IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 8, 2021, accepted March 1, 2021, date of publication March 8, 2021, date of current version March 16, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3064397

Integrating Edge Computing into Low Earth Orbit Satellite Networks: Architecture and Prototype

CHENGCHENG LI[1,2], YASHENG ZHANG[1], RENCHAO XIE[2], (Senior Member, IEEE), XUEKUN HAO[3], AND TAO HUANG[2], (Member, IEEE)

[1]The 54th Research Institute of CETC, Shijiazhuang 050081, China
[2]School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3]The 50th Research Institute of CETC, Shanghai 200331, China

Corresponding author: Renchao Xie (renchao_xie@bupt.edu.cn)

This work was supported in part by the Excellent Postdoctoral Study Project Funding of Hebei Province under Grant B2019005006.

ABSTRACT Low Earth Orbit (LEO) satellite network is a cost-efficient way to achieve global covering for wide-area Internet of Things (IoT). As more and more IoT applications require large amounts of computing resources, cloud computing paradigm becomes one of the IoT's main enablers. Abundant resources can be used to execute computation-intensive IoT applications in the cloud. Moreover, edge computing has emerged to alleviate the high latency and low bandwidth problem of cloud computing. The integration of edge computing into LEO networks (which is called *LEC* in this paper) can improve satellite IoT network's performance. In addition, it is an effective way to support delay-sensitive and resource-hungry wide-area IoT applications. However, there are many technical challenges for LEC, which is different from edge computing in terrestrial networks. Therefore, we study LEC in depth and a novel system architecture is proposed. A LEC prototype system is implemented which verifies our design. The simulation result demonstrates that LEC can improve system performance compared with cloud computing in LEO networks.

INDEX TERMS Edge computing, Internet of Things, LEO satellite network.

I. INTRODUCTION

Satellite network is a cost-efficient solution to solve many covering problems faced by terrestrial-based wide-area Internet of Things (IoT) [1]. Firstly, satellite network is not limited by extreme topographies due to its covering advantages. Secondly, as a supplement and extension to the terrestrial wireless network, satellite network is an important approach to achieve global IoT devices covering. Moreover, Low Earth Orbit (LEO) satellite network [2] has advantages over Geostationary Earth Orbit (GEO) satellite communication system for satellite IoT network. First, satellite-ground links' propagation latency of LEO satellites is much less than that of GEO satellites. Second, most satellite IoT devices are designed to be small-sized, long-life and consume less power. The signal loss due to propagation shall be smaller in LEO networks, which helps the terminal design to reach the ideal pattern.

As the IoT generates a huge amount of data that needs processing and storage, cloud computing paradigm becomes one of the IoT's main enablers [3]. Abundant and elastic resources are available to run computation-intensive IoT applications in the cloud. However, as more and more applications (such as smart transportation, augmented reality) require large amounts of computing and network resources [4], IoT devices suffer from low bandwidth and high latency when communicating with cloud servers.

Edge computing has emerged as an enabling technology to alleviate the high latency and low bandwidth problem [3]. Edge computing can enhance system performance and QoE of users by deploying processing and storage resource closer to users. Additionally, many resource-hungry and delay-sensitive applications can be supported by utilizing edge computing. Examples for these applications include Autonomous Driving and AR.

The integration of edge computing [5] into LEO networks can improve satellite IoT network's performance by providing near-device processing capability. Furthermore, delay-sensitive and resource-hungry wide-area IoT applications can be supported. Autonomous border monitoring is an example of those applications.

We think that deploying processing resource on LEO satellites is an effective way to enhance satellite IoT network with

The associate editor coordinating the review of this manuscript and approving it for publication was Takuro Sato.

edge computing. In this way, large amount of data generated by IoT devices can be processed directly by LEO satellites, instead of terrestrial cloud servers. Therefore, much network traffic is avoided in satellite IoT network and the processing delay is reduced. For instance, to make the overall delay lower, data generated by an IoT device can be processed by a LEO satellite. Such LEO network's edge computing is called ***LEC*** in this paper.

Although satellite IoT network's performance can be improved by LEC, there are many technical challenges different from edge computing in terrestrial networks. Firstly, one satellite's storage and processing resource is relatively scarce. This is because the size and weight of a satellite is limited and space environment is rigorous. This raises the need for integrating all the satellites' resource to provide satisfactory service. Secondly, an IoT device frequently handovers from one satellite to another, because LEO satellites fly at high speed.

Therefore, it is not appropriate to adopt terrestrial edge computing architectures in LEO networks, e.g. ETSI's Multi-access Edge Computing (MEC). When designing system architecture of LEC, one needs to take LEO networks' characteristics into account.

So this paper studies LEC in depth and a novel LEC system architecture is proposed. In specific, our proposed architecture has a 3-layer user plane and a two-level control plane. The control plane is comprised of a global LEC controller and controlling agents on all the satellites. In this architecture every satellite's resource can be ''pooled'', so it is possible that the whole system's resource is efficiently utilized to process IoT data. Moreover, we implement a LEC prototype system based on proposed architecture.

The contribution of this paper can be summarized as follows.

- Considering LEO networks' characteristics, we give design principles of the LEC system architecture. Then based on these design principles, we propose the LEC system architecture. Additionally, we analyze the key technologies that are necessary for the efficient operation of the system.
- The implementation method of the LEC prototype system is described, including the virtualization technologies, service migration, the application for demonstration, etc.
- We present functionalities that are achieved in the LEC prototype system and compare the performance of LEC between cloud computing.

The rest of this paper is organized as follows: Related works are introduced in Section II. We discuss alternatives of edge computing deployment in LEO networks and propose LEC architecture in Section III. In Section IV the implementation method of the LEC prototype system is described. The functions of the LEC prototype system and performance evaluation are presented in Section V. In Section VI we conclude this paper and envision some future work.

II. RELATED WORK

There are three widely-adopted edge computing architecture in the terrestrial network: Cloudlet [6], Fog computing [7] and MEC [8].

Cloudlet aims to extend remote datacenter cloud services closer to end users [8]. A cloudlet is merely a micro datacenter offering access to end users over Wi-Fi. End users can deploy and manage their own VMs in Cloudlets [6].

Fog computing enables a cloud computing architecture away from centralized cloud datacenters. It considers a lot of geographically wide-spread edge nodes as part of a distributed and collaborative cloud [8]. It also enables device management and network management at the network edge [9].

MEC enables cloud computing capabilities at the Radio Access Network (RAN) edge close to end users. Utilizing MEC, one can develop a wide range of new applications and services. Service providers can also benefit from using MEC by collecting more information about customers, e.g. their location and interests [8].

Next we review related works about edge computing in satellite networks.

Satellite MEC is proposed in [10] and the authors investigate 3 offloading location alternatives: proximal terrestrial offloading, satellite-borne offloading and remote terrestrial offloading. Cooperative computation offloading is also presented: satellite MEC servers can execute users' processing tasks by cooperation. However, they don't consider cooperative computation offloading in the scenario of satellite-borne offloading.

A system architecture for cloud/edge computing in the space-air-ground network is presented in [11]. In that architecture, communication to cloud servers is provided by satellites and edge computing is provided by drones. It can take limitation about remote processing and energy into account when offloading resource-hungry tasks.

