IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON EMERGING TRENDS, ISSUES, AND CHALLENGES IN ENERGY-EFFICIENT CLOUD COMPUTING

Received November 13, 2017, accepted March 1, 2018, date of publication March 19, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2815629

User-Participatory Fog Computing Architecture and Its Management Schemes for Improving Feasibility

WON-SUK KIM, (Member, IEEE), AND SANG-HWA CHUNG, (Member, IEEE)
Department of Department of Computer Engineering, Pusan National University, Busan 46241, South Korea

Corresponding author: Sang-Hwa Chung (shchung@pusan.ac.kr)

This work was supported by the Institute for Information and Communications Technology Promotion funded by the South Korea Government (MSIT) under Grant B0126-16-1051 (a study on hyper connected self-organizing network infrastructure technologies for IoT service).

ABSTRACT The evolution of computing and networking technologies has opened the era of cloud computing, and the advent of the Internet of Things (IoT) paradigm has been questioning its limitations. Owing to advances in computer networks, cloud computing is improving, and the most promising technology is fog computing. Although fog computing is recognized as the most appropriate computing model for the IoT, it has not yet been widely used, and the major reasons are as follows. The replacement of the firmware and hardware of network equipment is inevitable; however, the operator in charge of carrying out this expensive task is unclear, and even if the operator is selected, the reason may be not rational. In addition, although fog computing is based on collaboration between several infrastructure operators and service providers, it is not clear who operates and manages the infrastructure. Furthermore, there is still a resource allocation problem for a fog service instance. In this paper, we propose a user participatory fog computing architecture and its management schemes to address the above problems related to its feasibility. In the proposed architecture, fog service instance placement optimization is performed based on service usage of participating users, which is formulated into a mixed-integer non-linear programming problem and then linearized. The proposed architecture and the fog service placement method are evaluated based on simulation, taking into account actual parameters of the IoT services and devices.

INDEX TERMS Fog computing, Internet of Things, optimization, software defined networking.

I. INTRODUCTION

Significant advances in computer networks have been made together with many new technologies. This is demonstrated by recent studies on new network paradigms, such as information-centric networking (ICN), software-defined networking (SDN), as well as data center networking (DCN) from cloud computing [1]–[3]. Cloud computing is a computing model that provides access from anywhere to several configurable resources, such as network, storage, computing, platforms, and services [4]. With this model, Internet service operators and corporate IT departments are encouraged to switch to cloud computing because they can obtain hardware flexibility and reduce maintenance with minimal administration [5]. This means that the transition of server instances of many Internet applications to the cloud data center is increasing very rapidly [6].

The Internet of things (IoT) is a new paradigm where many devices are connected to the Internet. Several services have emerged from the IoT, including crowd sensing, smart city, smart home, healthcare, autonomous vehicle, and augmented reality [7]–[9]. These services include features, such as high-volume traffic, large number of devices, and big data, which have triggered changes in viewpoints on the network [10]. In particular, big data has characteristics, such as volume, velocity, variety, and geo-distribution; therefore, the computing platform, as well as the network, should be able to support these features as an underlying technology [11].

The paradigm shift to the IoT presents new challenges for widely used cloud computing. IoT services with features, such as low latency and real-time, as well as big data, cannot be properly supported owing to fundamental weaknesses, such as round-trip latency of data center-based cloud

computing. In particular, it is not reasonable at present for a service based on devices with low-level computation capabilities to expect a low latency while leasing the high computational capacity of the cloud at the same time. There is a definite limitation when services that focus on context location, such as shopping center map service and services targeting a wide area, such as smart city operate collectively in the cloud [12], [13].

Fog computing is a new computing model proposed to solve various problems that occur when the above mentioned IoT services operate in a cloud data center. This is a concept that extends existing cloud services, such as computing, storage, and network to the network edge near the users or devices as shown in Fig. 1 [14]. In other words, it implies that some of the roles that cloud data centers play are moved to a number of geographically distributed physical network devices, such as switches, Wi-Fi access points (APs), IoT gateways (GWs), and mobile base stations. Therefore, when applying fog computing, it can naturally reduce the response time of network services while realizing all the advantages of cloud computing, supporting real-time services, and enabling networking taking regional characteristics into account.

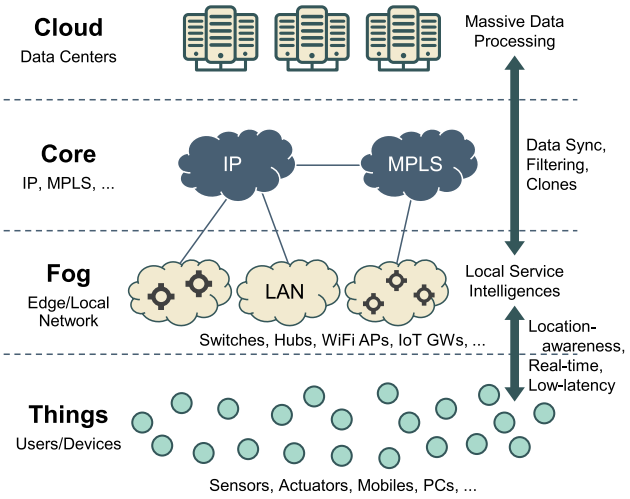

FIGURE 1. Fog computing hierarchy: the cloud and fog layers perform data synchronization, and the fog layer provides location-awareness, low-latency, and real-time services.

Typical advantages of fog computing include low latency, location awareness, mobility support, and device heterogeneity. For example, in the smart city based on fog computing, the smart traffic system and network access can be provided at the same time by fog devices, which operate fog service instances, such as Wi-Fi APs, base stations, and road side units. Autonomous vehicles can make considerable use of the area-specific characteristics, which can provide local traffic information or local features collected locally and report road-related events, such as construction or accident. When reporting specific information through augmented reality using the head-up display on the windscreen, gaze tracking, data analysis, and video processing can be performed inside the vehicle or processed using computing resources of adjacent fog devices. This is possible because it has a significantly lower latency compared to cloud computing. In the case of big data processing for smart city disaster monitoring service, pattern discovery can be performed in the cloud data center by collecting data from a geographically broad range of sensors over a long period of time. In addition, a function requiring rapidity, such as actuator operation by pattern matching can be configured to be performed quickly in fog devices.

A fog device can be primarily a network device and is typically located between the IoT devices or users and the core network as shown in Fig. 1. Naturally, a router in the core network—especially an edge router—can also become a fog device, but this can degrade overall network performance because the edge routers perform heavy load networking functions, such as access control and MPLS labeling. Moreover, because it is located at a fairly high layer, it cannot adequately reflect local characteristics, the delay time is difficult to predict, and the replacement cost can also be very high. A smart phone or an IoT gateway may be considered as a fog device, but it is likely to exist as a dedicated fog device for a specific service. Therefore, it is ideal that switches, hubs, Wi-Fi APs, IoT GWs and base stations in the local network will be general purpose fog devices.

Two essential resources are required to realize fog computing: hardware virtualization and SDN. As with the cloud data center, when fog devices are used as generic hardware, dynamic resource provisioning must be able to be performed to support flexibility and scalability. There are two main types of hardware virtualization, a hypervisor-based virtual machine (VM) and a container. The difference between these two virtualization is, in a nutshell, that the VM includes the guest OS and the container does not. That is, the VM provides independent hardware through full hardware virtualization, while the container only supports resource isolation by sharing the kernel. Although each technique has advantages and disadvantages, the container represented by Docker is very easy to deploy and requires less resources than the VM [15]. Therefore, in this study, the fog service instance is based on the container.

SDN supports fine-grained, flexible networking, and is well suited to gathering multi-level statistics. Similar to the way SDN supports VM migration in the cloud data center, it can route traffic from the IoT device in the local network to the target fog container (the fog service instance) through packet processing or routing configuration. It can also collect detailed network statistics and provide such statistics to IoT services.

A. MOTIVATIONS

Although fog computing has been recognized as a suitable computing model for the IoT era in many studies, it has not yet been widely used in real industry. The core application and necessity are not lacking, but the feasibility of that model becomes a significant challenge because of its nature. First, the role of existing cloud servers, such as data

processing and storage in network devices must be performed, which inevitably necessitates the replacement of firmware and hardware of the network equipment. This is equivalent to the fact that an infrastructure company spends a huge capital expenditures (CAPEX) in a business that may not be profitable. In addition, utilizing fog computing for IoT services is not developer-friendly because the fog containers should be developed considering various factors separately. That is, the service providers will consider the development of a fog container only after the appropriate fog computing infrastructure is installed in at least the target area, such as a city, a country, and the world.

The key challenging point of the feasibility of fog computing is the question of who should realize this? Since fog computing is based on the use of edge network devices, Internet service providers (ISPs), mobile carriers, and switch vendors can be the realizers individually or collaboratively. In terms of extending the capabilities of the cloud data center to the network edge, the cloud operator can be the realization entity. Even in terms of expanding the functionality of the IoT service itself, the IoT service provider can directly install and operate the fog devices. However, proper infrastructure is required for fog computing to operate in the correct meaning as mentioned earlier or as a general-purpose tool. In other words, it is not reasonable to install dedicated fog devices directly in target areas in order for an IoT service provider to provide its fog computing service to users, or to install a fog device in an area where the network infrastructure operator is not profitable. In addition, who will manage the resources of the installed fog devices?

The challenging issues in the feasibility of fog computing can be summarized as follows.

1) The core entity of fog computing should be determined outside existing interests, namely, an independent realization entity is required.

2) The construction of fog infrastructure should be carried out in the local area network.

3) The installation cost of fog devices should be shared.

4) Information that can enhance the use of fog computing by the IoT service operators should be provided.

5) Fog containers should be placed where they can satisfy the requirements of all entities.

B. CONTRIBUTIONS

We propose the user-participatory fog computing control architecture and its management method in this paper to address the above challenges. The main contributions of this study are as follows.

1) First, we propose a fog computing architecture based on an independent management entity called fog portal, which contributes to the feasibility of fog computing.

2) The proposed system is a user-participatory architecture that solves the problem of fog computing infrastructure expansion by the user or network administrator purchasing the fog device directly, connecting to the local network, and registering it in the fog portal. It is similar to crowdsourcing.

3) The fog portal can provide detailed service-specific usage through collaboration with the SDN controller in the local network, which is provided to the service operator in the form of a local usage distribution.

