EdgeIoT: Mobile Edge Computing for the Internet of Things

Xiang Sun and Nirwan Ansari

In order to overcome the scalability problem of the traditional Internet of Things architecture (i.e., data streams generated from the distributed IoT devices are transmitted to the remote cloud via the Internet for further analysis), the authors propose edgeIoT, a novel approach to mobile edge computing for the IoT architecture, intended to handle the data streams at the mobile edge.

ABSTRACT

In order to overcome the scalability problem of the traditional Internet of Things architecture (i.e., data streams generated from distributed IoT devices are transmitted to the remote cloud via the Internet for further analysis), this article proposes a novel approach to mobile edge computing for the IoT architecture, edgeIoT, to handle the data streams at the mobile edge. Specifically, each BS is connected to a fog node, which provides computing resources locally. On the top of the fog nodes, the SDN-based cellular core is designed to facilitate packet forwarding among fog nodes. Meanwhile, we propose a hierarchical fog computing architecture in each fog node to provide flexible IoT services while maintaining user privacy: each user's IoT devices are associated with a proxy VM (located in a fog node), which collects, classifies, and analyzes the devices' raw data streams, converts them into metadata, and transmits the metadata to the corresponding application VMs (which are owned by IoT service providers). Each application VM receives the corresponding metadata from different proxy VMs and provides its service to users. In addition, a novel proxy VM migration scheme is proposed to minimize the traffic in the SDN-based core.

INTRODUCTION

Today, a tremendous number of smart devices and objects are embedded with sensors, enabling them to sense real-time information from the environment. This phenomenon has culminated in the intriguing concept of the Internet of Things (IoT) in which all smart things, such as smart cars, wearable devices, laptops, sensors, and industrial and utility components, are connected via a network of networks and empowered with data analytics that are forever changing the way we work, live, and play. In the past few years, many startups have embraced and actualized the concept of IoT in areas including smart homes/buildings, smart cities, intelligent healthcare, smart traffic, smart environments, and so on. Although IoT can potentially benefit all of society, many technical issues remain to be addressed.

First, the data streams generated by the IoT devices are high in volume and at fast velocity (the European Commission has predicted that there will be 50 to 100 billion smart devices connected to the Internet by 2020 [1]). Also, Cisco has predicted that the devices connected to the Internet will generate 507.5 ZB/year by 2019 [2]. Meanwhile, due to the flexible and efficient resource provisioning in the cloud [3], the big IoT data generated from the distributed IoT devices are transmitted to the remote cloud, a smart "brain" for processing big data, via the Internet in the traditional IoT architecture [4, 5], as shown in Fig. 1. However, the Internet is not scalable and efficient enough to handle IoT big data. Meanwhile, transferring the big data is expensive, consuming a huge amount of bandwidth, energy, and time. Second, since the IoT big data streams are transmitted to the cloud in high volume and at fast velocity, it is necessary to design an efficient data processing architecture to explore the valuable information in real time. Third, user privacy remains a challenging unsolved issue; that is, in order to obtain services and benefits, users should share their sensed data with IoT service providers, and these sensed data may contain users' personal information. Thus, it is critical to design a data sharing framework so that users can acquire IoT services while their privacy is guaranteed. In this article, we propose an efficient and flexible IoT architecture, edgeIoT, by leveraging fog computing and software defined networking (SDN) to collect, classify, and analyze the IoT data streams at the mobile edge. The article makes the following contributions:

- We propose edgeIoT by bringing the computing resources close to IoT devices so that the traffic in the core network can be alleviated and the end-to-end (E2E) delay between computing resources and IoT devices is minimized.
- We design a hierarchical fog computing architecture to provide flexible and scalable computing resource provisioning for each user as well as each IoT service provider.
- We propose and evaluate a novel proxy virtual machine (VM) migration scheme to minimize the traffic in the core network.

The rest of the article is structured as follows. We introduce a new mobile edge computing for IoT architecture (i.e., edgeIoT), and explain its efficiency and flexibility; we unveil the challenges in designing the edgeIoT architecture and propose some possible solutions; we conclude the article.

Digital Object Identifier:
10.1109/MCOM.2016.1600492CM

The authors are with the New Jersey Institute of Technology.

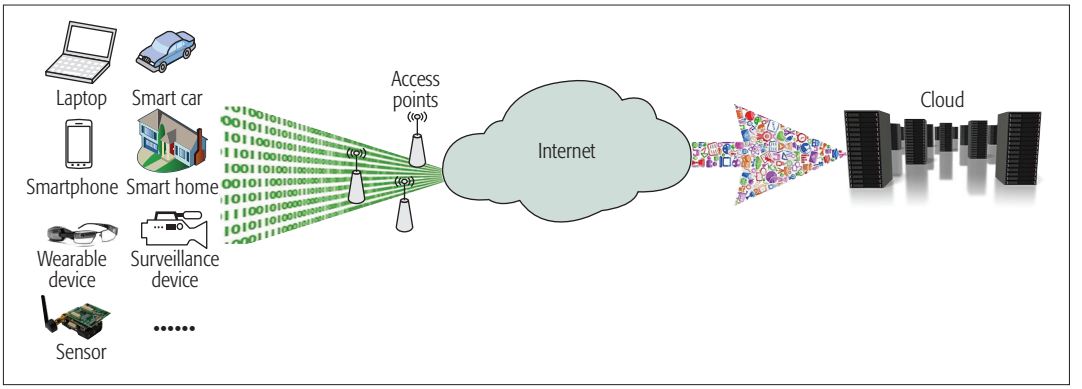

FIGURE 1. The traditional IoT architecture.