In [12] the authors design a strategy to efficiently schedule the edge servers in the satellite terrestrial networks to provide more powerful edge computing services. In order to efficiently allocate satellite edge computing resource in this strategy, they propose a double edge computation offloading algorithm to optimize energy consumption and reduce latency by assigning tasks to edge servers with minimal cost.

The authors of [13] propose a game-theoretic approach to the optimization of computation offloading strategy in satellite edge computing. As metrics of optimizing performance, a task's response time and energy consumption are computed based on the queuing theory. They theoretically prove the existence and uniqueness of the Nash equilibrium and propose an iterative algorithm to find the Nash equilibrium.

In [14], the authors formulate the latency and energy optimization for MEC enhanced satellite-based internet of things networks as a dynamic mixed-integer programming

problem. The optimal solutions of this problem are hard to obtain. Therefore, the complex problem is decomposed into two sub-problems. The first sub-problem is computing and communication resource allocation with fixed user association and offloading decision. The second sub-problem is joint user association and offloading with optimal resource allocation. For the sub-problem of resource allocation, it is proven that one can obtain the optimal solution based on Lagrange multiplier method. The authors further formulate the second sub-problem as a Markov decision process. And they propose a joint user association and offloading decision with optimal resource allocation based on deep reinforcement learning.

In order to intelligently use the satellite Internet of Things, in [15] an application scheme of satellite IoT edge intelligent computing is proposed, and how edge computing and deep learning play a role in satellite IoT image data target detection is analyzed. The authors simulated the proposed solution and experimented with the existing embedded processing board. Experiments show that the scheme can reduce the delay of acquiring images from satellites and performing target detection, and save backhaul bandwidth.

There are also more existing works on satellite empowered communication and computing systems. For example, in [16] the authors point out that the densification of large numbers of static small cells faces many fundamental challenges. This motivates them to develop software-defined space-air-ground integrated moving cells (SAGECELL), a programmable, scalable, and flexible framework to integrate space, air, and ground resources for matching dynamic traffic demands with network capacity supplies. The conceptual architecture of SAGECELL is elaborated in detail, and the technological benefits are emphasized. Their proposed framework SAGECELL is able to achieve obvious capacity improvement. The difference between this paper and [16] is that this paper utilizes satellites as edge computing node.

In [17], the authors consider a space-air-ground integrated mobile edge caching IoT system composed of satellite and unmanned aerial vehicles (UAVs), where LEO satellite broadcasts data, and UAVs collect the data from decentralized ground sensors. Since the sensors' low-power property leads data loss, fault-tolerant codes are employed for availability protection. This paper is different from [17] because this paper utilizes LEO satellites to provide computation offloading service.

The authors of [18] note that due to unique characteristics of satellite environment, one of the main challenges in this system is to accommodate massive random access (RA) requests of IoT devices while minimizing their energy consumptions. In that paper, they focus on the reliable design and detection of RA preamble to effectively enhance the access efficiency in high-dynamic low-earth-orbit (LEO) scenarios. The difference between this paper and [18] is that they focus on the communication aspect and we focus on the edge computing aspect.

III. EDGE COMPUTING IN LEO NETWORKS

A LEO network [19] usually includes a lot of LEO satellites, satellite network gateway(s) and user terminals, e.g. IoT devices. We assume that there are inter-satellites links (ISLs) in the LEO network [20], so there is no need to deploy a lot of gateways in geographically dispersive sites to achieve global network access. Taking advantage of edge computing, data generated by IoT devices in any geographical area can be processed in timely fashion.

Next we explain the performance gain by combine edge computing and LEO network. Satellite network is used by users when there is no ground BSs. If the user can access the terrestrial mobile network by ground BSs, he/she usually won't access satellite network. In this scenario, it is better to deploy edge computing resource near to the ground BSs. So there is no need to combine LEO network with cloud computing.

However, this work focuses on the scenario that the user can't access the network by ground BSs, e.g. the user is on the sea. In such scenario, if the user wants to get computing service from the cloud (deployed on the ground), he/she suffers from low bandwidth and high latency. This is because the network path between the user and the cloud server includes at least 2 satellite-ground links and probably many inter-satellite links as shown in Figure 1. The combination of LEO network and edge computing can alleviate the high latency and low bandwidth problem, because users can get computing service directly from satellites (LEC nodes). The network path between the users and the LEC node only include 1 satellite-ground link and probably a few inter-satellite links.

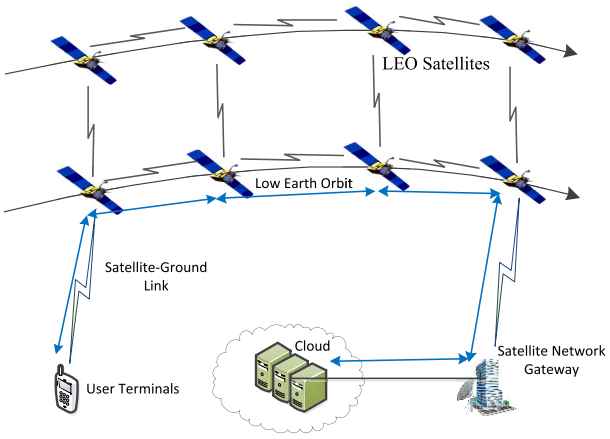


FIGURE 1. A user utilizes cloud in the satellite network.

A. ALTERNATIVES OF LEO NETWORKS' EDGE COMPUTING

When studying LEO networks' edge computing, one need to firstly determine the way of deployment, i.e. where to deploy edge computing resource. If one deploys edge computing resource in the satellite network gateway, all the data

generated by massive IoT devices needs to be transported to the edge computing node in the satellite network gateway to be processed. There are several disadvantages for this deployment option. Firstly, there may be many ISLs on the network path between the edge computing node (in the satellite gateway) and an IoT device. Transporting data from IoT devices to the edge computing node will consume these ISLs' capacities. Secondly, the aim of introducing edge computing into LEO satellite IoT network is to utilize edge computing nodes to reduce the delay of data processing. However, the data processing delay is probably very high because of the large communication delay between the edge computing node (in the gateway) and IoT devices.

If one deploys edge computing resource on LEO satellites, data generated by IoT devices can be processed directly by LEO satellites. When the IoT device's accessing satellite (or its neighbor satellite) has the requested processing function, the end-to-end path between the IoT device and the satellite providing processing function doesn't include many ISLs. So the bandwidth of these ISLs can be saved. Additionally, the data processing delay will be largely reduced.

Therefore, by deploying edge computing resource on LEO satellites, network traffic and processing delay of data can be largely reduced. So we investigate edge computing in LEO networks where edge computing resource is deployed on LEO satellites. In this paper we denote edge computing system in LEO networks as "*LEC*" and every LEO satellite is corresponding to a LEC node. Because the satellite network gateway with edge computing resource can be used to support delay-tolerant applications, we think deploying edge computing resource in the satellite network gateway is still helpful.