4) The decision to develop and distribute the fog container is entirely up to the service operator, but if he decides to place the fog container in a particular area, then the local fog manager can locate the fog container on any fog device in the network.

5) A user or a network operator who has installed a fog device can declare his desired service to be placed on his device. In addition, by analyzing the service usage of the fog device owners, the fog containers are placed where the workload is proportional to the usage of the owners while at the same time minimizing power consumption.

C. ORGANIZATION

The remainder of this paper is organized as follows. Section 2 presents related work, and Section 3 describes the architecture based on the fog portal and its detailed operations. Section 4 introduces the fog container placement optimization, and Section 5 demonstrates this through simulations. Section 6 concludes the paper.

II. RELATED WORK

Since the introduction of fog computing concepts, studies related to fog computing have been actively conducted in recent years. In the early years of research, studies were mainly conducted on the usability of fog computing; the focus of research subsequently shifted, such that resource management and architecture are current areas of research.

The availability of fog computing has been suggested primarily as the wording that this concept can create a new form of IoT service. It is referred to as an enabler of a specific service in addition to being able to simply improve the performance of the existing service or to provide real-time service. Based on geo-distribution, which is one of the most prominent features, the possibility of location-based advertising and providing shopping center maps is mainly raised [16]. Owing to the real-time services that fog computing can provide, it is also possible to delegate the computationally intensive module of the augmented reality application to the fog device or to provide quick response of the healthcare application [17], [18]. In more technical terms, it is treated as the controller of access mode of the radio access network, or as the core technology of 5G [19], [20]. In [21], a study was conducted to maximize the utilization of fog computing, such as telemedicine service, which provides rapid response through data collection, analysis, and pattern matching in fog containers, and discovers pattern through big data analysis in the cloud. While these studies have focused primarily on new application processes, fog computing was treated only conceptually.

Fine-grained design for management of fog device resource is essential because fog infrastructure comprises heterogeneous resources in computing and networking.

Nishio *et al.* [22] proposed a mathematical framework and architecture for heterogeneous resource sharing based on service-oriented utility functions, and optimized service delay. It is necessary to consider heterogeneous resources in the development of a fog container. However, the differential contributions of users and the provision of services as rewards for that were not considered. Hong *et al.* [23] proposed a simplified programming abstraction called PaaS programming model that can orchestrate highly dynamic heterogeneous resources at different levels of network hierarchy and support low latency and scalability. Research on fog device selection scheme has been actively conducted, and Arkian *et al.* [24] introduced a system called MIST that optimizes the matching of sensors and specific fog devices for crowd sourcing. In addition, there are studies on fog device selection schemes that can reduce carbon dioxide gas emissions on the green technology side, and studies that allocate resources to preferentially use eco-friendly energy-based data centers [25], [26]. These studies have analyzed key elements to fog computing through an in-depth approach, but assumed that the fog device was already widely deployed and all its resources can be controlled overall. In other words, the optimization of fog container placement was described entirely from a network point of view, which is not suitable for the user-driven fog infrastructure construction model presented in this paper.

Although the above studies approached the issues from a specific application point of view, there are limitations that do not reflect the characteristics of various services. However, these studies become relevant when future fog computing ecosystem will be built in reality.

Research on the structure of fog computing and other considerations has also been actively conducted. Tang *et al.* [27] proposed a hierarchical distributed fog computing architecture for big data analysis of a smart city service. This four-layered architecture allocates roles by dividing existing fog layers into specific regions. There are also studies on visualization of application structure and the analysis of the pricing model for the use of heterogeneous resources [28], [29]. In addition, there is a study that shows the detailed structure of a fog device [30]. These studies have analyzed fog computing from a business perspective or presented a description of individual components; however, the feasibility of fog computing was not considered.

Yi *et al.* [31] mentioned ''Join fog computing with private local cloud at the edge'' as an interesting business model of fog computing and discussed accounting and billing in fog computing. In that model, end users lease computing and storage capabilities to the fog service provider to reduce the cost of ownership. Compared with the architecture proposed in this study, it is similar in that the user participates in fog computing for some benefit, but the purpose is different. It differs from renting surplus resources from their own equipment to the fog service provider to reduce the operating costs of a private local cloud in the above model and leaving the entire delegation by installing a fog device to receive the desired fog computing services from the proposed model. In the former, similar to grid application, an ISP, a mobile carrier, or a cloud provider pays end users and leases resources to run services desired by the provider. On the other hand, in the latter, the users take advantage of the fog service they desire by providing the fog device to an independent entity, such as a fog service broker. In addition, detailed control schemes for the proposed system will be presented and we will optimize fog container placement.

III. USER-PARTICIPATORY FOG COMPUTING ARCHITECTURE AND ITS MANAGEMENT PROCESSES

The model in which the proposed architecture is built is similar to the expansion of Wi-Fi infrastructure. In the expansion, users install Wi-Fi APs in their homes to receive Wi-Fi services, and enterprise network administrators build infrastructure by installing multiple Wi-Fi APs in the office or building. Similarly, in the proposed architecture, users install fog devices to benefit from their desired fog services, and the local fog manager must operate fog container placement to realize this.

The existing realization method of fog computing is achieved through collaboration between cloud operators, IoT service operators, network infrastructure providers, and switch vendors. However, in this method, when an IoT service operator wants to provide a fog service in a specific area, it is unclear how the specific area should be determined, who will provide traffic usage information, which device of the specific vendor will be used, which network of the network infrastructure provider will be used, and who will be able to route traffic to the fog container?

In this paper, we propose a user-participatory fog computing architecture based on the fog portal to answer the above questions. Fig. 2 shows the conceptual diagram of the proposed architecture. The fog portal is a server located on the Internet, and it shares information with the fog manager and SDN controller located in each local network. The fog container placement is handled by the fog manager within the local network, and a fog device such as a switch, hub, Wi-Fi AP, and IoT GW is directly installed by an individual user or enterprise network administrator who wants to benefit from desired fog services. The fog device can also be installed by the IoT service providers who desire to provide high quality services. The device owner connects the device to the local network and registers the device information in the fog portal. At this time, the device owner can declare the desired service as compensation for participation. In summary, the fog portal performs intermediation globally between the device owner and the service operator, and the fog manager carries out the fog container placement.

The fog manager is located within the local network and monitors and controls the computing resources of the fog devices in the network. That is, the fog manager tracks the resource usage of all containers in the network, maintains the available resource of each device, and periodically reports it to the portal. In other words, similar to the SDN controller,

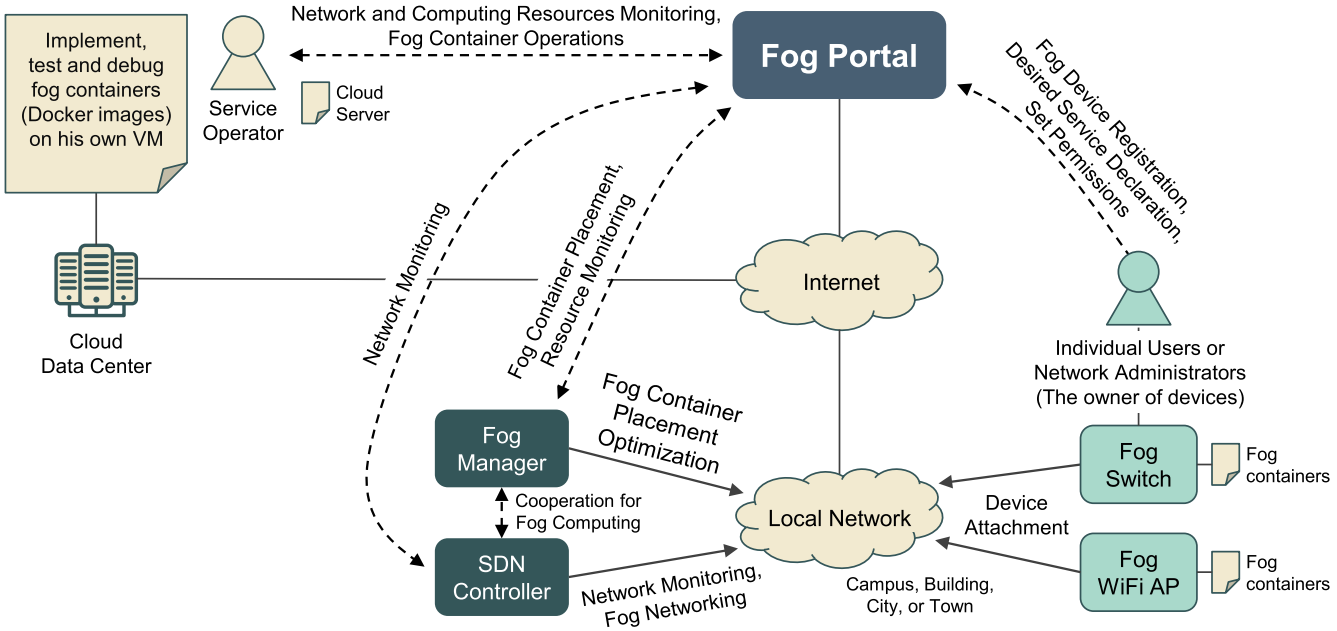

FIGURE 2. Overview of the fog portal-based, user-participatory fog computing architecture and its control flows.

the fog manager is a local entity that has control over computing resources in the network.

The fog portal also analyzes and processes detailed statistical information gathered from the SDN controller and provides such information to the appropriate service operator. The service operator determines the necessity of fog service through the user interface provided by the fog portal. If it is determined that the fog service is needed, the service developer implements the fog container in the form of a Docker image, and the implemented fog container is deployed by the fog portal. In other words, if a service operator wishes to deploy fog containers in a specific area, he can easily make the deployment request to the specific area through the interface provided by the fog portal. The operations, such as container development, placement, and migration are performed based on Docker.

A. ENGAGING IN USER PARTICIPATION

In the proposed model, user participation is achieved by directly purchasing a device, connecting to the network, and then registering the device in the fog portal. This subsection describes the content related to user participation.

The user purchases the fog device directly and connects it to his home network or building network. This is the same as purchasing and installing a Wi-Fi AP. The fog device installed at this time should be a network device capable of operating a fog computing service. In other words, Docker-based container management and SDN-based networking should be possible and equipped with standardized hardware, such as a CPU, memory, and storage, capable of running fog services. The standardization of fog devices has not been implemented yet, but the above hardware are a natural component of configuring devices [32]. Importantly, it is essential for the fog device to have the ability to deliver control messages from the fog portal or fog manager to its internal Docker manager or SDN agent.