When the fog nodes do not have enough computing resources to process their local data streams, they can offload their computing workloads to the cloud at the expense of consuming more network resources and higher communications latency. Furthermore, IoT applications can be deployed in the local fog nodes or in the remote cloud to offer services to users.

Mobile Edge Computing for IoTs

Fog computing [6], which is defined as a distributed computing infrastructure containing a bunch of high-performance physical machines (PMs) that are well connected with each other, is an emerging computing paradigm bringing the computing capabilities close to distributed IoT devices. Thus, deploying a number of fog nodes in the network can locally collect, classify, and analyze the raw IoT data streams, rather than transmitting them to the cloud; this can significantly alleviate the traffic in the core network and potentially speed up the IoT big data process. However, where to deploy the fog nodes to facilitate the communications between IoT devices and fog nodes is still an open issue. The optimal fog computing deployment ensures that each IoT device has access to computing capabilities everywhere with low E2E delay and without significantly increasing the traffic of the core network. It is difficult to optimize the deployment of fog nodes due to the mobility and heterogeneity features of the IoT devices. For example, wearable devices and mobile phones move over time, and different IoT devices have different data transmission requirements, that is, some energy-insensitive devices (e.g., mobile phones and surveillance devices) need high-speed data rate, and some energy-sensitive devices (e.g., sensor nodes) require low-speed and low-energy data transmission. The heterogeneous data transmission requirements among IoT devices result in different devices adopting different wireless access technologies.

Multi-Interface Base Stations in Cellular Network

A huge number of base stations (BSs), which have already been deployed in the mobile network, provide high radio coverage. Thus, distributed BSs have the potential to connect all IoT devices whether they are moving or static. In order to support different data transmission requirements of IoT devices, each BS may be equipped with multiple wireless interfaces, as shown in Fig. 2, to facilitate emerging IoT-based wireless communications technologies such as Zigbee, device-to-device (D2D) communications with relay, Bluetooth low energy, millimeter-wave and massive multiple-input multiple-output (MIMO) communications, low-power wide area technologies, and narrowband IoT communications. Thus, a multi-interface BS can be considered as a wireless gateway to aggregate all the raw data streams from local IoT devices. Therefore, a potential deployment is to connect each BS to a fog node to process the aggregated raw data streams.

The EdgeIoT Architecture

Figure 3 shows the proposed edgeIoT. The locations of the fog nodes are flexible: a fog node can be directly connected to a BS via high-speed fibers to transmit the local data streams with the minimum E2E delay, or can be deployed at the edge of the cellular core network so that different BSs can share the same fog node to process their local data streams. Instead of applying the traditional cellular core network, which leads to inefficient, inflexible and unscalable packet forwarding and quality of service (QoS) management, the SDN-based cellular core was introduced [7, 8]. OpenFlow switches are adopted in the SDN cellular core to separate out all control functions from the data forwarding function. All the switches as well as BSs are controlled by the OpenFlow controller via the OpenFlow protocol [9]. The OpenFlow controller manages the forwarding plane of BSs and OpenFlow switches, monitors the traffic at the data plane, and establishes user sessions. Also, it provides application programming interfaces (APIs) to network management operators so that different network functionalities, such as mobility management, user authentication, authorization and accounting, network visualization, and QoS control, can be added, removed, and modified flexibly.

Note that each fog node has the ability to access the cloud through the Internet to provision computation availability and flexible application service deployment. That is, when the fog nodes do not have enough computing resources to process their local data streams, they can offload their computing workloads to the cloud at the expense of consuming more network resources and higher communications latency. Furthermore, IoT applications can be deployed in the local fog nodes or in the remote cloud to offer services to users. The flexible application service deployment is detailed later.

Hierarchical Fog Computing Architecture

Most of the data generated by users' devices contain personal information, such as photos/videos taken by mobile phones and smart cars, GPS information, health information sensed by wearable devices, and smart home status sensed

Proxy VM decomposition refers to the deconsolidation of the original proxy VM into two separate proxy VMs, each of which serves a subset of the registered IoT devices from the original proxy VM; conversely, proxy VM composition refers to the consolidation of two proxy VMs (which belong to the same user) into one proxy VM.