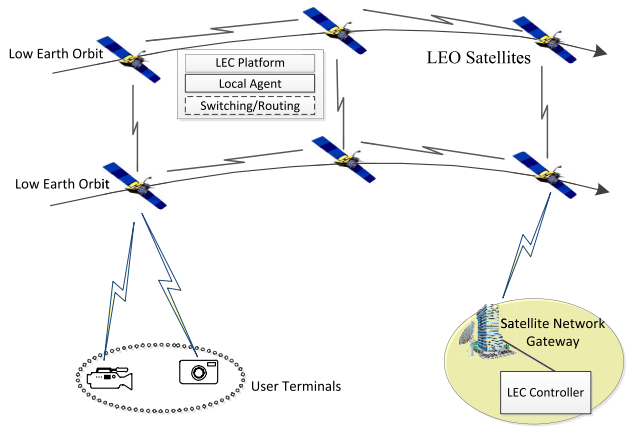


FIGURE 2. LEC system.

As shown in FIGURE 2, LEC system is composed of the LEO satellites constellation, the satellite network gateway and user terminals, e.g. IoT devices. LEO satellites fly at very high speed. Any two satellites in the constellation can communicate with each other by one hop or several hops using ISLs. To achieve the multi-hop communication between satellites, there are switching and routing devices on the satellites, which is out of the scope of this paper. The satellite network gateway is deployed on the ground. It provides interconnection between the terrestrial network and the satellite network. In the LEC system, the LEC controller is placed in the satellite network gateway and the local controlling agent is deployed on every satellite. The LEC controller and local agents constitute the two-level control plane of the system, which will be described in Section III-C. User terminals can get edge computing service from the LEC system, e.g., data generated by IoT devices can be processed by LEO satellites (LEC nodes).

B. DESIGN PRINCIPLES

LEO satellites fly at very high speed and the topology of the LEO network changes frequently. Additionally, the processing and storage resource of the LEO satellite is more scarce than MEC node in terrestrial networks. Moreover, the propagation delay of satellite-ground links and inter-satellite links are longer than the wireline links in terrestrial networks. Due to the above characteristics of the LEO networks, the design of the LEC system architecture should obey the following design principles:

(1) Service mobility capability: Because LEO satellites fly at very high speed, the hop count between the user terminal and the service will increase if a service can't be migrated from one satellite to another. This will cause higher service delay and more network traffic. Therefore the system should be able to migrate the service. Moreover, after the service is migrated, the system should help the user to find the service.

(2) Pooling of satellites' resource: Considering that a single LEO satellite's resource is much more scarce, in order to increase the number of IoT devices served by the system, it is necessary to utilize the resource of all the satellites in the system efficiently. Additionally, large computation tasks need cooperative computing by multiple satellites, which also requires pooling of all the satellites' resource.

(3) Direct communication: In current cloud computing system, there is usually a load balancing server, which can receive the request from the user and redirect the request to the actual serving node. However, if there is a proxy node (like the load balancing server) in LEC system, the communication between the proxy node and the user terminal will cause redundant network traffic. The communication between the proxy node and the actual serving LEC node (LEO satellite) will also cause redundant network traffic. This is unacceptable in LEO satellite IoT network. Therefore, the communication between the user terminal and the serving LEC node should be direct.

C. SYSTEM ARCHITECTURE

Our proposed LEC system architecture is illustrated in FIGURE 3. The whole system can be divided into user plane and control plane.

The user plane of every satellite is composed of 3 layers. The satellite-based infrastructure layer includes satellite payloads such as CPU, GPU, FPGA, DSP, disk, switch, which

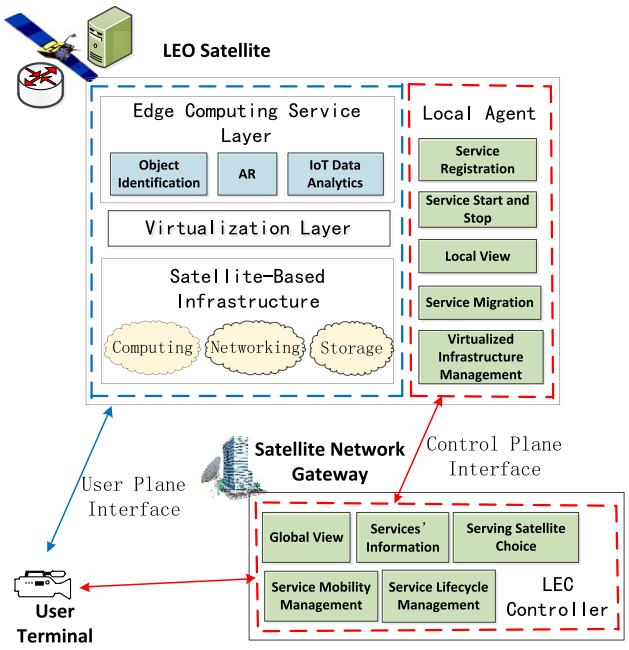


FIGURE 3. System architecture.

can be classified into computing, storage and networking devices. The virtualization layer is responsible for abstract these devices to virtualized resource. The edge computing services layer is composed of various services which are developed by the 3rd party, such as object identification, augmented reality, sensing data analytics and automatic decision. This layer is based on computing, storage and networking resource provided by the virtualization layer.

The control plane is composed of the LEC controller and every satellite's local controlling agent. Every satellite's local agent can communicate with the LEC controller through the feeder link and satellites' relay. For better performance and reliability, we recommend to place the LEC controller in the satellite network gateway. The LEC controller manages the LEC system's operation and the LEC system's global knowledge is maintained by it. The agent is a lightweight daemon that controls the behavior of hosting LEO satellite's user plane according to the LEC controller's decision. It also collects the user plane's status.

The main functions of the LEC controller are as follows:

(1) Global view: Some LEC system's global information is maintained by the LEC controller, which includes all the links' latency and bandwidth, constellation topology, all the satellites' available resource and capacity of resource, etc.

(2) Services' location information: The LEC controller should know all the services' and their associated images' location information. During users' service discovery process, the location information is necessary.

(3) Serving satellite choice: The LEC controller should be able to choose an appropriate satellite to provide service to the user, which is needed by the service discovery process described in Section III-D.

(4) Service mobility management: A service continuity mechanism is needed after a service instance migrates from a satellite to another. Some signaling messages between the LEC controller, the source LEC node, the destination LEC node and the user terminal are needed.

(5) Service lifecycle management: The LEC controller is responsible for managing the whole lifecycle of a service, from pulling the container image for the service to deploying the service, and finally deleting the service.

The main functions of the local agent are as follows:

(1) Service registration: The agent monitors the running service on the hosting satellite and registers these services to the LEC controller.

(2) Service start and stop: The agent starts and stops the service on the hosting satellite according to the LEC controller's decision.

(3) Local view: The agent monitors and collects the resource usage statistics, resource capacity and links status of the hosting satellite.

(4) Service migration: The agent is responsible for the signaling about service migration with the LEC controller and the agent on the destination LEC node. It also transfers the necessary data for service migration to the destination LEC node.

(5) Virtualized infrastructure management: The local agent manages virtualized infrastructure on the hosting satellite, which is abstracted from satellite-based physical devices.