The user that participates in fog computing must register his fog device in the fog portal after connecting it to the network. When the device is connected to the network, it is possible to register itself in the portal, and it is also possible to register directly using the user interface (UI) provided by the portal. However, the service declaration and permission setting to be described later must be performed by the user. During registration, information and capabilities of the device may be required, such as the number of cores, CPU clock, memory size, storage size, network capacity, MAC addresses, and device type. Among these, information related to computing capabilities may be in the database of the portal, along with the device model number.

Obviously, the user that participates in fog computing should be rewarded. The core of the proposed model is to make the user expect performance enhancement of the desired services. The user declares the desired services at the time of registering the device or thereafter. At this time, it is assumed that the user already knows through promotions what services are provided in the fog service. When the user declares the desired service, the service provides the fog service by installing a fog container in the local network to which the user belongs. In addition, the participating user can set permission for his device as participating users or all users in the network, and if the participating user is a public

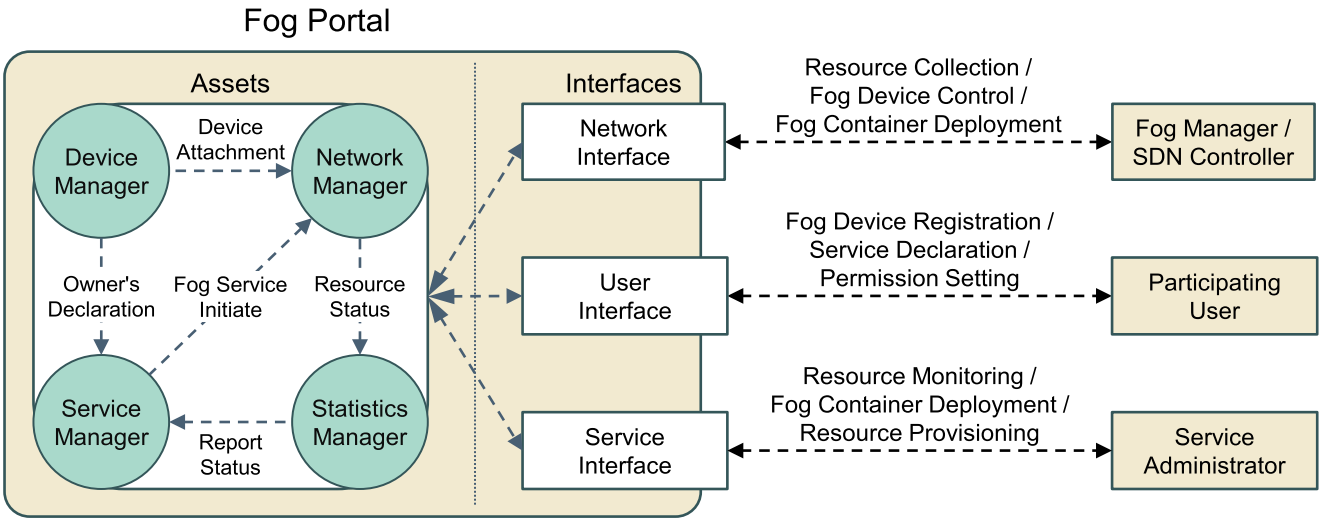

FIGURE 3. There are four asset managers and three interfaces of the fog portal. Each manager manages the assets such as fog devices, local networks, IoT services, and statistics, and each interfaces supports an API and UI for local fog managers, SDN controllers, participating users, and service administrators.

service provider, it is possible to provide public fog service to all users.

These user declaration can be applied for various IoT services. If the user is an individual, he can declare a smart home service and utilize his fog device as a smart home hub, or can declare a healthcare service to access his personal health data quickly and safely. If the user is a building manager, he can declare a smart building service to realize the corresponding functions by using each installed sensor, or can enable the shopping center map or the location-based advertisement service by declaring these services. In addition, urban infrastructure managers can widely deploy and register fog devices, and then declare smart traffic, autonomous vehicle support, and smart city services.

Direct monetary rewards can also be sufficient motivation for participation. Fog computing is an intertwined architecture of various business operators; thus, if the incentive is paid to the participating users, the incentive payment relationships can become complex. Infrastructure operators such as ISPs, cloud businesses, and the portal will pay incentives as compensation for their participation, and the payment criteria can be determined in a way that does not exceed the reduced OPEX. In addition, participating users may receive fees by leasing device resources to other non-participating users. The fog portal can play a key role in this brokerage process because the fog manager provides a detailed view of the resource provisioning and usage of the users. Above processes and models are covered in detail in other studies in our lab.

B. ROLES OF THE FOG PORTAL

The fog portal plays a pivotal role in the proposed architecture. Fig. 3 shows the conceptual internal structure of the portal. The resources managed by the portal are networks, devices, services, and statistics. The roles of the portal are to provide UI to the users and service operators for registering or deployment, to manage the area or computing capabilities of the registered fog devices, to provide traffic statistics to the appropriate services, and to collect and manage local network information by communicating with the fog manager and SDN controller.

1) FOG DEVICE MANAGEMENT

Several types of network devices can be fog devices and can be registered if they have the required capabilities, such as computing resources, Docker, and SDN. The portal should maintain information related to capabilities and types of registered devices. In addition, it is necessary to keep owner information of the device, and the relationship between a device and the owner can be one-to-one, one-to-many, many-to-one, and many-to-many. In other words, one user may install multiple devices or multiple users many share one device.

2) LOCAL NETWORK MANAGEMENT

The portal manages a large number of local networks at a single logical point, and mediates resources between services and the local networks. Local network information is collected from the local SDN controller and the fog manager. The portal maintains approximate location information for each local network in order to provide location information to the services. There may be questions about what to do if there is no SDN controller or fog manager in the network. Note that the proposed architecture is user participatory, and that the resources of the fog devices are delegated to the portal. If it is the first registered device in the network, the portal can make the fog manager work on this device. Similarly, if the SDN controller does not exist, the portal can directly install and

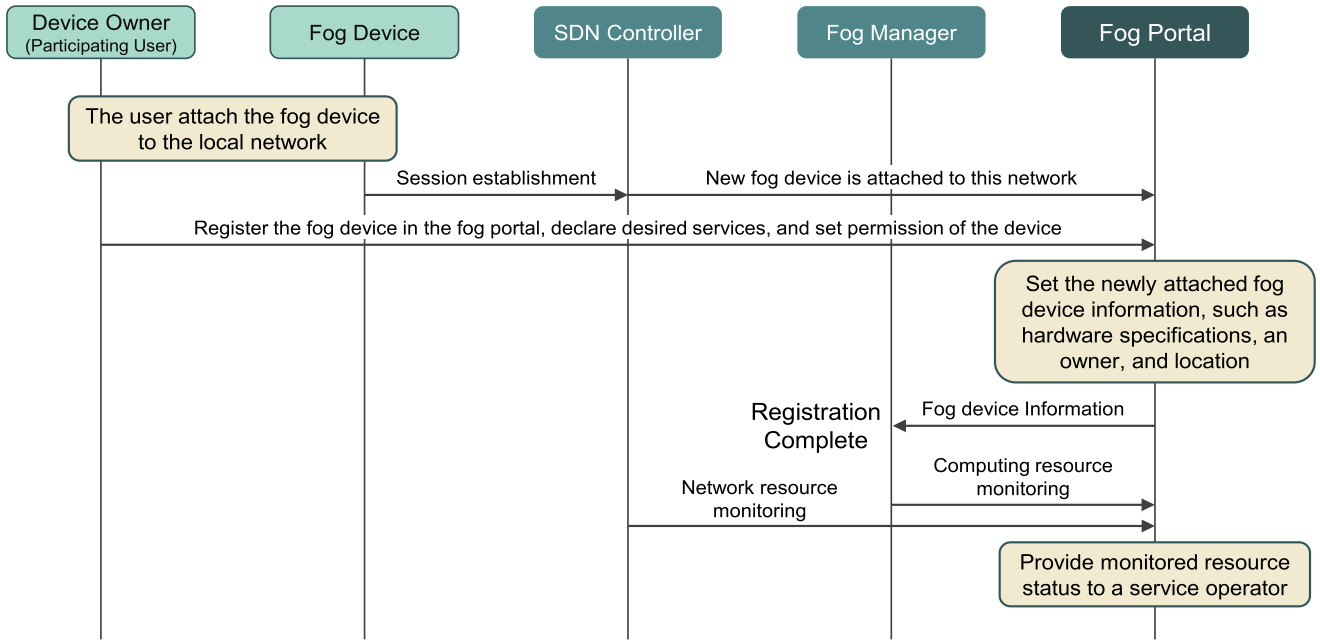

FIGURE 4. Fog device registration process.

operate the controller within the device. In this case, there is a high possibility that there is no SDN-enabled switch in the network, so the controller installed by the portal performs only limited functions, such as statistics gathering.

3) IoT SERVICE MANAGEMENT

The service administrator may be faced with the requirement to install fog containers in a specific area by making decisions based on network statistics provided by the SDN controllers or their own network analysis tools, or other operational policies. As in the case of the cloud, the portal enters into a resource lease agreement with the service operator in order to allow the service operator to lease and utilize some of the resources of the fog devices. After the resources have been allocated to the service as stipulated by the agreement with the portal, the operator can simply select a specific area through the intermediary interface provided by the portal. Alternatively, the service may be configured to automatically deploy its prebuilt containers when user declaration occurs.

4) STATISTICS MANAGEMENT

The portal collects service-specific statistics for each local network and provides such information to the service. Providing analytics and processed statistics to the service can help the service administrator to make resource allocation more sophisticated.

C. FOG PORTAL-BASED CONTROL AND MANAGEMENT PROCESSES

Fig. 4 shows the fog device registration process. The device owner purchases a fog device and connects it to the network. The local SDN controller establishes a session with the device as soon as it is connected to the network, and then reports it to the portal. Next, the user or device itself registers the device with the capabilities, approximate location information, service declaration, and permission in the portal. The portal uses information from the SDN controller and registration information to determine how and where the device configures the topology, and delivers this information to the fog manager of the corresponding network. Through this process, the fog manager can grasp topological information and computing capabilities of the fog device and determine the placement position based on information when the fog container is placed. After the fog device is registered in the portal, the SDN controller and fog manager periodically provide the monitored resource of the device to the portal.