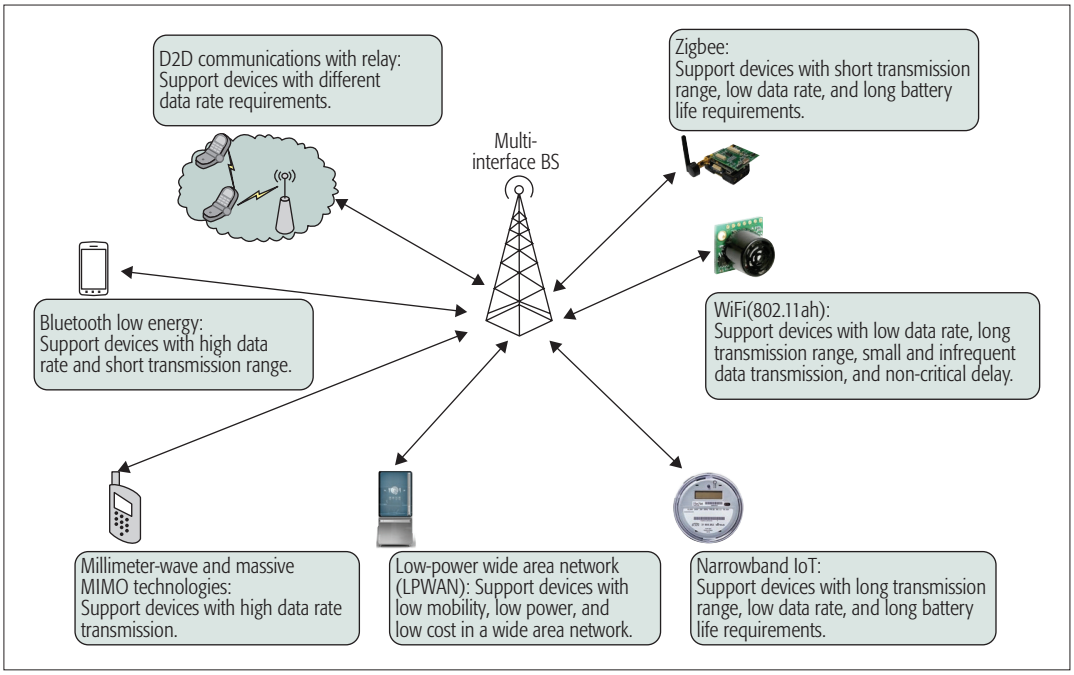

FIGURE 2. An illustration of a multi-interface BS.

by the sensors deployed in a smart home. Analyzing such humongous data can benefit not only the user her/himself but also all of society. For instance, analyzing the photos/videos taken by devices can identify and track a terrorist. Specifically, the application provider sends a photo of the terrorist to each fog node, and each fog node locally performs face matching to compare the terrorist's photo with the photos/videos taken by local devices. If matched, the fog node will upload the corresponding photos/videos to the cloud for further processing. Thus, it seems that users have to share their personal data in order to provision such services. The main challenge is to maintain user privacy in provisioning such services.

To tackle this challenge, we propose a hierarchical fog computing architecture. As shown in Fig. 4, each user[1] is associated with a proxy VM, which is considered as the user's private VM (located in a nearby fog node) that provides flexible computing and storage resources. IoT devices belonging to the user are registered to the user's proxy VM, which collects the raw data streams generated from its registered devices via a multi-interface BS, classifies them into different groups based on the type of data (i.e., structurize the raw data steams), generates the metadata by analyzing the corresponding data streams, and sends the metadata to the corresponding application VM. Note that the metadata contains valuable information generated from the raw data streams without violating user privacy. For instance, in the terrorist detection application, only the locations and timestamps of the matched photos/videos, rather than the original photos/videos, are uploaded to the application VM. The application VM, which is owned by the IoT service provider, offers the semantic model for generating the metadata by each proxy VM (e.g., the face matching algorithm in the terrorist detection application), receives the metadata from different proxy VMs, and provides services to users. For instance, all the terrorists will be identified, tracked, and arrested by analyzing the metadata from different proxy VMs, thus safeguarding our society.

The locations of proxy VMs can be dynamic: if the registered devices are statically deployed (e.g., the sensors in the smart home), the proxy VM can also remain static in the nearby fog node; if some of the registered devices are mobile (e.g., a user's mobile phone and wearable devices move from home to workplace), as shown in Fig. 5, the user's proxy VM can be decomposed into two proxy VMs: one proxy VM continues to serve the static IoT devices (in the home), and the other proxy VM migrates to the other fog nodes as the mobile IoT devices roam away. The purpose of proxy VM migration is to minimize the traffic (i.e., uploading the raw data streams from mobile devices to a proxy VM in the fog node) of the cellular core network as well as the E2E delay between a user's mobile IoT devices and its proxy VM.

Proxy VM decomposition refers to the deconsolidation of the original proxy VM into two separate proxy VMs, each of which serves a subset of the registered IoT devices from the original proxy VM (i.e., each proxy VM contains profiles and semantic models of its served IoT devices); conversely, proxy VM composition refers to the consolidation of two proxy VMs (which belong to the same user) into one proxy VM, which serves all the registered IoT devices from the original two proxy VMs. In addition, proxy VM migration involves moving the whole proxy VM (containing profiles, semantic models, and recent sensed data of the registered IoT devices) from a source PM to a destination PM. The proxy VM composition/decomposition process always invokes the proxy VM migration process.

