication demands and latency-sensitivity in Edge-Cloud environments. The presented work in [24,38,39] considered these

application parameters in order to minimize the service time. However, these works lack to consider the impact of resource parameters such as server utilization and VMs heterogeneity.

Table 4: Comparison of the works addressing task offloading decisions

Criteria	Objective	Application Characteristics Considerations			Edge-Cloud Resources Considerations			Evaluation Method
By		Compute	Network	Delay	Resource Utilization	Resource Type	Number of Devices	
[37]	Minimize Resource Utilization	Considered	Considered	Not Considered	Considered	Homogeneous	-	Simulation
[24]	Minimize Service Latency	Considered	Considered	Not Considered	Considered	Homogeneous	Single	Mathematical Modelling
[16]	Minimize Network Latency	Not Considered	Considered	Not Considered	Considered	Homogeneous	-	Simulation
[43]	Minimize Service Latency	Considered	Considered	Not Considered	Not Considered	Heterogenous	Multi	Simulation
[41]	Minimize Cost	Not Considered	Not Considered	Considered	Considered	Homogeneous	Multi	Simulation
[44]	Minimize Execution Time	Considered	Not Considered	Not Considered	Not Considered	Heterogenous	Single	Direct Experiment
[38]	Minimize Completion Time	Considered	Considered	Not Considered	Not Considered	Homogeneous	Multi	Simulation
[34], [27]	Minimize Service Latency	Considered	Considered	Considered	Considered	Homogeneous	Multi	Simulation
[40]	Minimize Service Latency	Considered	Considered	Considered	Not Considered	Homogeneous	Single	Simulation
[45]	Minimize Service Time & Maximize Resource Utilization	Considered	Considered	Not Considered	Considered	Heterogenous	Multi	Simulation
[39]	Minimize Service Time	Considered	Considered	Not Considered	Not Considered	Homogeneous	Multi	Simulation
[6]	Maximize Resource Utilization	Considered	Not Considered	Considered	Considered	Homogeneous	-	Simulation
[42]	Minimize Energy Consumption & Maximize Resource Utilization	Not Considered	Not Considered	Not Considered	Considered	Homogeneous	Multi	Data-Driven Analysis

As discussed earlier in **Section 6.2**, the work presented in [37,43,45] considered the resource utilization and resource heterogeneity as key parameters to schedule offloading tasks in the

Edge-Cloud environment. While, some related works such as [16,42,44] have considered application requirements (e.g., computation or communication) but, without explicitly considering the latency-sensitivity of IoT applications. Hence, there is still a need for an efficient resource management technique that takes into account the application characteristics (competition, communication and latency), as well as resource parameters (resource utilization and heterogeneity) in order to meet the requirements of IoT applications (service time and task offloading) and utilize Edge-Cloud resources efficiently. Tab. 4 provides a comparison summary of the closely related work on offloading tasks that consider both application and resource parameters in the Edge-Cloud environment.

7 Conclusion and Future Work

This paper has presented an Edge-Cloud system architecture that supports scheduling offloading tasks of IoT application, as well as the explanation of the required components and their interactions within the system architecture. Furthermore, it has presented the offloading latency models that consider computation and communication as key parameters with respect to offloading to the local edge node, other edge nodes or the cloud. This paper has concluded by discussing a number of simulation experiments conducted on EdgeCloudSim to investigate and evaluate the latency models of three different offloading scenarios/schemes, followed by a comprehensive review of the current state-of-the-art research on task offloading issues in the Edge-Cloud environment.

As a part of future work, we intend to extend our approach by adopting the fuzzy logic algorithm which considers application characteristics (e.g., CPU demand, network demand and delay sensitivity) as well as resource utilization and resource heterogeneity in order to minimize the overall time of latency-sensitive applications.

Funding Statement: In addition, the authors would like to thank the Deanship of Scientific Research, Prince Sattam bin Abdulaziz University, Al-Kharj, Saudi Arabia, for supporting this work.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Zaslavsky, C. Perera and D. Georgakopoulos, "Sensing as a service and big data," *in Proc. of the Int. Conf. on Advances in Cloud Computing*, Bangalore, India, pp. 21–29, 2012.

[2] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *Computer Communication Review*, vol. 44, no. 5, pp. 27–32, 2014.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta *et al.* "On multi-access edge computing: A survey of the emerging 5G network edge architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[4] N. Choi, D. Kim, S. Lee and Y. Yi, "Fog operating system for user-oriented IoT services: Challenges and research directions," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 44–51, 2017.