Fig. 5 shows the service admission mediation process by decision of the service operator. The portal collects monitored resources of all devices from the SDN controller and fog manager of each network. The service operator can ascertain the usage of his or her own service and can decide to start fog service within the area if the popularity of the service in a particular area exceeds his or her placement threshold. The service operator develops a fog container in the form of a Docker image, and then determines its deployment target area through the portal.

The portal instructs the fog manager located in the selected area to choose the appropriate fog device to place the fog container of the service. The fog manager determines the possibility of admission of the service based on the resource status in the network. If a new service cannot be entered due to insufficient available resources, the resource allocation amount of the currently operating services is adjusted to the predetermined ratio. If this adjustment is not possible, the fog

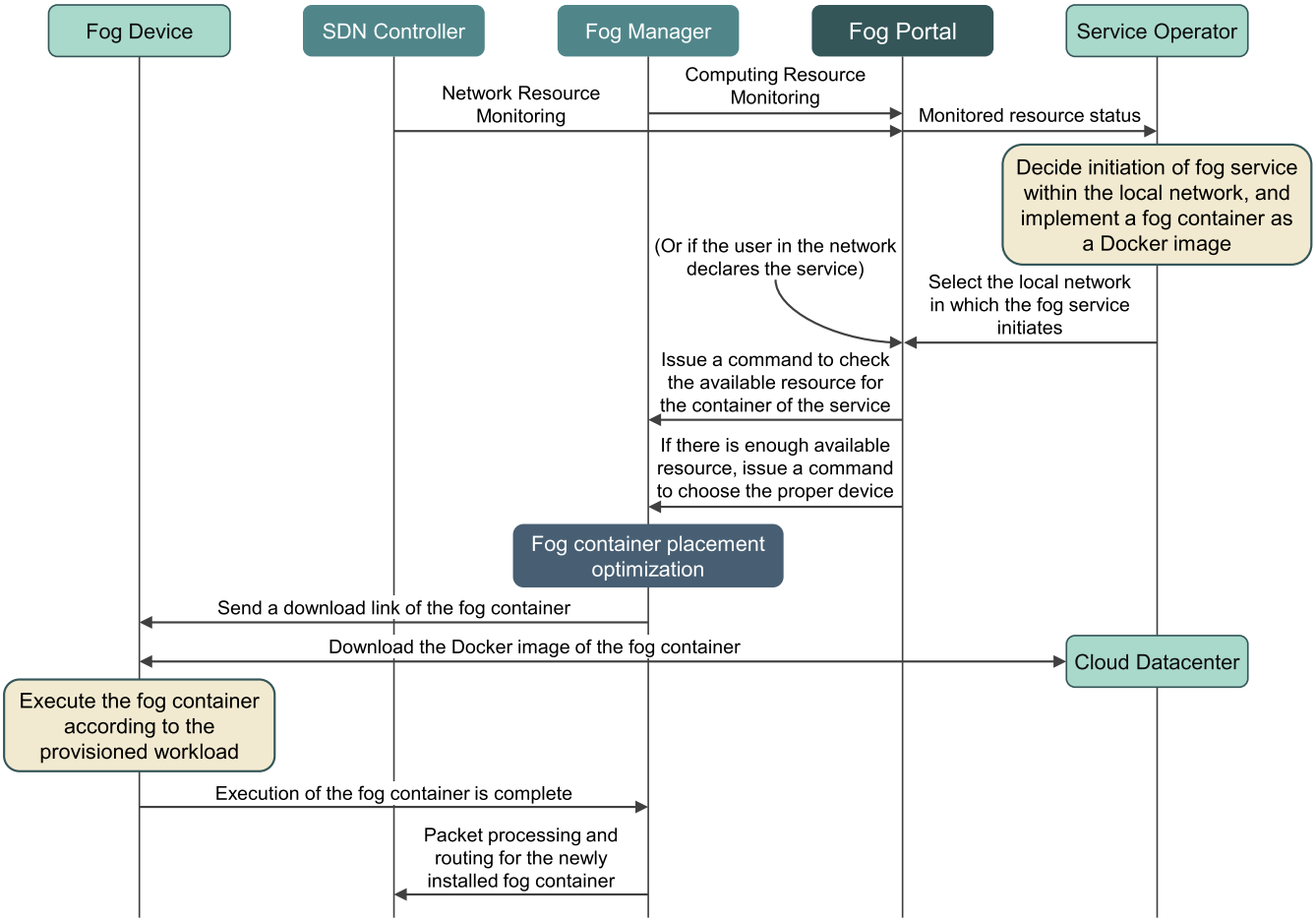

FIGURE 5. Fog service admission mediation process.

manager responds to the portal that the service cannot enter the network.

If the service can enter, the fog manager chooses the device through container placement optimization. The fog manager delivers a link to download the fog container image in the form of a Docker image to the selected device, and the device downloads the image from the link using its own Docker manager. When the download is complete, the Docker manager in the device sets up the operation resources of the containers based on resource provisioning information agreed in advance between the service operator and the portal, and creates both the application container and the virtualized network function containers, such as a database, load balancer, and firewall. When the container execution is complete, the fog manager sends a request to the SDN controller to perform packet processing and routing to allow the service traffic to be diverted to the target device.

Of course, the initiation of fog services in the network for new services is possible not only by the decision of the service operator, but also through user declarations. The user can declare the desired service when joining the network or while using the network, and it is assumed that the operator of the declared service has already agreed the resource lease with the portal. Service admission by user declaration is the same as the process by the decision of the service operator.

D. SCALABILITY AND RELIABILITY ISSUES

The fog portal is logically centralized and can exist in a physically distributed system for scalability. Fortunately, scalability is not a big consideration when it comes to managing two typical roles of the portal—fog device registration and service admission mediation.

Unlike the portal, the fog manager exists in each local network and performs computational resource monitoring and fog container placement optimization. The fog manager may be disconnected from the fog devices or the portal owing to failure of the network or link. Despite this disconnection, the containers in the fog devices can continue to provide services and only actions, such as container placement optimization are temporarily suspended.

The failure of the fog device can cause internal containers to face a crisis. The application container that provides the functionality of the service itself is responsible for responding to requests from specific users or IoT devices or providing local information on the service. That is, the users or the IoT devices that have been provided with the fog service from

the container of the failed fog device may experience service suspension. The fog manager can restart the fog service in a short period of time by updating the association relationship between the container and the users as soon as device failure is detected. Also, in the case of the stateful service, the failure can be overcome by maintaining a redundancy server or checkpointing to neighboring devices or neighboring local networks.

IV. FOG CONTAINER PLACEMENT BASED ON SERVICE USAGE OF PARTICIPATING USERS

A. CONSIDERATIONS AND PROBLEM STATEMENTS

The basic concept of the proposed architecture is that users participating in the network with their own devices declare a specific service, and the fog service is provided by placing the fog container of declared service in the local network. As described above, the initial development of the fog container and the agreement with the fog portal are determined by the service operator, and the subsequent admission process is performed according to the user declaration or the decision of the service operator. The fog container placement process is performed by the local fog manager at the time of service admission or during operation, with the following additional considerations.

The first consideration is that the reliability of user declarations is not expected to be high-level. In other words, regardless of whether a user is an individual or a corporation, the service declaration is a privilege and a benefit given by participating in the fog infrastructure, but this is not mandatory. For example, the user can use the fog service without declaring any service, and can mainly use the specific service without declaring the corresponding service, and the declared service and the actual service being used can be completely different. In fact, these situations can occur frequently because the users will not update the declaration whenever the network usage pattern changes. In the proposed architecture where the user declaration is optional, a periodic automatic fog container placement must be performed to accommodate situations in which services actually used by the users change over time.

Second, even if the container is placed somewhere in the local network, it does not affect user experience. The most important features of fog computing that enhances user experience are user location consideration and the reduction of latency. A local network with a 24 bits subnet prefix is not geographically broad, unlike a WAN or the Internet. This means that the location of the user can be figured out roughly regardless of where the container is placed in the local network. The other is that the round trip time in the local network is not so long. The latency of the Internet can range from a few tens of milliseconds to a few seconds that people can perceive, while the latency of the local network is only a few milliseconds. This means that even if the container is placed on a device farthest from the user in the local network topology, the user only needs to wait a few milliseconds to use the service, which is not critical.

The third is that most response-oriented services need not to split request from one user into several containers. Especially, if a particular API request requires authentication or authorization, it is reasonable to be processed in a single container for a single request. For example, services, such as healthcare, smart home, autonomous vehicle, telemedicine, and video streaming are accompanied by server responses rather than processing data, such as filtering data at the fog layer for big data processing, and these responses may require appropriate authentication or authorization. Therefore, it is assumed that the requests from one user are processed by one container in this architecture.

Fourth, services are configured to support scalability through flexible resource management within each local network. In a network, each service can operate across multiple devices, and flexible resource management is achieved based on the execution of multiple identical containers. A container providing a major function of a service is called an application container, that is, a service can consist of a plurality of application containers. In addition, a service can include network functions, such as a database, a load balancer, and a firewall according to its operating policy. Each function is a VNF instead of separate hardware, and each VNF operate as a Docker container.

Fig. 6 shows the container status when two services in the network operate across multiple devices. If the service uses a database, a load balancer, a firewall, and an authentica-

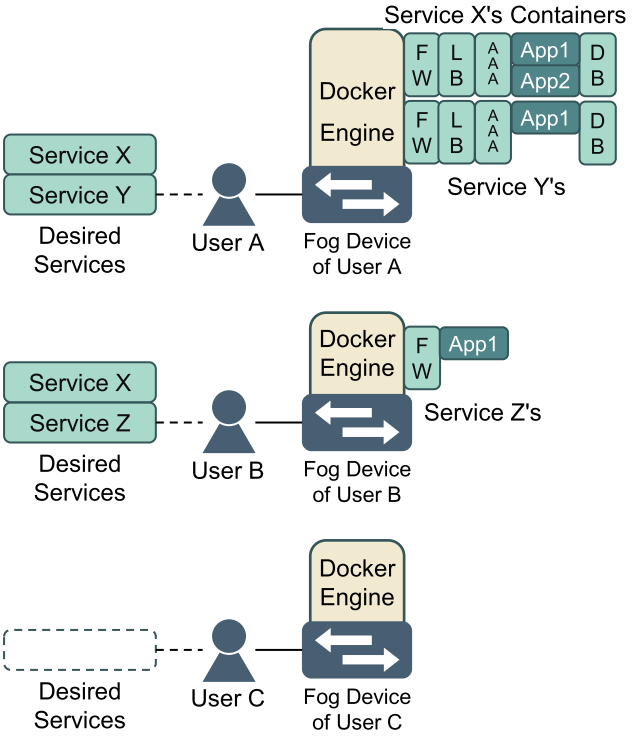

FIGURE 6. Example of container placement. When user A declares service X and Y and user B declares service X and Z, the fog device of user A operates the containers of service X and Y, and the fog device of user B operates the containers of service Z.

tion, authorization, and accounting (AAA) server, the service operates in a single device with four VNF containers and at least one application container. Whenever the number of users using the service increases, or when all requests cannot be processed in one application container by increasing the load of the service, the application container is further executed. At this time, all service traffic to the service goes to the application container through the load balancer, the firewall, and the AAA server container, and all application containers share one database container.