The locations of application VMs are also dynamic and flexible: each application VM can be deployed in the local mode, remote mode, or add-on mode.

[1] A user can be a person who owns various private IoT devices, an entity/company that deploys a set of IoT devices in the area, such as the surveillance cameras, or a group of users who trust each other and share the same proxy VM.

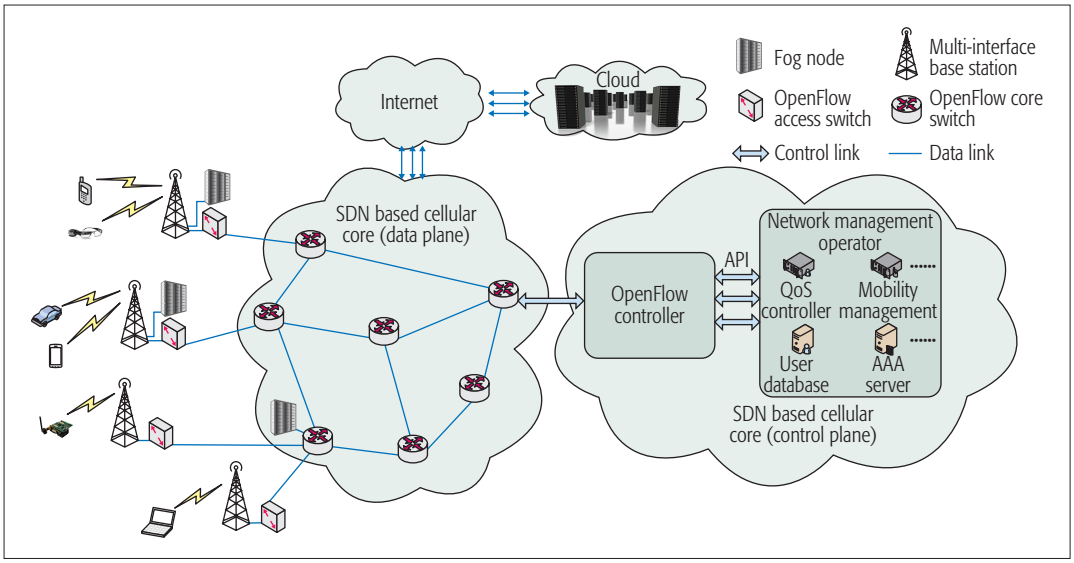

FIGURE 3. The edgeIoT architecture.

Local application VM deployment refers to the deployment of an application VM in the fog node to analyze the metadata generated by the local proxy VMs;[2] for instance, in the ParkNet application [10], which helps users find available parking spots in the urban area, each local proxy VM collects the sensed data streams from its smart cars (note that each smart car is equipped with a GPS receiver and a passenger-side-facing ultrasonic range finder to generate the location and parking spot occupancy information) and generates the metadata, which identify the available parking spots, to the application VM. The application VM will inform and assign the available parking spots to the local smart cars.

Remote application VM deployment refers to the deployment of an application VM in the remote cloud to analyze the metadata generated by the proxy VMs from different fog nodes. This deployment is necessary if an application VM needs information from a large area, such as traffic rerouting applications. Specifically, the goal of the application is to detect the traffic hotspots and select the best routing (i.e., the shortest time to reach the destination) for users. In order to detect the traffic hotspots, each smart car is equipped with sensors to measure the location and speed of the car. The sensed data streams are transmitted to the proxy VMs, which locally analyze the data streams and generate the metadata indicating the traffic congestion degree of the location. The central server in the remote cloud receives the metadata from the proxy VMs and selects the best route for each user.

Add-on application VM deployment (i.e., event-triggered application VM deployment) implies that an application VM can be locally created by some events, such as the terrorist detection application and the find-missing-children application [11]. The events, like lost children and terrorist activities detection, are reported in a specific area, and the applications need to identify and track the lost children/terrorists. Then the applications will be created in each fog node in the area and request each proxy VM in the fog node to run the face matching algorithm in order to compare recent photos/videos captured by the proxy VMs' registered devices to the photos of lost children/terrorists, and return the locations and time stamps of the photos/videos if found.

How to Implement EdgeIoT Applications

If a user is interested in one IoT application (e.g., the ParkNet application), they can download and install this app in their smart car/mobile phone. Accordingly, the user's proxy VM will install the semantic model (which calculates the available parking spots based on the sensed data) provided by the ParkNet application, and the semantic model in the user's proxy VM would have permission to access the sensed data generated by the GPS receiver and the passenger-side-facing ultrasonic range finder equipped in the user's smart car. As a reward, the user can request to find and reserve an available parking spot via the ParkNet application.

Challenges in Implementing EdgeIoT

In this section, we point out some challenges in implementing the proposed edgeIoT architecture and the corresponding solutions.