D. KEY TECHNOLOGIES

In the proposed system architecture, in order to operate efficiently, the LEC system also needs some key technologies:

- Service Discovery: Because the LEO satellite's resource is more scarce than the MEC node in terrestrial networks, placing every service on all the satellites is not possible. So when a user request for a service, the service is only deployed on part of satellites in the constellation. So the LEC system needs to find the satellite which can serve the user. Moreover, the LEC system needs to help the user terminal to select the 'good' serving satellite, i.e. the satellite that can satisfy the user's QoE demand. Additionally, if there is no satellite that both has the requested service and can satisfy the user's QoE demand, the LEC system should select a good satellite for service deploying. It is intuitive that the user should get edge computing service from its access satellite. However, this may degrade the user's QoE because the access satellite may be already overloaded or it will fly away immediately and can't be accessed by the user terminal. It is clear that the serving satellite choice strategy has important influence on users' QoE and the system's performance. Moreover, to ensure backward compatibility, it is better to integrate the service discovery process into the DNS process.
- Service Migration: Service migration is necessary to provide high-quality service to users consistently. Service migration can be classified into two kinds: stateless

migration and stateful migration. Running states of the application aren't moved by stateless migration [21]. After stateless migration, the computation process of the application has to begin from scratch on the destination LEC node. This means that the computation on the source LEC node is useless and the computation resource is wasted. So it is better to adopt stateful migration. Additionally, mobility management with respect to service is needed to help the user request the migrated service. This is different from mobility management with respect to the user terminal.

- Service Migration Decision: On the one hand, if a service always stays on a LEO satellite, the service delay and redundant network traffic will be larger and larger. On the other hand, if the service is migrated too frequently, the service migration process will cause huge traffic overhead. Additionally, it may not be the best choice to migrate the service to the current access satellite of the user terminal, because the access satellite may be overloaded. Therefore, the LEC system should decide whether to migrate the service and to which LEO satellite the service is migrated. There are already many works which study service migration decision strategy in MEC. But it is necessary to consider the dynamicity and periodical characteristics of the LEO network when designing the migration decision strategy in LEC.
- Distributed Computing: When a user wants to offload a computation-intensive task to the LEC system, a single satellite can't complete the task in time, because its computation resource is limited. In this use case, it is necessary to utilize many satellites' resource to complete the offloaded computation task with less time. Therefore, the computation task needs to be divided into small subtasks and these subtasks needs to be scheduled to several satellites. The subtasks scheduling problem is very complex because many factors should be considered, such as the available computation resource and residual energy of all the satellites, links' delay and capacity between all the satellites. In addition, there may be different optimization objectives, e.g. minimizing task completion delay, balancing satellites' load and minimizing communication overhead.

IV. IMPLEMENTATION OF PROTOTYPE

This section introduces the implementation method of the LEC prototype system. In order to implement a LEC prototype system as described in Section III, it is needed to solve some problems as follows:

How to achieve lightweight virtualization: To provide separate and secure environment for every edge computing service on the satellite, it is necessary to use virtualization technologies. There are mainly two kinds of virtualization technologies, i.e. container and virtual machine. We adopt container that is more lightweight, considering the relatively small resource capacity of every LEO satellite. An instantiated container implements an instance of service. Specifically, we use opensource container management tool Podman [22] to build containers which provide services. Moreover, the local agent uses Podman's API to start, stop, checkpoint, restore containers, etc.

How to make the global view available to the LEC controller: The global view is vital for the LEC controller's function, such as serving satellite choice, service migration decision. In order to make the global view available to the LEC controller, every local agent gets the hosting satellite's information using Linux command and Podman command. Moreover, agents should send the hosting satellite's information to the LEC controller. We use a Python's module - Socket to transport signaling messages (including the above mentioned information) between every agent and the LEC controller. In the initialization phase, the LEC controller opens the socket and waits for connection from every agent. Because LEC controller needs to get the above mentioned information from agents and send command to agents in a timely fashion, the socket will stay open.

How to achieve stateful service migration: Taking into account that a service is implemented by a container in the prototype system, stateful service migration can be implemented by migrating stateful container. Specifically, we utilize CRIU (Checkpoint/Restore In Userspace) to migrate container [23]. When the agent on the source LEC node receives the command to migrate service to a destination LEC node, it firstly checkpoints the corresponding container to get a snapshot of this container. Then the agent on the source LEC node sends, using Linux SCP, the snapshot file to the agent on the destination LEC node. Finally, the destination LEC node restores an identical container that can provide the service.

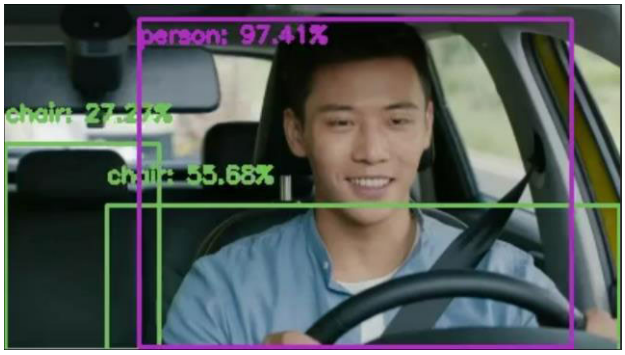
How to implement a computation offloading application for demonstration: We adopt object tracking as the demo service, which identifies specific objects and marks them in a video. The first reason to adopt target tracking is that it is an important functional module for many emerging applications. The second is that target tracking needs to offload computation tasks to LEC node because its computation-intensive nature. In edge computing this is a common use case. Because this paper doesn't focus on object tracking algorithm, this service is developed based on Open Source Computer Vision Library (OpenCV) and Django. OpenCV is an open source computer vision software library. Django is a mature Python Web framework. We use Django to implement original video uploading, processed video downloading and service's front-end Web page.

V. FUNCTION VERIFICATION AND PERFORMANCE EVALUATION

In this section, we first present functionalities that are already achieved in the LEC prototype system, then compare the performance of LEC and cloud computing.

Firstly, we show that the object tracking application can be supported by the LEC prototype system. In this test, a car advertisement video (the original video) is uploaded to the

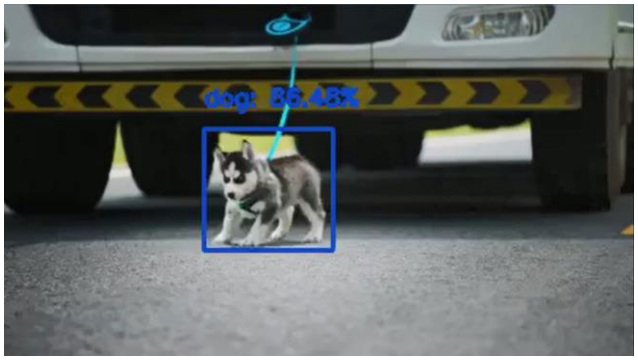
(a) The original video's first frame.



(b) The processed video's first frame.

FIGURE 4. The original video's and processed video's first frame.

(a) The original video's second frame.



(b) The processed video's second frame.

FIGURE 5. The original video's and processed video's second frame.

LEC system and the processed video is downloaded. The service identifies and marks some objects in the processed video. Both the original video's and processed video's same frame are presented in FIGURE 4 and FIGURE 5 respectively.