The last consideration is that the fog manager operates and manages fog devices based on leasing the operating rights to the fog portal, and the assets themselves are owned by the users. Therefore, the perspective of the container placement performed by the fog manager should center the individual devices of the users rather than the entire local network. The fog manager should place the container so that the device of the user has a result similar to the expected power consumption based on the usage of the owner. In other words, it is not reasonable to pay much more than what the owner used by running the containers for other users. For example, if the network efficiency is considered for placing the containers, when all participating users in a local network require one service with low network cost and high workload, the containers of the service will be located only in the topological centermost device and operate at 100% load for 24 hours. The power consumption of the device at this time is obviously unreasonable for the owner of the device. Therefore, fog container placement in the local network should be performed to minimize the power consumption, while simultaneously considering the usage of the fog service of the users as a priority.

B. SYSTEM MODEL AND CONSTRAINTS

This section introduces the system model and constraints of the container placement of the proposed architecture. The notations used in this paper are described in Table I.

The user set U means all participating users in the network. The user is not a one-to-one relationship with an actual user, and refers to a host or an IoT device connected to the fog device to generate traffic. That is, all traffic generated under the specific device is traffic generated by the owner of that device. The service set S includes services that are declared or used in the network and considered to require container placement. The fog device set D means all devices making up the network.

The service operates in a container form across multiple devices in the network. The portal allocates the resource requested by the service, and the resource is considered to be the total workload used in each local network. Each service consumes the allocated workload across several devices in the network, and the assignment variable at the device d relative to the total workload of the service s can be expressed as follows:

$$\alpha_s^d \in [0, 1], \quad \forall s \in S,\ d \in D \tag{1}$$

TABLE 1. Parameters for the system model.

Symbol	Description
U	the set of participating users in the network
S	the set of services in the network
D	the set of fog devices in the network
T	the set of traffic related to services from users
α_s^d	the assignment variable, an allocation proportion of a device d for a service s
β_u^{sd}	the selection variable, which indicates whether a container of a service s for a user u is on a device d or not
τ_u^s	the traffic generated by a user u for a service s in a unit time
ω_ρ^s	the provisioned workload for a service s
C_d	the workload capacity of a device d
γ_a^s	the workload per traffic of the application container of a service s
γ_f	the workload per traffic of a VNF container
N_f^s	the number of VNFs of a service s
ε_B^d	the power efficiency of a device d for loading a container including the idle state
ε_A^d	the power efficiency of a device d for operating a container in the active state
ε_N^d	the power efficiency of a device d for transmitting data
e_d	the power consumption of a device d
$\hat{e}_d$	the expected power consumption of a device d

For all services, the sum of the ratio assigned to each device is 1, which can be expressed as:

$$\sum_{d \in D} \alpha_s^d = 1, \quad \forall s \in S \tag{2}$$

The assignment variable is set so that it does not exceed the workload capability of each device for all devices, and can be expressed as follows:

$$\sum_{s \in S} \alpha_s^d \omega_\rho^s \le C_d, \quad \forall d \in D \tag{3}$$

The workload capacity constant C_d depends on hardware specification of the device, and this can be calculated by the portal when registering the device.

As assumed previously, one request does not have to be processed by multiple containers, so the service request or communication of one user is directed to a specific container in any device in the network. In other words, the user can use several services, but the traffic for the specific service of the user is processed in only one container. The selection variable, which indicates that the traffic for the service s generated by the user u is directed to device d, can be expressed as:

$$\beta_u^{sd} \in \{0, 1\}, \quad \forall u \in U,\ s \in S,\ d \in D \tag{4}$$

If β_u^{sd} is set to 1, it means that the traffic related to the service s of the user u should be sent to the device d, and if β_u^{sd} is zero, it means it should not be sent.

The workload refers to the average usage amount of the CPU, the memory, and the I/O work that a particular container uses per unit time. In this paper, we expressed it as a normalized value to avoid excessive complexity. There are other studies that present resources used by processes as workloads [33]. As described above, a service consists of VNF containers used by a service and application containers in one device. The workload of an application container has a different value depending on the type of service. For example, if a service only stores data, the workload will be low level, and if a service performs CPU-intensive real-time video processing, the workload will be quite high. Therefore, the workload of an application container is different for each service and can be expressed as a constant value γ_a^s according to network input data amount. On the other hand, in the case of a VNF container, the workload for input data amount can be represented by a constant value γ_f since there is no significant difference between the services.

The traffic usage constant $\tau_u^s \in T$ represents the traffic generated per unit time by the user u communicating with the container of the fog service s in the network. User generated traffic has characteristics that occur within the range of the workload of a specific service, which is controlled naturally by the existing network stack. The selection variable β_u^{sd} should be set only for the user that generates traffic for a specific service, and can be expressed as follows:

$$\frac{\tau_u^s}{\|T\|} \leq \sum_{d \in D} \beta_u^{sd}, \quad \forall u \in U,\ s \in S \tag{5}$$

Furthermore, since there is only one target container for the service that the user uses, the following constraint is defined:

$$\sum_{d \in D} \beta_u^{sd} \leq 1, \quad \forall u \in U,\ s \in S \tag{6}$$

For all services, the sum of workloads due to traffic generated by all users for a particular device is less than the workload assigned to that device by the assignment variable. This is because the traffic generated by the user for a specific service is processed in only one container, and the following relationship holds.

$$\sum_{u \in U} \beta_u^{sd} \tau_u^s \left(N_f^s \gamma_f + \gamma_a^s \right) \leq \alpha_s^d \omega_\rho^s, \quad \forall s \in S,\ d \in D \tag{7}$$

C. PROBLEM FORMULATION

In the proposed architecture, the fog manager places the containers to minimize the power consumption owing to fog computing in the network considering the service usage by users. The power consumption of each device can be defined as follows:

$$e_d = e_B^d + e_A^d + e_N^d, \quad \forall d \in D \tag{8}$$

The power consumption per unit time of the device is the sum of the base power consumption of the containers, the power consumption of the containers in active state, and the power consumption for communication. e_B^d is the power consumed by each container in the device, and is defined as follows:

$$e_B^d = \varepsilon_B^d \sum_{s \in S} \left\lceil \alpha_s^d \right\rceil \left(N_f^s + \sum_{u \in U} \beta_u^{sd} \right), \quad \forall d \in D \tag{9}$$

where N_f^s represents the number of VNFs of the service s, and ε_B^d represents the power efficiency of the device d for holding the containers, which include the idle state. Note that the device type depends on the user selection, so the power efficiency of the installed device will be different. The number of application containers also depends on the number of users requested.

The power consumption of the active mode containers is defined as follows:

$$e_A^d = \varepsilon_A^d \sum_{s \in S} \sum_{u \in U} \beta_u^{sd} \tau_u^s \left(N_f^s \gamma_f + \gamma_a^s \right), \quad \forall d \in D \tag{10}$$

The power consumption is determined by the sum of the workloads of the VNF containers of the service s in the device and the application containers, where ε_A^d represents the power efficiency of the device d per workload, which also has a different value for each device. The power consumption for networking can be defined as follows:

$$e_N^d = \varepsilon_N^d \sum_{s \in S} \sum_{u \in U} \beta_u^{sd} \tau_u^s, \quad \forall d \in D \tag{11}$$

If there is no control by the fog manager, the users will run the containers of the services they use on their device. The expected power consumption at this time can be expressed as follows:

$$\hat{e}_d = \hat{e}_B^d + \hat{e}_A^d + \hat{e}_N^d, \quad \forall d \in D \tag{12}$$

As in (8), it consists of the base power consumption of the containers, power consumption of the active mode containers, and the network power consumption. However, the expected selection constant $\hat{\beta}_u^{sd}$ used in calculating the expected power consumption is set in advance so that the containers of the services used by the users operate in their own device. Therefore, the components of (12) are defined as follows:

$$\hat{e}_B^d = \varepsilon_B^d \sum_{s \in S} \sum_{u \in U} \hat{\beta}_u^{sd} \left(N_f^s + 1 \right), \quad \forall d \in D \tag{13}$$

$$\hat{e}_A^d = \varepsilon_A^d \sum_{s \in S} \sum_{u \in U} \hat{\beta}_u^{sd} \tau_u^s \left(N_f^s \gamma_f + \gamma_a^s \right), \quad \forall d \in D \tag{14}$$

$$\hat{e}_N^d = \varepsilon_N^d \sum_{s \in S} \sum_{u \in U} \hat{\beta}_u^{sd} \tau_u^s, \quad \forall d \in D \tag{15}$$

The fog manager places the containers based on the service usage amount of the user. Considering the usage is equivalent to considering the expected power consumption in (12); that is, the power consumption of each device should be determined based on the expected power consumption. Therefore, the following additional constraint should be applied for all devices.

$$e_d \leq \kappa \hat{e}_d, \quad \forall d \in D,\ \kappa \in [1, 2] \tag{16}$$

where κ is the tolerance constant, which determines how much additional power consumption ratio the user can tolerate based on his service usage. If κ is set to 1, it indicates that power consumption of the fog devices should not to be more than the power consumed by the service usage of their owners, and if κ is set to 1.2, it means allow additional power to be consumed in a range not exceeding 20% of the power used by the service usage.