[5] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. of the 1st ACM Mobile Cloud Computing Workshop*, Helsinki, Finland, pp. 13–15, 2012.

[6] J. Xu, B. Palanisamy, H. Ludwig and Q. Wang, "Zenith:Utility-aware resource allocation for edge computing," in *Proc. of the IEEE 1st Int. Conf. on Edge Computing*, Honolulu, HI, USA, pp. 47–54, 2017.

[7] H. Flores, X. Su, V. Kostakos, A. Y. Ding, P. Nurmi *et al.* "Large-scale offloading in the internet of things," in *Proc. of the IEEE Int. Conf. on Pervasive Computing and Communications Workshops*, Kona, HI, USA, pp. 479–484, 2017.

[8] D. Santoro, D. Zozin, D. Pizzolli, F. De Pellegrini and S. Cretti, "Foggy: A platform for workload orchestration in a fog computing environment," in *Proc. of the Int. Conf. on Cloud Computing Technology and Science*, Hong Kong, China, pp. 231–234, 2017.

[9] A. Hegyi, H. Flinck, I. Ketyko, P. Kuure, C. Nemes *et al.* "Application orchestration in mobile edge cloud: placing of IoT applications to the edge," *in Proc. of the IEEE 1st Int. Workshops on Foundations and Applications of Self-Systems*, Augsburg, Germany, pp. 230–235, 2016.

[10] K. Imagane, K. Kanai, J. Katto, T. Tsuda and H. Nakazato, "Performance evaluations of multimedia service function chaining in edge clouds," in *Proc. of the 15th IEEE Annual Consumer Communications and Networking Conf.*, Las Vegas, NV, USA, pp. 1–4, 2018.

[11] A. Carrega, M. Repetto and A. Zafeiropoulos, "A middleware for mobile edge computing," *IEEE Cloud Computing*, vol. 4, no. 4, pp. 12, 2017.

[12] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 38–43, 2017.

[13] 5G. Americas, "New services & applications with 5G ultra-reliable low latency communications," 2018. [Online]. Available: https://www.5gamericas.org/new-services-applications-with-5g-ultra-reliable-low-latency-communications/.

[14] R. A. Dziyauddin, D. Niyato, N. C. Luong, M. A. M. Izhar,M Hadhari *et al.* "Computation offloading and content caching delivery in vehicular edge computing: A survey," *ArXiv Preprint ArXiv:1912.07803*, vol. abs/1912.07803, pp. 1–29, 2019.

[15] R. Mahmud, S. N. Srirama, K. Ramamohanarao and R. Buyya, "Quality of experience (QoE)-aware placement of applications in Fog computing environments," *Journal of Parallel and Distributed Computing*, vol. 132, pp. 190–203, 2019.

[16] R. Deng, R. Lu, C. Lai, T. H. Luan and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[17] X. Sun and N. Ansari, "Latency aware workload offloading in the cloudlet network," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1481–1484, 2017.

[18] S. Maheshwari, D. Raychaudhuri, I. Seskar and F. Bronzino, "Scalability and performance evaluation of edge cloud systems for latency constrained applications," in *Proc. of the Third ACM/IEEE Symp. on Edge Computing*, Seattle, WA, USA, pp. 286–299, 2018.

[19] S. Shekhar, A. Chhokra, A. Bhattacharjee, G. Aupy and A. Gokhale, "INDICES: exploiting edge resources for performance-aware cloud-hosted services," in *Proc. of the 1st IEEE/ACM Int. Conf. on Fog and Edge Computing*, Madrid, Spain, pp. 75–80, 2017.

[20] S. Sarkar, S. Chatterjee and S. Misra, "Assessment of the suitability of fog computing in the context of internet of things," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 46–59, 2018.

[21] K. Sasaki, S. Makido and A. Nakao, "Vehicle control system for cooperative driving coordinated multi-layered edge servers," in *Proc. of the IEEE 7th Int. Conf. on Cloud Networking*, Tokyo, Japan, pp. 1–7, 2018.

[22] Y. Gao, W. Hu, K. Ha, B. Amos, P. Pillai *et al.* "Are cloudlets necessary?," Technical Report CMU–CS–15–139, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, 2015.