The original video's first frame and processed video's corresponding frame are shown in FIGURE 4. The subfigure (a) shows the original video's first frame, which contains a man sitting in a car. The service identifies this man and two chairs as shown by subfigure (b). It also marks them with boxes. The identification results' probabilities are given too.

Similarly, FIGURE 5 shows the original video's second frame and processed video's corresponding frame. A dog is contained in subfigure (a). The service marks the dog with a box as shown by the subfigure (b). The identification result's probability 86.48% is given too.

From above comparison, object tracking service can be supported by our LEC prototype system.

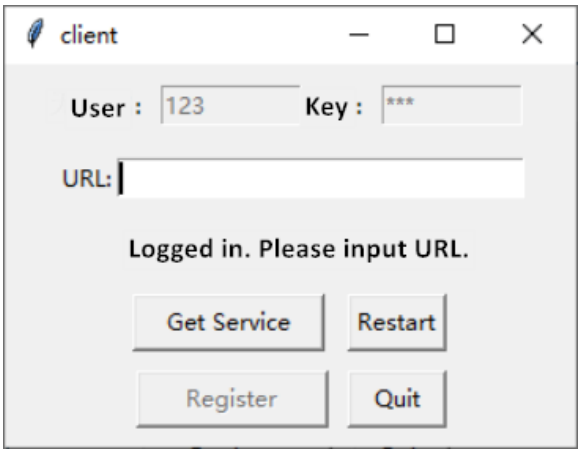
Next the service discovery function is presented. In current version of the LEC prototype system, an appropriate serving LEC node is chosen for deploying service and the user terminal is told to request the chosen LEC node for service in the service discovery process. The LEC system can provide service to the user after service discovery, i.e. from a chosen LEC node.

The service discovery process is shown in FIGURE 6 to FIGURE 7. As shown in FIGURE 6 (a), at first, the user needs to provide the name of requested service in the ''URL'' of the client end. Then this name is sent to the LEC controller. The LEC controller will choose a LEC node and the corresponding service is deployed on the LEC node. After that, the LEC controller tells the user terminal about the IP address of the LEC node and port number of service. Then the user knows how to get the service. As shown in FIGURE 6 (b), the user provides a service name and receives the IP address and port number information. Then the client end starts the web browser and uses the received information to access the service's web page, as shown in FIGURE 7.

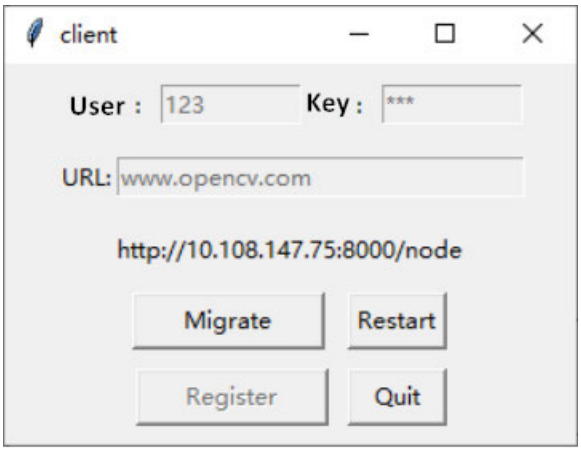
Thirdly, the LEC prototype system's service migration function is presented. Stateful service migration is to move the service from one LEC node (the source LEC node) to another specified LEC node (the destination LEC node). Additionally, after stateful migration, the service instance on the destination LEC node should have the same state as the service instance on the source LEC node.

Some logs are provided to verify the stateful service migration. As shown in the red box in FIGURE 8, the LEC node outputs ''video is not uploaded'' if it waits for uploaded video. And there is a number increasing by one every second after this statement.

The destination LEC node has a new container after service migration as shown in FIGURE 9, which means that the service is successfully migrated. The destination LEC node's log is shown in FIGURE 10. The log before the red box is obviously the same as that in FIGURE 8. Additionally in the



(a) The client-side that hasn't requested for a service.



(b) After the service discovery process, the client-side has discovered where the service is.

FIGURE 6. The client-side before and after service discovery process.

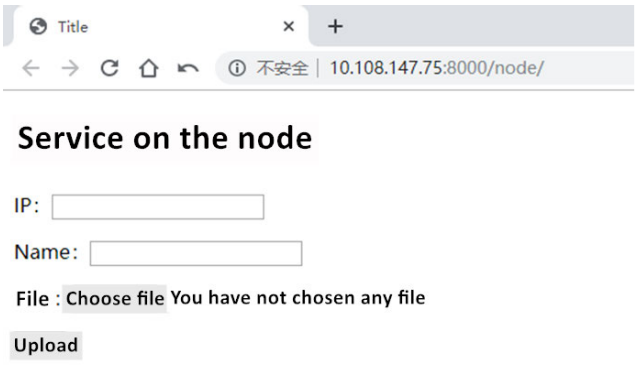


FIGURE 7. The user can get service from the selected satellite.

red box, the number continues to increase. The destination LEC node's log demonstrates that service migration is indeed stateful in the LEC prototype system, because it shows that this container is the same as the source LEC node's one.

Finally, we demonstrate the performance gains of combining edge computing and LEO (LEC) by compare LEC with

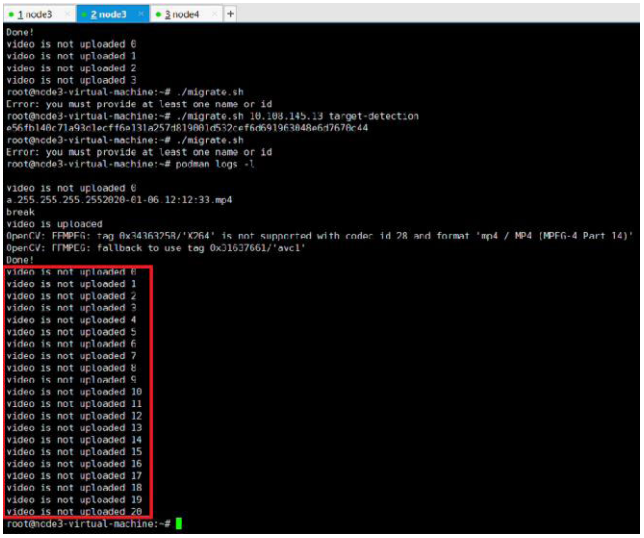


FIGURE 8. The source LEC node's log.

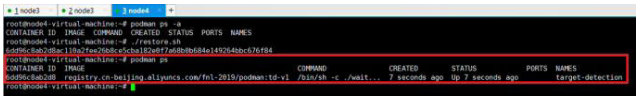


FIGURE 9. The information about destination LEC node's container.

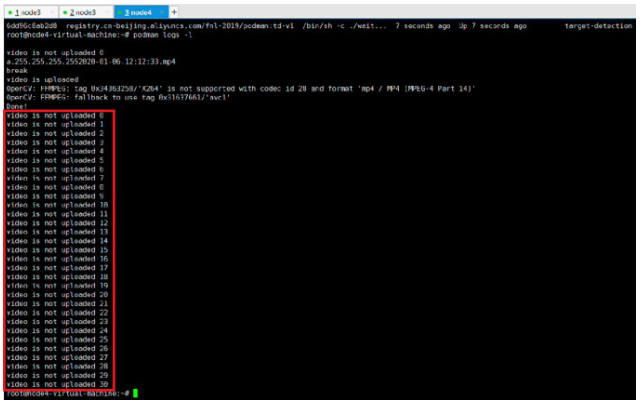


FIGURE 10. The destination LEC node's log.

cloud computing in LEO satellite networks. The performance metrics include users' service latency and occupied links' bandwidth caused by users. Users' service latency affects their QoE. Because satellite-ground links' and inter-satellite links' capacities are limited, occupied links' bandwidth should be small.