The objective function for the container placement is defined as follows:

$$f\left(\alpha_s^d, \beta_u^{sd}\right) = \sum_{d \in D} e_d, \quad \forall u \in U,\ s \in S \tag{17}$$

Equation (17) minimizes the sum of power consumption of all devices in the network according to the assignment variable α_s^d and the selection variable β_u^{sd}. This optimization problem is mathematically formulated as follows:

$$\begin{aligned} &min\, f\left(\alpha_s^d, \beta_u^{sd}\right), \\ &s.t.\ (1), (2), (3), (4), (5), (6), (7), (16), \\ &\quad \forall u \in U,\ s \in S, d \in D \end{aligned} \tag{18}$$

Equation (18) is a mixed-integer non-linear programming (MINLP) problem, requiring variable relaxation [34].

D. RELAXING VARIABLES

To linearize (18) in (9), the ceiling function for α_s^d and the product term $\lceil \alpha_s^d \rceil \sum_u \beta_u^{sd}$ should be removed. First, the complexity is further reduced by changing the assignment variable α_s^d to be dependent on the selection variable β_u^{sd} and the traffic vector τ_u^s. The containers can be placed in several fog devices according to the user traffic and the selection variable, which is represented as follows:

$$\sigma_u^s = \sum_{u \in U} \tau_u^s, \forall s \in S \tag{19}$$

$$\alpha_s^d = \begin{cases} 0, & if\ \sigma_u^s = 0 \\ \sum_u \tau_u^s \beta_u^{sd} / \sigma_u^s & otherwise \end{cases}, \quad \forall s \in S,\ d \in D \tag{20}$$

By redefining the assignment variable α_s^d with the selection variable β_u^{sd} and the traffic vector τ_u^s, α_s^d becomes just a coefficient. Also, the constraints related to α_s^d can be eliminated, which are (1), (2), and (7).

Next, the ceiling function for α_s^d should be linearized. The linearization is performed by defining a new auxiliary variable and replacing the ceiling function with that variable. The auxiliary variable is defined as follows:

$$\zeta_s^d = \left\lceil \alpha_s^d \right\rceil, \quad \forall s \in S,\ d \in D \tag{21}$$

And this auxiliary variable ζ_s^d also has the following constraints to linearize the ceiling function.

$$\zeta_s^d \in \{0, 1\}, \quad \forall s \in S,\ d \in D \tag{22}$$

$$\zeta_s^d \geq \alpha_s^d, \quad \forall s \in S,\ d \in D \tag{23}$$

$$\zeta_s^d < \alpha_s^d + 0.9999, \quad \forall s \in S,\ d \in D \tag{24}$$

Finally, linearizing of the product term can also be achieved by defining another auxiliary variable. The auxiliary variable is defined as follows:

$$\eta_u^{sd} = \zeta_s^d \beta_u^{sd}, \quad \forall u \in U,\ s \in S,\ d \in D \tag{25}$$

And this auxiliary variable η_u^{sd} has the following constraints.

$$\eta_u^{sd} \in \{0, 1\}, \quad \forall u \in U,\ s \in S, d \in D \tag{26}$$

$$\eta_u^{sd} \leq \zeta_s^d, \quad \forall u \in U,\ s \in S,\ d \in D \tag{27}$$

$$\eta_u^{sd} \leq \beta_u^{sd}, \quad \forall u \in U,\ s \in S,\ d \in D \tag{28}$$

$$\eta_u^{sd} \geq \zeta_s^d + \beta_u^{sd} - 1, \quad \forall u \in U,\ s \in S,\ d \in D \tag{29}$$

By using these auxiliary variables, (9) is redefined as follows:

$$e_B^d = \varepsilon_B^d \sum_{s \in S} \left(N_f^s \zeta_s^d + \sum_{u \in U} \eta_u^{sd} \right), \quad \forall d \in D \tag{30}$$

The optimization problem (18) can be reformulated as an integer linear programming problem as follows:

$$\begin{aligned} &min\, f\left(\beta_u^{sd}, \zeta_s^d, \eta_u^{sd}\right), \\ &s.t.\ (3)\text{-}(6), (16), (22)\text{-}(24), (26)\text{-}(29), \\ &\quad \forall u \in U,\ s \in S, d \in D \end{aligned} \tag{31}$$

V. SIMULATION RESULTS AND PERFORMANCE EVALUATION

In this paper, we propose the fog portal-based, user-participatory fog computing architecture and its management schemes, and propose the container placement optimization according to the service usage of participating users. This section describes the simulation environment and the results analysis related to the power consumption of the devices of the users according to the container placement by the fog manager.

A. SIMULATION SETUP

The simulation was conducted by implementing an environment in MATLAB where one fog manager carried out container placement operations on a single local network. As mentioned earlier, the user directly installs the fog device in his home or office and accesses the network via that device. In the local network mentioned in this paper, latency effects were not significant enough to be perceived by the users anywhere in the network, because all the fog devices and routers were assumed to be connected via Gigabit Ethernet as in the typical configuration. Therefore, the simulation was conducted while assuming a full mesh topology environment.

The service may utilize multiple VNFs for that purpose. For simplicity of the environment, it was assumed that the VNF computation workload per traffic was the same. Obviously, the average generated traffic and the workload per traffic were service-specific, and this information can be obtained empirically or by learning through system monitoring. Since the simulation assumed the coexistence of multiple services, the services were necessary to properly categorize the average generated traffic, the workload per traffic, and

the total allocated workload. The average generated traffic per second of the service can be classified into 1, 50, and 1000 kbps. The workload per unit time (in second) can be similarly classified as 10, 100, and 1000. That is, the simulation assumed 9 types of services; for example, a service with an average traffic of 50 kbps and a workload per traffic of 100, the workload per 1 kbps of traffic, that is represented as γ_a^s, is 2. Each service has 1–5 VNFs, and these VNFs exist in the same number of containers for each device that operates the service. The allocated workload for the service depends on the average number of users, which use the corresponding service, expected by the service operator. It was assumed that the maximum number of users for the services with an average workload per traffic of 10, 100, and 1000 were 50, 30, and 10 users, respectively. In other words, the allocated workload of the service with an average workload per traffic of 10, 100, and 1000 were 500, 3000, and 10000, respectively.

The fog device can also consist of various models. In the simulation, it was assumed that the workload capacity of a device C_d is in the range of 20000–50000, the power efficiency associated with maintaining one container ε_B^d is in the range 0.3–0.7 W, the power efficiency per an active container ε_A^d is in the range 0.6 × 10-2–1.5 × 10-2 Watts per workload, and the power efficiency for networking ε_N^d is in the range 0.4 × 10-5–15.4 × 10-5 W/kb with reference to [35]–[37]. Naturally, in the real world, these values can be provided by hardware vendors, empirically or through learning.

In addition, the users were using approximately 20% of the services currently providing fog service on the network, and the tolerance constant κ was set to 1.2. The workload of the VNF containers per 1 kbps traffic γ_f was defined as 0.1. Unless otherwise noted, the number of services, devices, and users was 6 in every simulation.

B. SIMULATION RESULTS AND DISCUSSION

The simulation evaluated the power consumption reduction ratio by the container placement optimization. The power consumption reduction ratio (PRR) is expressed as:

$$PRR = \sum_d \left(\hat{e}_d - e_d\right) \Big/ \sum_d \hat{e}_d \tag{32}$$

First, the simulations were conducted to investigate the effect of workload capacity of fog devices on the system. Fig. 7 shows the effect of workload capacity of the device on PRR. In this case, the number of devices was 6, and the workload capacity of all devices varied from 10000 to 30000 in 1000 units. Each user randomly used 10, 20, 30, 40, and 50% services among 9 types of services per simulation. Obviously, the increase in the PRR owing to device workload capacity was limited for all service utilization rates. On the other hand, the increase in reduction ratio owing to the service utilization rate was conspicuous because the application containers of the corresponding service were likely to share the same VNF containers in a power efficient device when the usage of the specific service was high.

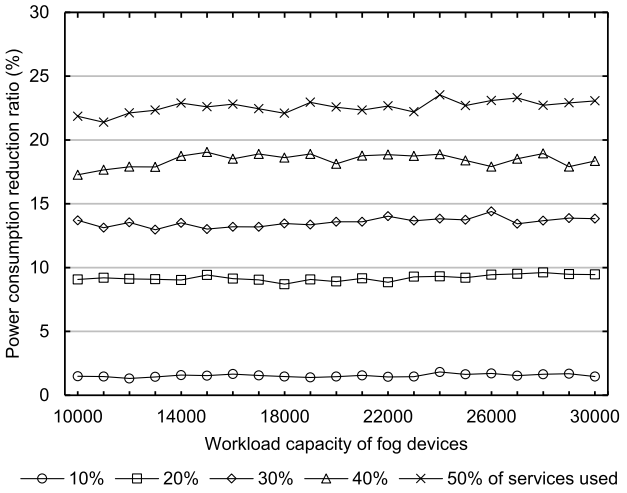

FIGURE 7. Power consumption reduction ratio according to workload capacity of fog devices for each service utilization rate.

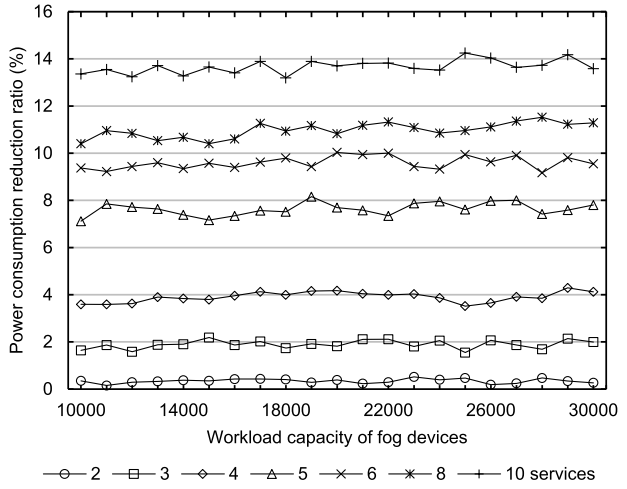

FIGURE 8. Power consumption reduction ratio according to workload capacity for each number of services.

Fig. 8 shows the effect of workload capacity of the device on PRR for each number of services. In this case, changes in the number of devices and the workload capacity were the same as in the previous simulation, the service utilization rate was fixed at 20%, and only the number of services varied from 2 to 10. In other words, this means that the number of services that are currently providing fog service in the network varied from 2 to 10. There was no significant change in the PRR depending on the workload capacity of the device, similar to the previous simulation. Although device workload capacity did not affect the PRR, an increase the number of services increased the PRR. That is, even if the service utilization rate was fixed at 20%, the service usage increased as the number of services increased.