Identifications between IoT Devices and Their Proxy VMs

Initially, each user's IoT devices should be identified/registered by its proxy VM. The proxy VM should know the IDs[3] of all the user's devices and their corresponding characteristics (i.e., static or mobile devices, smart sensors sensing data or actuators responding with actions, the types of sensed data, etc.). On the other hand, the user's IoT devices should also be informed of the ID of the proxy VM so that sensor devices can transmit the private information to the correct proxy VM or actuator devices can receive commands from the correct proxy VM.

Each mobile IoT device's proxy VM may vary over time due to the decomposition/composition processes. Thus, the proxy VM needs to inform its registered mobile IoT devices when the decomposition/composition processes are triggered.

Each mobile IoT device's proxy VM may vary over time owing to the decomposition/composition processes. Thus, the proxy VM needs to inform its registered mobile IoT devices when the decomposition/composition processes are triggered.

[2] Local proxy VMs refer to the proxy VMs and application VMs located in the same fog node.

[3] Recently, many methods have been proposed to identify IoT devices, such as electronic product codes, ubiquitous codes, and the IPv6 addressing method.

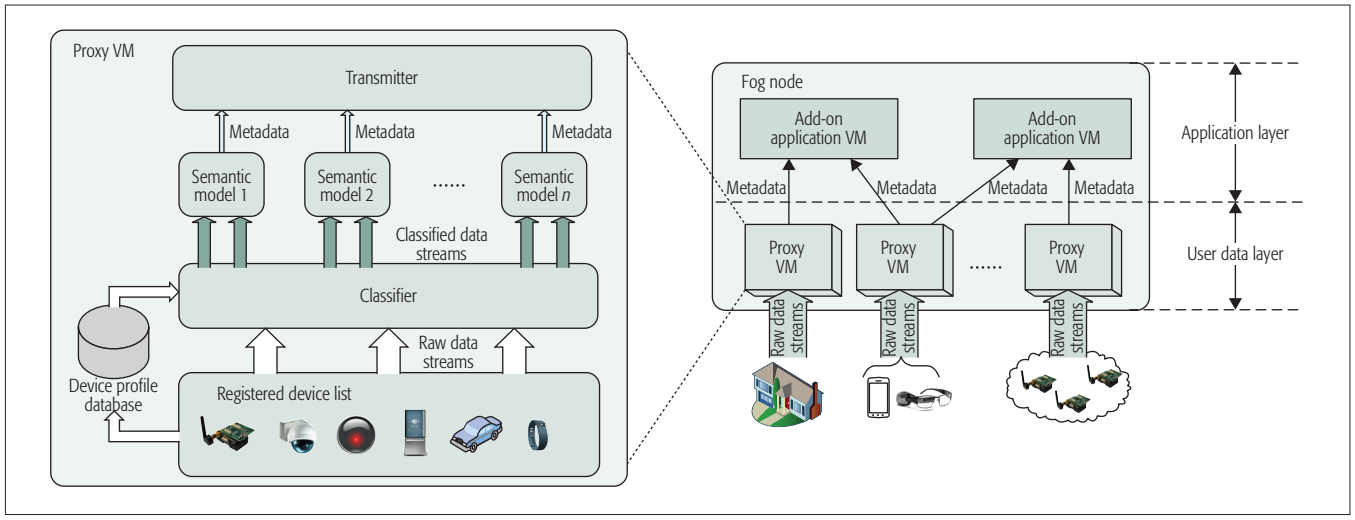

FIGURE4. The hierarchical fog computing architecture.

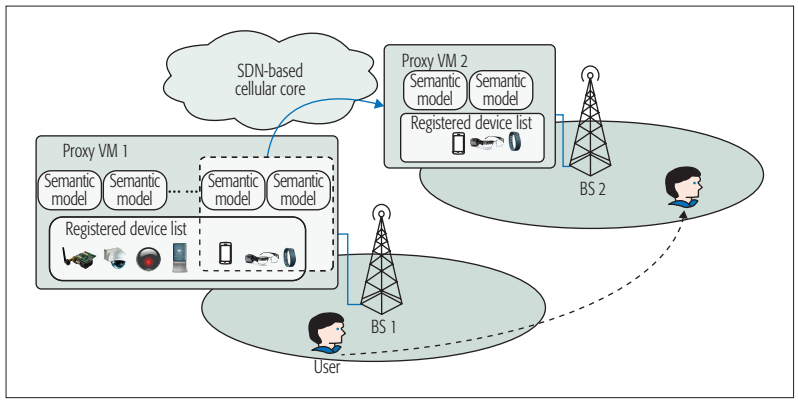

FIGURE 5. The illustration of the proxy VM decomposition and migration process.

Proxy VM Mobility Management

When a mobile IoT device roams from one BS to another BS, it should report its new location (i.e., the mobile IoT device is within the BS's coverage) to the mobility management entity (MME), which is a network management operator in the OpenFlow control layer, through the location update procedure. Mobility management is critical in edgeIoT because proxy VM decomposition/composition processes and proxy VM migration processes are determined by the locations of the proxy VMs' registered IoT devices. The proxy VM should be aware of the locations of its registered IoT devices so that it can communicate with the corresponding IoT devices via the IoT device's associated BS.