The simulation settings are presented as follows. We consider 3 different satellite constellations. The first constellation is composed of 6 orbits with 11 satellites in every orbit. The link delay between a user terminal and his access satellite is 5 ms and between two neighboring satellites is 14 ms. The second constellation is composed of 6 orbits with 12 satellites in every orbit. The link delay between a user terminal and his access satellite is 7 ms and between two neighboring satellites is 13 ms. The third constellation

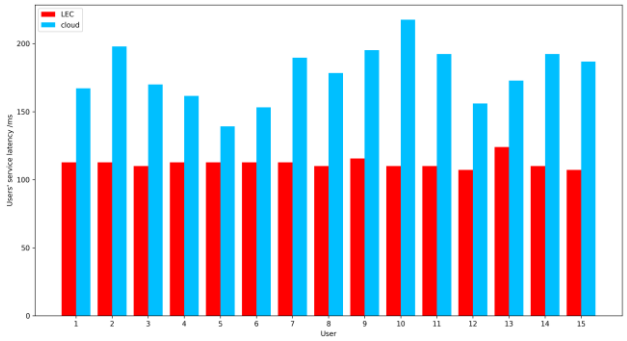


FIGURE 11. Every user's service latency in the scenario of 66 satellites and 15 users when using LEC and cloud computing respectively.

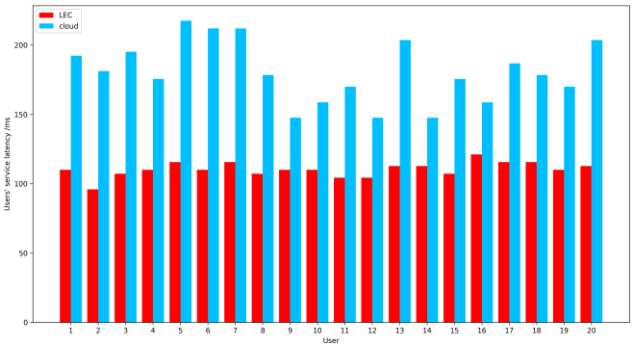


FIGURE 12. Every user's service latency in the scenario of 66 satellites and 20 users when using LEC and cloud computing respectively.

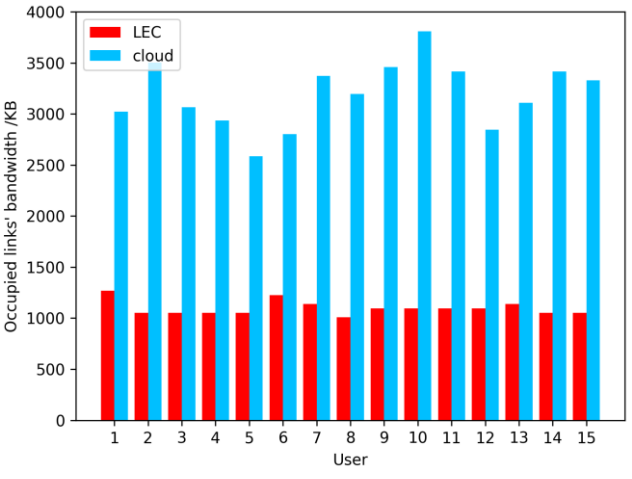


FIGURE 13. The occupied links' bandwidth caused by every user in the scenario of 66 satellites and 15 users when using LEC and cloud computing respectively.

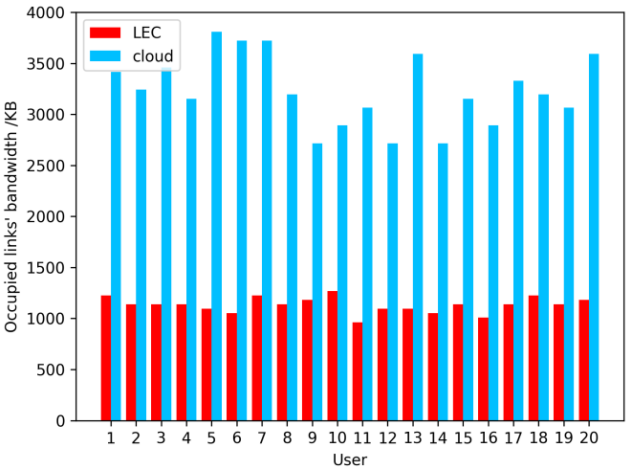


FIGURE 14. The occupied links' bandwidth caused by every user in the scenario of 66 satellites and 20 users when using LEC and cloud computing respectively.

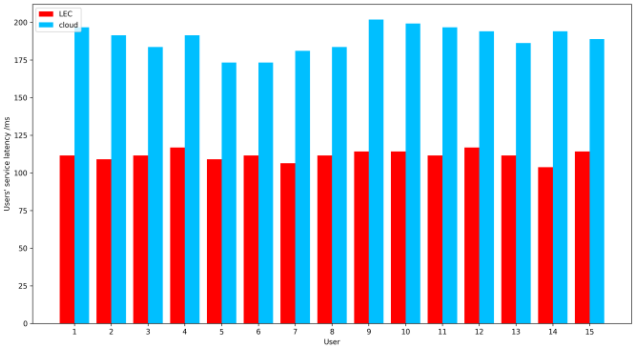


FIGURE 15. Every user's service latency in the scenario of 72 satellites and 15 users when using LEC and cloud computing respectively.

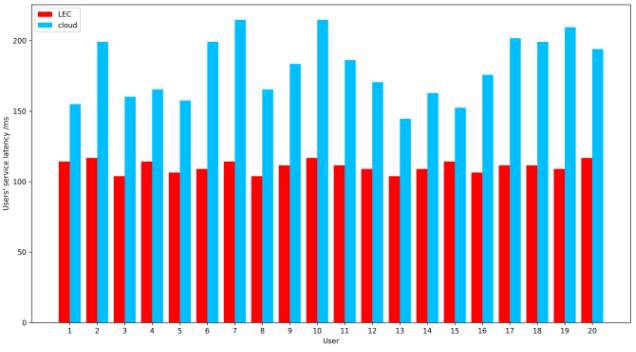


FIGURE 16. Every user's service latency in the scenario of 72 satellites and 20 users when using LEC and cloud computing respectively.

is composed of 12 orbits with 20 satellites in every orbit. The link delay between a user terminal and his access satellite is 8 ms and between two neighboring satellites is 7 ms. We simulate the scenario of 15 users which get the object tracking service from the LEC and cloud in the satellite network gateway respectively. We also simulate the scenario of 20 users. The location of every user terminal is randomly chosen and the serving satellite is randomly chosen among the 2-hop neighborhood of the access satellite. The traffic needed for uploading and downloading the original and processed video clip are 222 and 216 KB respectively. The processing delay by a satellite and the cloud are 30 ms and 10 ms respectively. To make the simulation result more convincing, we run the simulation 10 times and present the average result. For readers' convenience, we list simulation parameters in Table 1.