The next simulations were conducted based on the number of fog devices in order to evaluate how many users were involved in infrastructure construction to achieve sufficient PRR. In the simulations, since the number of participating

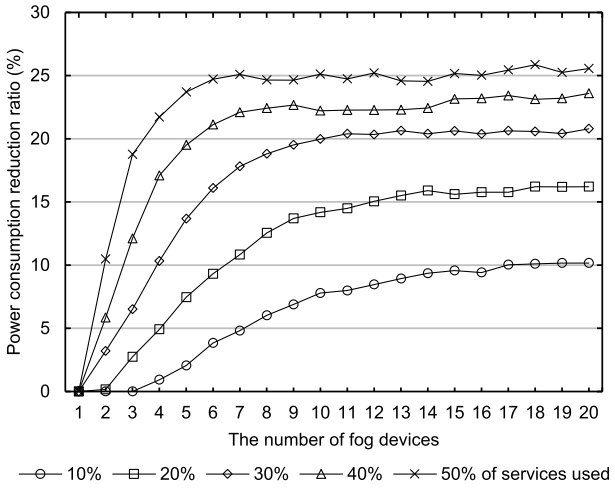

FIGURE 9. Power consumption reduction ratio based on the number of fog devices for each service utilization rate.

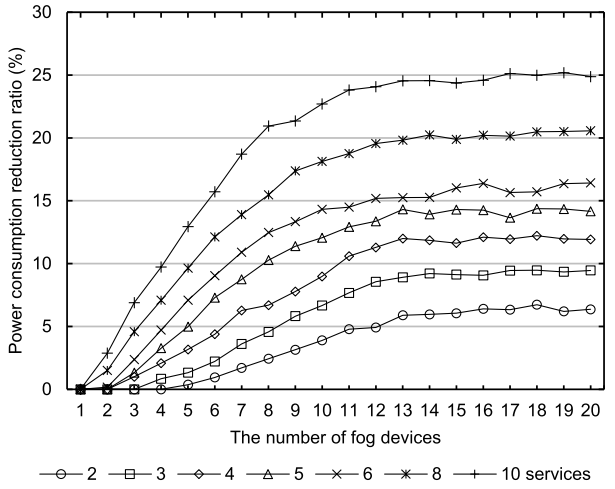

FIGURE 10. Power consumption reduction ratio based on the number of fog devices for each number of services.

users and the number of fog devices was one-to-one, the number of devices was the same as the number of users.

Fig. 9 shows the effect of the number of devices on the PRR for each service utilization rate. In this case, the number of devices varied from 1 to 20, and the hardware specification of the devices was selected randomly within the range mentioned in the simulation setup subsection. Six types of services were assumed and the service utilization rate was increased step by step from 10% to 50%. From the simulation results, the number of devices at the saturation point and the maximum saturated reduction ratio value varied depending on the service utilization rate. When the service utilization rate was 20%, the reduction ratio increased linearly until the number of devices was 14, and then it saturated to the level of the reduction ratio of approximately 16% at 14 and thereafter. On the other hand, when the service utilization rate was 50%, the reduction ratio increased sharply until the number of devices increased to 6, and thereafter, the reduction rate saturated to approximately 25%. The container placement optimization by the fog manager considered the service usage of individual users. The high service utilization rate indicates that the service usage of individual users was also high on the average, which means that the number of containers that can be efficiently operated on an individual device during the optimization process increased. As a result, the PRR increased with the service utilization rate.

Fig. 10 shows the effect of the number of devices on PRR for each number of services. In this case, the number of devices varied from 1 to 20, and the number of services entering the network varied from 2 to 10. The service utilization rate was fixed at 20%. Compared with Fig. 9, which is the simulation result on 6 services, Fig. 10 shows that the service utilization rate, rather than the number of services, improved the reduction ratio more steeply when the number of devices was little. It can be seen that the benefits of providing the same service to more users at the same time were greater than the benefits of providing various services to various users.

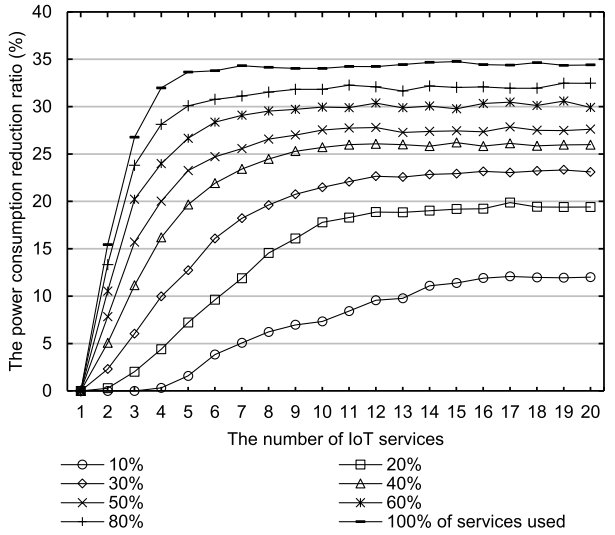

FIGURE 11. Power consumption reduction ratio according to the number of services for each service utilization rate.

Fig. 11 shows the impact of the number of services on PRR for each service utilization rate. In this case, the number of services varied from 1 to 20, and the service utilization rate increased gradually from 10% to 100%, and the number of devices was fixed to 6. 80% of service utilization rate means, for example, that when there are 100 services in the network, all users use 80 services simultaneously on the average. On the other hand, one service serves 80% of users on the average at the same time. The PRR increased significantly when the service utilization was high and the number of services was small. Also, it was confirmed that the PRR was saturated at the small number of services when the service utilization rate is high. On the other hand, when the service utilization rate was low, the PRR variation according to the increase of the number of services was relatively limited.

Fig. 12 shows changes of PRR according to the variation of the tolerance constant for each service utilization rate. The tolerance constant is a value that determines how much

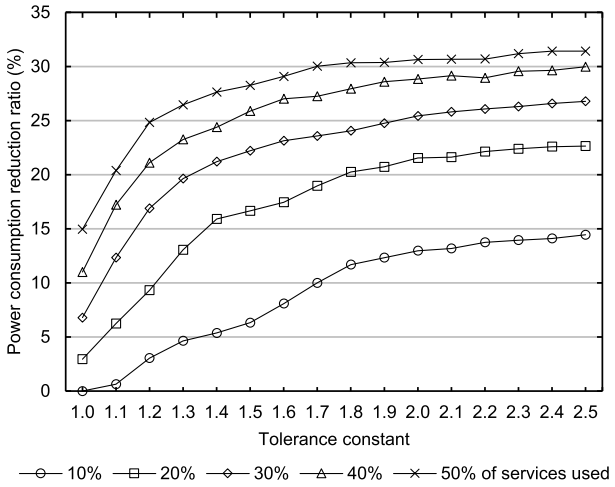

FIGURE 12. Power consumption reduction ratio owing to tolerance constant for each service utilization rate.

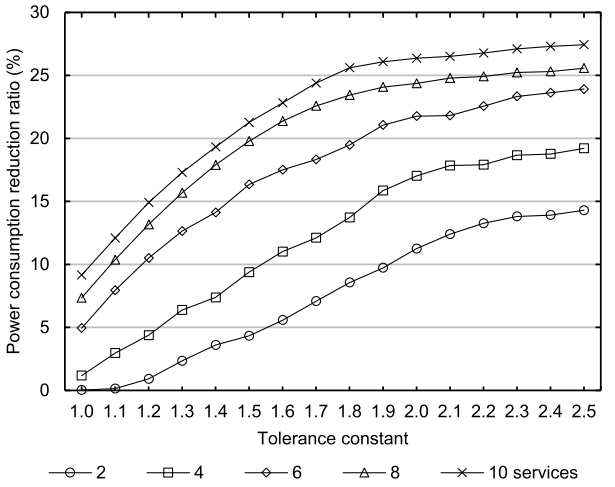

FIGURE 13. Power consumption reduction ratio owing to tolerance constant for each number of services.

additional power consumption ratio the user can tolerate based on his service usage. This can be also called the inequality index, because the power consumption of the user device becomes similar as the one used by the user as the tolerance constant is lower, and the higher tolerance constant, the higher possibility that the containers are intensively placed on the devices with higher power efficiency.

In this simulation, the tolerance value varied from 1.0 to 2.5. When the service utilization rate is low and the tolerance constant is smaller than 1.2, it shows a very low PRR. Also, in all service utilization rates, the change of PRR according to the constant value increases sharply in the beginning, and then gradually becomes slow. The low PRR when the utilization rate and the tolerance constant were low was because the power consumption benefitted only when the containers of the services used by users were placed in an interchangeable fashion. For example, when two users use the same two services, the power consumption gain occurs only when the two services are operated on one device, one by one. That is, in this situation, the mutual exchange of the services must occur in order to obtain the power consumption reduction. A fairly constant PRR even though the tolerance constant increases was the result of averaging the rates at which device-specific power consumption was reduced even though the situation that most containers operated in one device. That is, when the number of services, the number of devices, and the service utilization rate were fixed, the power consumption gain that can be obtained at the maximum in the entire network became a nearly constant value.

Fig. 13 shows the change in PRR according to the variation of the tolerance constant for each number of services. In this simulation, as in the previous simulation, the tolerance constant varied from 1.0 to 2.5. The PRR was approximately saturated at 14% when the number of service was 2 and the tolerance constant is above 2.3, and was saturated at 27% when the number of services was 10 and the tolerance value is above 1.8. These results were similar to the previous simulation, that is, from the two simulations it was evident that there was no reason for a user to be inequitable.

Fig. 14 shows the difference in power consumption according to tolerance constant for each participating user. In this simulation, the number of services was set to 6, the number of devices and users was set to 12, and the service utilization rate was fixed at 20%. The tolerance constant varied from 1.0 to 100.0. The leftmost bar for each user is the estimated power consumption owing to the actual service usage by the user. The user 3 installed a device with high power efficiency, and used more services than other users. As a result, the higher tolerance constant, the more number of containers operated in the device. Notable is the user 7, which used relatively few services and installed a device that has high power efficiency. Therefore, when the tolerance constant was very large, the number of containers operated on the own device was overwhelmingly larger than that of other devices. Nonetheless, a comparison of the PRRs for the tolerance constants 1.2 and 100.0 yielded only a 13% difference. In other words, the optimization from the network point of view will result in an unreasonable result for the user 7. This is the reason that the tolerance constant of 1.2 is reasonable.