Adopting the existing mobility management in the existing LTE network is one solution; however, it requires each mobile IoT device to be equipped with a SIM card for identification and to support the location update protocol involved in the LTE network, which is not scalable and economical. Since most IoT devices are attached to their users, one alternative is to establish a local cluster network (e.g., a body area network) consisting of mobile IoT devices. The user's mobile phone or other wearable device acts as a cluster head, which can be considered as a gateway to report the locations of the IoT devices in the network, aggregate the IoT devices' sensed data streams, and upload the data streams to the corresponding proxy VM. Note that the cluster head should have the localization capability to identify its geographic location or its associated BS's ID by applying existing wireless localization technologies, such as WiFi-based localization, LTE mobility management, and Bluetooth Low Energy (BLE) beacon-based localization. The location of the cluster head represents the locations of all the members in the local cluster network.

IoT Devices Migration Management

As mentioned earlier, the IoT device's proxy VM can be decomposed and migrated among the fog nodes in order to minimize the latency for uploading the sensed data streams from the IoT devices as well as reduce the traffic load of the SDN-based cellular core. It is not necessary to migrate the IoT device's proxy VM whenever the IoT device roams into a new BS's coverage area, that is, some proxy VM migrations cannot reduce the latency but increase the traffic load of the core network. For instance, as shown in Fig. 5, a user's mobile IoT devices roam from BS 1 to BS 2, and thus their proxy VM (denoted as proxy VM 2) is decomposed from the original proxy VM (denoted as proxy VM 1) and migrates to fog node 2. If migrating proxy VM 2 from fog node 1 to fog node 2 takes T units of time (note that before the migration process is completed, the mobile IoT devices still need to upload their raw data streams to proxy VM 1 via the SDN-based cellular core) and the mobile IoT devices move out of the coverage area of BS 2 before the migration process is completed, such migration is obviously inappropriate because it increases the traffic load of the SDN-based cellular core (i.e., all the raw data streams generated from the user's mobile IoT devices should still traverse the SDN-based cellular core; in addition, extra traffic is introduced by migration) without improving the E2E delay between a user's mobile IoT devices and their proxy VM.

It is thus necessary to estimate the profit for migrating the proxy VM among the fog nodes whenever the user's mobile IoT devices roam to a new BS. The migration profit, denoted as p, is

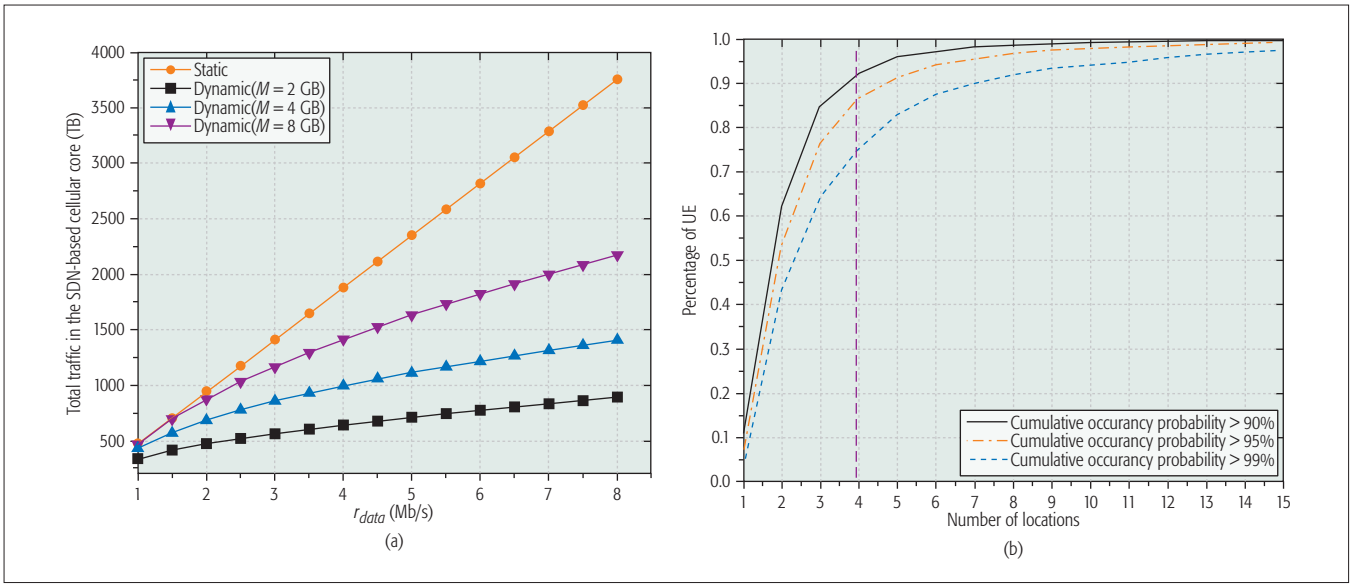

FIGURE 6. Simulation results: a) total traffic in the SDN-based cellular core vs. the average data rate of mobile IoT devices (given ς = 500 kb); b) the statistical results of the user mobility trace.