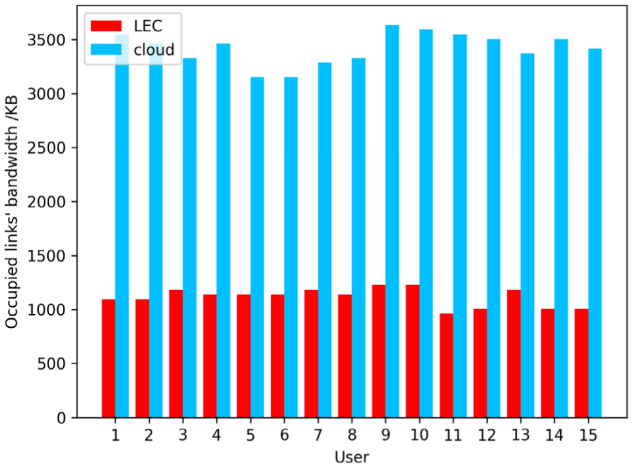


FIGURE 17. The occupied links' bandwidth caused by every user in the scenario of 72 satellites and 15 users when using LEC and cloud computing respectively.

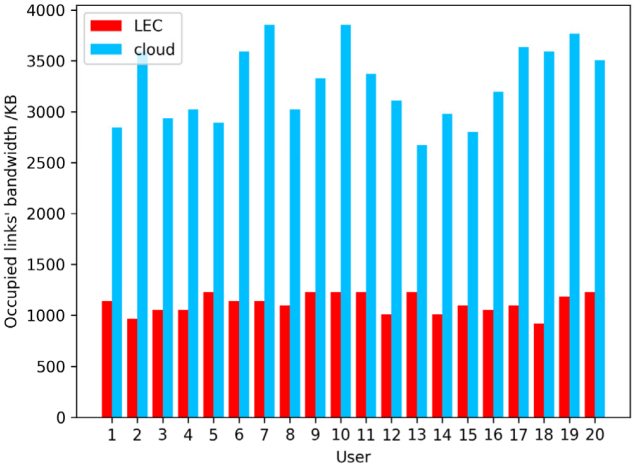


FIGURE 18. The occupied links' bandwidth caused by every user in the scenario of 72 satellites and 20 users when using LEC and cloud computing respectively.

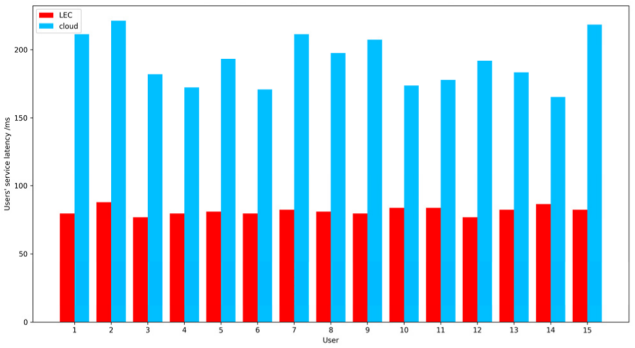


FIGURE 19. Every user's service latency in the scenario of 240 satellites and 15 users when using LEC and cloud computing respectively.

As shown in Figure 11, the service delay of every user when using LEC is less than that of the cloud. The similar result can be found in Figure 12, Figure 15, Figure 16, Figure 19 and Figure 20. The reason is that the hop count

TABLE 1. Simulation parameters.

	Constellation 1	Constellation 2	Constellation 3
Number of orbits	6	6	12
Satellites in every orbit	11	12	20
Number of users	15/20	15/20	15/20
Traffic needed for uploading the original video clip	222 KB	222 KB	222 KB
Traffic needed for downloading the processed video clip	216 KB	216 KB	216 KB
Link delay between a user terminal and his access satellite	5 ms	7 ms	8 ms
Link delay between two neighboring satellites	14 ms	13 ms	7 ms
Processing delay by a satellite	30 ms	30 ms	30 ms
Processing delay by the cloud	10 ms	10 ms	10 ms

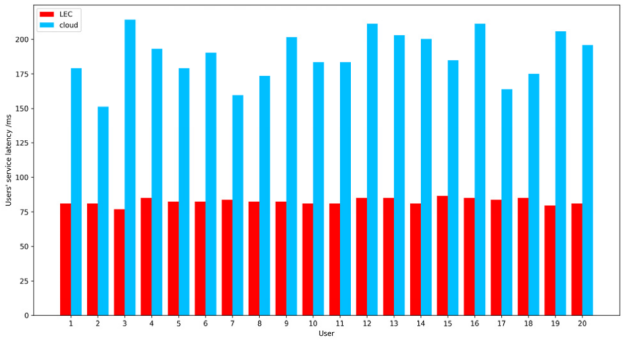


FIGURE 20. Every user's service latency in the scenario of 240 satellites and 20 users when using LEC and cloud computing respectively.

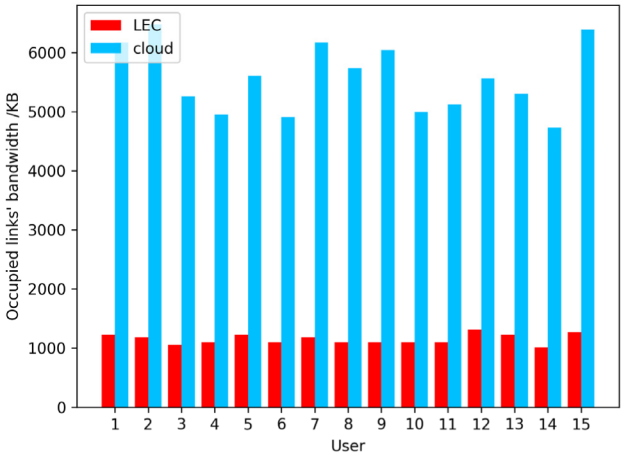


FIGURE 21. The occupied links' bandwidth caused by every user in the scenario of 240 satellites and 15 users when using LEC and cloud computing respectively.

between the user terminal and the service instance is usually smaller when using LEC compared with using cloud. This means that LEC helps to improve users' QoE.

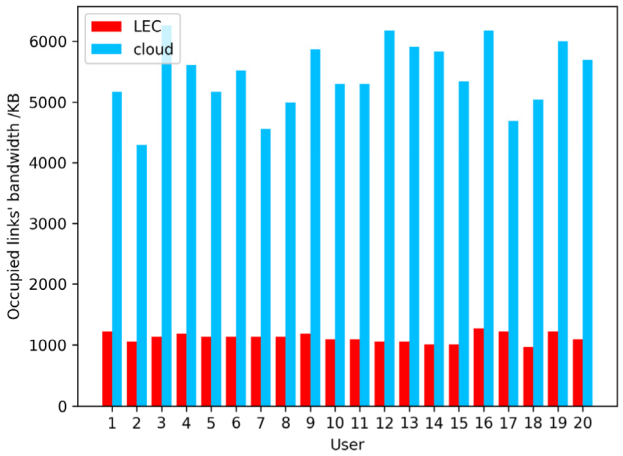


FIGURE 22. The occupied links' bandwidth caused by every user in the scenario of 240 satellites and 20 users when using LEC and cloud computing respectively.