[23] M. Satyanarayanan, P. Bahl, R. Cáceres and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[24] T. G. Rodrigues, K. Suto, H. Nishiyama and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *IEEE Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[25] H. Wu, Y. Sun and K. Wolter, "Energy-efficient decision making for mobile cloud offloading," *IEEE Transactions on Cloud Computing*, vol. 8, no. 2, pp. 570–584, 2020.

[26] C. Li, J. Tang, H. Tang and Y. Luo, "Collaborative cache allocation and task scheduling for data-intensive applications in edge computing environment," *Future Generation Computer Systems*, vol. 95, pp. 249–264, 2019.

[27] C. Sonmez, A. Ozgovde and C. Ersoy, "Fuzzy workload orchestration for edge computing," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 769–782, 2019.

[28] W. Hu, Y. Gao, K. Ha, J. Wang, B. Amos *et al.*, "Quantifying the impact of edge computing on mobile applications," in *Proc. of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*, Hong Kong, China, pp. 1–8, 2016.

[29] C. Jiang, X. Cheng, H. Gao, X. Zhou and J. Wan, "Toward computation offloading in edge computing: A survey," *IEEE Access*, vol. 7, pp. 131543–131558, 2019.

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

[31] X. Lyu, H. Tian, L. Jiang, A. Vinel, S.Maharjan *et al.* "Selective offloading in mobile edge computing for the green internet of things," *IEEE Network*, vol. 32, no. 1, pp. 54–60, 2018.

[32] T. Q. Dinh, S. Member, J. Tang, Q. D. La, T. Q. S. Quek *et al.* "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, 2017.

[33] F. Samie, V. Tsoutsouras, L. Bauer, S. Xydis and D. Soudris, "Computation offloading management and resource allocation for low-power IoT edge devices," in *Proc. of the IEEE 3rd World Forum on Internet of Things*, Reston, VA, USA, pp. 7–12, 2016.

[34] R. Mahmud, K. Ramamohanarao and R. Buyya, "Latency-aware application module management for fog computing environments," *ACM Transactions on Internet Technology*, vol. 19, no. 1, pp. 1–21, 2018.

[35] S. Shekhar and A. Gokhale, "Dynamic resource management across cloud-edge resources for performance-sensitive applications," in *Proc. of the 17th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing*, Madrid, Spain, pp. 707–710, 2017.

[36] W. Tärneberg, A. Mehta, E. Wadbro, J. Tordsson, J. Eker *et al.* "Dynamic application placement in the mobile cloud network," *Future Generation Computer Systems*, vol. 70, pp. 163–177, 2017.

[37] S. Wang, M. Zafer and K. K. Leung, "Online placement of multi-component applications in edge computing environments," *IEEE Access*, vol. 5, pp. 2514–2533, 2017.

[38] D. Zeng, L. Gu, S. Guo and Z. Cheng, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3702–3712, 2016.

[39] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing-based IoT," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2146–2153, 2018.

[40] S. Azizi, "A priority-based service placement policy for fog-cloud computing systems," *Computational Methods for Differential Equations*, vol. 7, no. 4, pp. 521–534, 2019.

[41] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires *et al.* "Cost-effective processing for delay-sensitive applications in cloud of things systems," in *Proc. of the IEEE 15th Int. Symp. on Network Computing and Applications*, Cambridge, MA, USA, pp. 162–169, 2016.

[42] Y. Li and S. Wang, "An energy-aware edge server placement algorithm in mobile edge computing," in *Proc. of the IEEE Int. Conf. on Edge Computing*, San Francisco, CA, USA, pp. 66–73, 2018.

[43] V. Scoca, A. Aral, I. Brandic, R. De Nicola and R. B. Uriarte, "Scheduling latency-sensitive applications in edge computing," in *Proc. of the 8th Int Conf. on Cloud Computing and Services Science*, Madeira, Portugal, pp. 158–168, 2018.

[44] D. G. Roy, D. De, A. Mukherjee and R. Buyya, "Application-aware cloudlet selection for computation offloading in multi-cloudlet environment," *Journal of Supercomputing*, vol. 73, no. 4, pp. 1672–1690, 2017.

[45] M. Taneja and A. Davy, "Resource aware placement of IoT application modules in fog-cloud computing paradigm," in *Proc. of the IFIP/IEEE Int. Symp. on Integrated Network and Service Management*, Lisbon, Portugal, pp. 1222–1228, 2017.