defined as the total SDN-based core network traffic reduction with and without migrating the proxy VM whenever the user's mobile IoT devices roam into a new BS: $p = L^{static} - L^{mig}$, where L^{mig} and L^{static} are the total traffic amounts generated in the SDN-based core network for migrating and nt migrating, respectively. L^{mig} comprises two parts: the migration traffic and the total data streams transmitted between the proxy VM and its registered IoT devices during the migration process:[4] $L^{mig} = T^{mig}(r^{mig} + r^{data})$, where T^{mig} is the total migration time, r^{mig} is the average bandwidth provisioning for migration, and r^{data} is the average data rate for transmitting the data streams between the user's mobile IoT devices and their proxy VM. Meanwhile, L^{static} contributes to the total data streams transmitted between the proxy VM and its registered mobile IoT devices when the mobile IoT devices remain in the new BS, that is, $L^{static} = T^{BS}r^{data}$, where T^{BS} is the retention time of the mobile IoT devices remaining in the new BS.

Apparently, an appropriate proxy VM migration implies that the estimated migration profit is larger than a predefined value ε, that is, $L^{static} - L^{mig} > \varepsilon$, where $\varepsilon \geq 0$. Thus, we can derive

$$T^{BS}r^{data} - T^{mig}\left(r^{mig} + r^{data}\right) > \varepsilon$$

$$\Rightarrow T^{mig} < \frac{T^{BS}r^{data} - \varepsilon}{r^{mig} + r^{data}} \quad (1)$$

Equation 1 indicates that the migration can benefit the network only if the migration time T^{mig} is less than

$$\frac{T^{BS}r^{data} - \varepsilon}{r^{mig} + r^{data}}.$$

Due to the fact that about 10 to 30 percent of all human movements are attributed to their social relationships, and 50 to 70 percent to periodic behaviors [12], we believe that the dynamics of future human movements can be reliably predicted based on mathematical models. Mobile IoT devices are usually attached to their users, and thus the value of T^{BS} is predictable. Meanwhile, the values of r^{mig} and r^{data} can also be estimated based on their historical traces. Therefore, the value of

$$\frac{T^{BS}r^{data} - \varepsilon}{r^{mig} + r^{data}}.$$

can be reliably estimated. In order to evaluate the migration according to Eq. 1, the migration time T^{mig} should also be predicted.

Normally, the proxy VM migration process comprises many iterations. In the first iteration, all the memory of the source proxy VM is migrated to the destination. Since the source proxy VM is still serving the user's IoT devices, the content of the memory may change during the first iteration. Thus, in the second iteration, the dirty memory pages, which are generated in the first iteration, will be transmitted to the destination. The iteration is repeated until the dirty memory pages, which are generated in the previous iteration, are less than the predefined threshold, denoted as ς. Then the source proxy VM stops serving its IoT devices and transmits the rest of the dirty memory pages to the destination; finally, the destination proxy VM resumes to serve its IoT devices. Thus, the migration time should be a function of the average data rate for doing the migration r^{mig}, the average dirty memory pages generation rate r^{dir}, the initial proxy VM memory size M, and the threshold value ς, that is, $T^{mig} = f(r^{mig}, r^{dir}, M, \varsigma)$. Based on the model proposed by [13], the migration time can be reliably estimated given the average transmission data rate for doing migration.

In order to investigate how the proxy VM migration affects the total traffic in the core network, we evaluate the total traffic in the cellular core network during the day by applying the dynamic proxy VM migration compared to the static proxy VM deployment (i.e., each proxy VM does not migrate among fog nodes after its initial deployment). In order to emulate each user's behavior, we have obtained data trac-

[4] After migration is completed, the proxy VM is placed in the fog node, with a connected BS that serves the mobile IoT devices, and hence the data streams generated by the mobile IoT devices no longer traverse the SDN-based core network to reach the proxy VM.

The proposed edgeIoT architecture can substantially reduce the traffic load in the core network and the E2E delay between IoT devices and computing resources as compared to the traditional IoT architecture, and thus facilitate the IoT services provisioning.

es of more than 13,000 users and extracted their mobility in one day in Heilongjiang Province, China. The whole area contains 5962 BSs (each BS is connected with a fog node), and each user's location (i.e., a user within the BS's coverage area) is monitored for every minute during the day. Meanwhile, the SDN-based cellular core network can guarantee the average transmission rate for doing migration to be 20 Mb/s (i.e., r^{mig} = 20 Mb/s). Each user's mobile IoT devices is attached to its own user and generates the data streams over time with the same average data rate r^{data}. Figure 6a shows the total traffic in the cellular core network during the day by varying the average data rate for transmitting the data streams between the user's mobile IoT devices and their proxy VM. Clearly, applying dynamic proxy VM migration can reduce more traffic in the SDN-based cellular core compared to the static proxy VM deployment when r^{data} increases. However, as the total amount of memory of each proxy VM (i.e., M) increases, the total traffic in the core network significantly increases accordingly. This is because as the value of M increases, the migration time becomes longer (i.e., more traffic would be generated for migration), and thus it is preferable for more proxy VMs to stay in their original fog nodes in order to avoid a huge volume of migrating traffic.