As shown in Figure 13, for every user, all the links' bandwidth occupied by the user when using LEC is less than that bandwidth when using cloud. The reason is that LEC can provide service on LEO satellites that are closer to user terminals. This means that LEC helps to reduce bandwidth occupation in the LEO network.

VI. CONCLUSION

A system architecture is proposed for edge computing in LEO networks in this paper. We have developed a LEC prototype system that verifies our design based on this architecture. One can use it to test related algorithms and strategies, for example decision on service migration and choosing serving satellite. The performance evaluation has demonstrated that LEC outperforms cloud computing in LEO networks with respect to service delay and links' bandwidth.

For the future work, we are going to develop some classical satellite IoT application and leverage the LEC prototype system to demonstrate the advantage of LEC. Another direction is to integrate remote cloud into the LEC system and utilize both cloud and edge computing to accommodate more computation-intensive IoT applications. Moreover, it is also beneficial to study vehicle-based edge computing [24] which is similar to LEC.

REFERENCES

[1] Z. Qu, G. Zhang, H. Cao, and J. Xie, "LEO satellite constellation for Internet of Things," *IEEE Access*, vol. 5, pp. 18391–18401, 2017.
[2] C. Sun, Y. Zhang, L. Wang, and W. Zhao, "Analysis on adaptability of ground 5G technology in LEO constellation," *Radio Eng.*, vol. 48, no. 3, pp. 167–172, 2018.
[3] M. Goudarzi, H. Wu, M. S. Palaniswami, and R. Buyya, "An application placement technique for concurrent IoT applications in edge and fog computing environments," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1298–1311, Apr. 2021.
[4] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: Architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl.*, vol. 98, pp. 27–42, Nov. 2017.
[5] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2961–2991, 2018.
[6] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct. 2009.
[7] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2018.
[8] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.
[9] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014.
[10] Z. Zhang, W. Zhang, and F.-H. Tseng, "Satellite mobile edge computing: Improving QoS of high-speed satellite-terrestrial networks using edge computing techniques," *IEEE Netw.*, vol. 33, no. 1, pp. 70–76, Jan. 2019.
[11] 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.
[12] Y. Wang, J. Zhang, X. Zhang, P. Wang, and L. Liu, "A computation offloading strategy in satellite terrestrial networks with double edge computing," in *Proc. IEEE ICCS*, Dec. 2018, pp. 450–455.
[13] Y. Wang, J. Yang, X. Guo, and Z. Qu, "A game-theoretic approach to computation offloading in satellite edge computing," *IEEE Access*, vol. 8, pp. 12510–12520, 2020.
[14] G. Cui, X. Li, L. Xu, and W. Wang, "Latency and energy optimization for MEC enhanced SAT-IoT networks," *IEEE Access*, vol. 8, pp. 55915–55926, 2020.
[15] J. Wei and S. Cao, "Application of edge intelligent computing in satellite Internet of Things," in *Proc. IEEE SmartIoT*, Aug. 2019, pp. 85–91.
[16] Z. Zhou, J. Feng, C. Zhang, Z. Chang, Y. Zhang, and K. M. S. Huq, "SAGECELL: Software-defined space-air-ground integrated moving cells," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 92–99, Aug. 2018.
[17] S. Gu, Y. Wang, N. Wang, and W. Wu, "Intelligent optimization of availability and communication cost in satellite-UAV mobile edge caching system with fault-tolerant codes," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1230–1241, Dec. 2020.
[18] L. Zhen, A. K. Bashir, K. Yu, Y. D. Al-Otaibi, C. H. Foh, and P. Xiao, "Energy-efficient random access for LEO satellite-assisted 6G Internet of remote things," *IEEE Internet Things J.*, early access, Oct. 13, 2020, doi: 10.1109/JIOT.2020.3030856.
[19] X. Hu, H. Song, S. Liu, and W. Wang, "Velocity-aware handover prediction in LEO satellite communication networks," *Int. J. Satell. Commun. Netw.*, vol. 36, no. 6, pp. 451–459, Nov. 2018.
[20] Y. Liu, T. Huang, C. Zhang, and J. Liu, "Challenges and opportunities of future network," *Radio Commun. Technol.*, vol. 46, no. 1, pp. 1–5, 2020.
[21] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 140–147, Feb. 2018.
[22] *Podman*. [Online]. Available: https://podman.io/
[23] L. Ma, S. Yi, N. Carter, and Q. Li, "Efficient live migration of edge services leveraging container layered storage," *IEEE Trans. Mobile Comput.*, vol. 18, no. 9, pp. 2020–2033, Sep. 2019.
[24] V. Balasubramanian, S. Otoum, M. Aloqaily, I. A. Ridhawi, and Y. Jararweh, "Low-latency vehicular edge: A vehicular infrastructure model for 5G," *Simul. Model. Pract. Theory*, vol. 98, pp. 1–17, Jan. 2020.

CHENGCHENG LI received the B.S. degree from the Hebei University of Technology, in 2012, and the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2018. He is currently a Senior Engineer and a Postdoctoral Research Fellow with The 54th Research Institute of CETC. His current research interests include satellite networks, edge computing, and compute aware networking. He received the Excellent Postdoctoral Study Project Funding of Hebei Province. He was a Reviewer of IEEE China Communications and IEEE ICC 2018.

YASHENG ZHANG received the B.S. and M.S. degrees from Xidian University, in 1991 and 1994, respectively. He is currently a Professor with The 54th Research Institute of CETC. He is also one of the authors of the published book *The Integration Technology of Computer Network and Satellite Communication Network*. His current research interests include space-terrestrial integrated networks, satellite networks, and satellite mobile communication.

RENCHAO XIE (Senior Member, IEEE) received the Ph.D. degree from the School of Information and Communication Engineering, BUPT, in 2012. He is currently a Professor with BUPT. His current research interests include information-centric networking and 5G networks. He has served on the technical program committees (TPCs) of the CHINACOM 2016 and the 2012 IEEE Vehicular Technology Conference (VTC)-Spring. He has also served for several journals and conferences as a reviewer, including IEEE TRANSACTIONS ON COMMUNICATIONS, *Wireless Networks* (ACM/Springer), IEEE COMMUNICATIONS LETTERS, and 2011 IEEE GLOBECOM.

XUEKUN HAO received the Ph.D. degree from the PLA University of Science and Technology, in 2004. He is currently a Professor and the Associate Director of The 50th Research Institute of CETC. His current research interests include B5G/6G, satellite communication systems, MIMO/OFDM wireless systems, massive MIMO, M2M, and networked control systems.

TAO HUANG (Member, IEEE) received the B.S. degree in communication engineering from Nankai University, Tianjin, China, in 2002, and the M.S. and Ph.D. degrees in communication and information systems from the Beijing University of Posts and Telecommunications, in 2004 and 2007, respectively. He is currently a Professor with the Beijing University of Posts and Telecommunications. His current research interests include network architecture and software-defined networking.