In summary, if the number of devices and services in the network increased, the PRR became higher. In particular, the service utilization rate has had a significant impact on the PRR, which showed that when the same service serves more users, the overall network power consumption can be reduced while maintaining a balance between users. Also, if the tolerance constant is too high, the result is unreasonable for a specific user, but the power consumption reduction is not large. If the value is too low, the power consumption benefit is fair but very low. Therefore, it is necessary to set an appropriate tolerance constant value.

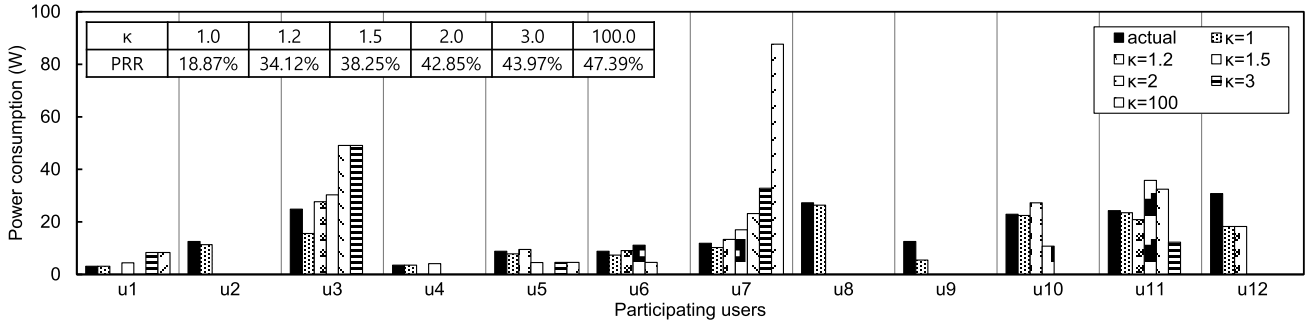

FIGURE 14. Difference of power consumption according to tolerance constant by participating users.

VI. CONCLUSION AND FUTURE WORK

Although fog computing is recognized as a computing model suited to the IoT, it is still not widely used. Uncertainty in the massive replacement of network equipment and the uncertainty surrounding infrastructure operators were identified as the major reasons. In this paper, we proposed the fog portal-based, user participatory fog computing architecture and its operation method to enhance the feasibility of fog computing. In the proposed architecture, the user registers the fog device in the fog portal after purchasing and connecting it to the network, and the fog portal performs an intermediary role between the corresponding resources and the IoT service operators. At this time, resource management was implemented based on SDN and Docker. In particular, unlike the optimization of fog computing at the network point of view, the resources were managed from the user perspective according to the characteristics of the proposed architecture. Furthermore, the fog manager performs an optimization to minimize power consumption based on service usage of participating users; the system model and problem formulation for the optimization were presented. Since the proposed objective function was MINLP, it was linearized by utilizing the auxiliary variables. The proposed architecture and optimization scheme were evaluated through simulations, and it was confirmed that the PRR was maximized when the same service was provided to more users at the same time.

In the future, we plan to optimize the container placement considering network viewpoint simultaneously. Specifically, a network model will be constructed by taking into consideration not only the internal communication within the local network, but also cooperation with adjacent local networks. In addition, we will aim to minimize the power consumption of individual devices while avoiding high latency and link congestion, which may be a problem when communicating with neighboring local networks.

REFERENCES

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 26–36, Jul. 2012.

[2] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.

[3] B. Wang, Z. Qi, R. Ma, H. Guan, and A. V. Vasilakos, "A survey on data center networking for cloud computing," *Comput. Netw.*, vol. 91, pp. 528–547, Nov. 2015.

[4] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," *J. Internet Services Appl.*, vol. 1, no. 1, pp. 7–18, May 2010.

[5] L. Wang *et al.*, "Cloud computing: A perspective study," *New Generat. Comput.*, vol. 28, no. 2, pp. 137–146, 2010.

[6] S. Khanagha, H. Volberda, and I. Oshri, "Business model renewal and ambidexterity: Structural alteration and strategy formation process during transition to a cloud business model," *R&D Manage.*, vol. 44, no. 3, pp. 322–340, Jun. 2014.

[7] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proc. ACM MobiSys*, 2014, pp. 68–81.

[8] V. Scuotto, A. Ferraris, and S. Bresciani, "Internet of Things: Applications and challenges in smart cities: A case study of IBM smart city projects," *Bus. Process Manage. J.*, vol. 22, no. 2, pp. 357–367, 2016.

[9] Y. Shi, G. Ding, H. Wang, H. E. Roman, and S. Lu, "The fog computing service for healthcare," in *Proc. 2nd Int. Symp. Future Inf. Commun. Technol. Ubiquitous Healthcare (Ubi-Healthcare)*, May 2015, pp. 1–5.

[10] C. L. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Inf. Sci.*, vol. 275, pp. 314–347, Aug. 2014.

[11] A. Fahad *et al.*, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 267–279, Sep. 2014.

[12] 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 Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[13] 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.

[14] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. ACM MCC*, 2012, pp. 13–15.

[15] *What is Docker*. Accessed: May 10, 2017. [Online]. Available: https://www.docker.com/what-docker

[16] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun. (Mar. 2016). "Fog computing: Focusing on mobile users at the edge." [Online]. Available: https://arxiv.org/abs/1502.01815

[17] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol. (HotWeb)*, Nov. 2015, pp. 73–78.

[18] T. N. Gia, M. Jiang, A.-M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare Internet of Things: A case study on ECG feature extraction," in *Proc. IEEE CIT*, Oct. 2015, pp. 356–363.

[19] S. Yan, M. Peng, and W. Wang, "User access mode selection in fog computing based radio access networks," in *Proc. IEEE ICC*, May 2016, pp. 1–6.

[20] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Netw.*, vol. 30, no. 4, pp. 46–53, Jul. 2016.

[21] H. Dubey, J. Yang, N. Constant, A. M. Amiri, Q. Yang, and K. Makodiya, "Fog data: Enhancing telehealth big data through fog computing," in *Proc. ASE BD&SI*, Oct. 2015, pp. 1–6.
[22] T. Nishio, R. Shinkuma, T. Takahashi, and N. B. Mandayam, "Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud," in *Proc. ACM MobileCloud*, Jul. 2013, pp. 19–26.
[23] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the Internet of Things," in *Proc. ACM MCC*, Aug. 2013, pp. 15–20.
[24] H. R. Arkian, A. Diyanat, and A. Pourkhalili, "MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications," *J. Netw. Comput. Appl.*, vol. 82, pp. 152–165, Mar. 2017.
[25] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of Internet of Things," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Jan./Mar. 2018.
[26] C. T. Do, N. H. Tran, C. Pham, M. G. R. Alam, J. H. Son, and C. S. Hong, "A proximal algorithm for joint resource allocation and minimizing carbon footprint in geo-distributed fog computing," in *Proc. IEEE ICOIN*, Jan. 2015, pp. 324–329.
[27] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proc. ASE BD&SI*, Oct. 2015, pp. 1–6.
[28] N. K. Giang, M. Blackstock, R. Lea, and V. C. M. Leung, "Developing IoT applications in the fog: A distributed dataflow approach," in *Proc. IEEE IOT*, Oct. 2015, pp. 155–162.
[29] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT," in *Proc. IEEE AINA*, Mar. 2015, pp. 687–694.
[30] M. Aazam and E.-N. Huh, "Fog computing and smart gateway based communication for cloud of things," in *Proc. IEEE FiCloud*, Aug. 2014, pp. 464–470.
[31] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. ACM Mobidata*, Jun. 2015, pp. 37–42.
[32] Y. Xu, V. Mahendran, and S. Radhakrishnan, "SDN docker: Enabling application auto-docking/undocking in edge switch," in *Proc. IEEE SWFAN*, Apr. 2016, pp. 1–6.
[33] A. Lewis, S. Ghosh, and N.-F. Tzeng, "Run-time energy consumption estimation based on workload in server systems," in *Proc. USENIX HotPower*, Dec. 2008, pp. 1–5.
[34] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: Freeman, 1979, pp. 245–248.
[35] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in *Proc. Networking*, May 2009, pp. 795–808.
[36] D. Meisner, B. T. Gold, and T. F. Wenisch, "PowerNap: Eliminating server idle power," in *Proc. ACM ASPLOS*, Mar. 2009, pp. 205–216.
[37] R. Morabito, "Power consumption of virtualization technologies: An empirical investigation," in *Proc. IEEE/ACM UCC*, Dec. 2015, pp. 522–527.

WON-SUK KIM was born in Seoul, South Korea, in 1985. He received the B.E., M.S., and Ph.D. degrees in electrical and computer engineering from Pusan National University, Busan, South Korea, in 2010, 2012, and 2017, respectively.

Since 2017, he has been a Post-Doctoral Research Fellow with Pusan National University, where he is involved in the BK21 Plus Creative Human Resource Development Program. He is currently a Research Fellow with the Southeastern Grand ICT Research Center. He has authored 18 articles and eight patents. His research interests include wireless mesh networks, enterprise wireless local area networks, software-defined networking, fog computing, and augmented reality.

Dr. Kim was a recipient of the Minister's Award, the highest award in creative talent evaluation, in 2017.

SANG-HWA CHUNG was born in Busan, South Korea, in 1960. He received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 1985, the M.S. degree in computer engineering from Iowa State University, Ames, IA, USA, in 1988, and the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles, CA, USA, in 1993.

From 1993 to 1994, he was an Assistant Professor with the Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, USA. He is currently a Professor with the Computer Engineering Department, Pusan National University, Busan. Since 2016, he has been the Director of Dong-Nam Grand ICT Research Center. He has authored over 240 articles and 70 patents. His research interests are in the areas of embedded systems, wireless networks, software-defined networking, and smart factory.

Dr. Chung received the Best Paper Award from ETRI Journal in 2010 and the Engineering Paper Award from Pusan National University in 2011. In 2017, he was selected as an Excellent Research Professor of the computer engineering at Pusan National University.