One solution to alleviate the traffic load of the core network (when the value of M is large) is to pre-allocate replicas of the users' proxy VMs in the fog nodes. Specifically, the major part of the memory is the semantic models and device profiles (which are not dynamically changed after initial installation) in the proxy VM. Thus, the replicas of the mobile IoT's semantic models can be pre-allocated to the corresponding fog nodes, the connected BSs of which are commonly visited by the user (e.g., the user's home and workplace). Note that we further analyze the mentioned user's mobility trace and find out that each user mainly settles in some areas covered by a few BSs; as shown in Fig. 6b, 92.22, 86.93, and 75.65 percent of the users spend 90, 95, and 99 percent of the time during the day (viz., 21.6, 22.8, and 23.76 h) at only four locations, respectively. This observation helps us determine the proper number and locations of replicas for each user's IoT devices. Thus, if a proxy VM tries to migrate to another fog node (which contains one of the proxy VM's replicas), rather than transmit the whole memory of the proxy VM, only the differences (between the proxy VM migration and its replicas) need to be transferred, thus dramatically reducing the migration time as well as the migration traffic.

Energy Consumption Consideration

Deploying fog nodes at the network edge may increase the operational cost for processing the IoT data streams compared to processing them in the centralized cloud (which provisions efficient and flexible resource and power management to minimize the energy consumption of the cloud). However, introducing green energy in the proposed edgeIoT architecture can substantially reduce the operational cost (i.e., reduce on-grid energy consumption) for edgeIoT providers [14]. Specifically, each fog node can be powered by both green energy and on-grid energy. The fog node would first consume green energy and then on-grid energy if green energy is not enough to satisfy the energy demands of the hosting proxy VMs in the fog node. Some fog nodes, which have less energy demand and more green energy generated, would have excessive green energy, while some, which have more energy demands and less green energy generated, would consume on-grid energy. Thus, proxy VMs can be migrated from the fog nodes (which consume on-grid energy) to the fog nodes (which have excessive green energy) in order to further reduce on-grid energy consumption.

Conclusion

This article proposes a new architecture, edgeIoT, in order to efficiently handle the raw data streams generated from the massive distributed IoT devices at the mobile edge. The proposed edgeIoT architecture can substantially reduce the traffic load in the core network and the E2E delay between IoT devices and computing resources compared to the traditional IoT architecture, and thus facilitate IoT services provisioning. Moreover, this article has raised three challenges in implementing the proposed edgeIoT architecture and has provided potential solutions.

References

[1] H. Sundmaeker *et al.*, "Vision and Challenges for Realising the Internet of Things," EC Info. Soc. and Media, tech. rep., Mar. 2010.

[2] Networking, Cisco Visual, "Cisco Global Cloud Index: Forecast and Methodology, 2014-2019," white paper, 2015.

[3] X. Sun and N. Ansari, "Optimizing Resource Utilization of a Data Center," *IEEE Commun. Surveys & Tutorials*, DOI: 10.1109/COMST.2016.25582032016, Early Access, 2016.

[4] H. L. Truong and S. Dustdar, "Principles for Engineering IoT Cloud Systems," *IEEE Cloud Computing*, vol. 2, no. 2, Mar.–Apr. 2015, pp. 68–76.

[5] L. Wang and R. Ranjan, "Processing Distributed Internet of Things Data in Clouds," *IEEE Cloud Computing*, vol. 2, no. 1, Jan.–Feb. 2015, pp. 76–80.

[6] F. Bonomi *et al.*, "Fog Computing and Its Role in the Internet of Things," *Proc. 1st Ed. MCC Wksp. Mobile Cloud Computing*, Aug. 13–17, 2012, Helsinki, Finland, pp. 13–16.

[7] X. Jin *et al.*, "Softcell: Scalable and Flexible Cellular Core Network Architecture," *Proc. 9th ACM Conf. Emerging Networking Experiments and Technologies*, Santa Barbara, CA, Dec. 09–12, 2013, pp. 163–74.

[8] X. Sun, N. Ansari, and Q. Fan, "Green Energy Aware Avatar Migration Strategy in Green Cloudlet Networks," *2015 IEEE 7th Int'l. Conf. Cloud Computing Technology and Science*, Vancouver, BC, Canada, Nov. 30–Dec. 3, 2015, pp. 139–46.

[9] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation using OpenFlow: A Survey," *IEEE Commun. Surveys & Tutorials*, vol. 16, no. 1, 2014, pp. 483–512.

[10] S. Mathur *et al.*, "Parknet: Drive-By Sensing of Road-Side Parking Statistics," *Proc. 8th Int'l. Conf. Mobile Systems, Applications, and Services*, San Francisco, CA, June 15–18, 2010, pp. 123–36.

[11] M. A. Khan *et al.*, "Moitree: A Middleware for Cloud-Assisted Mobile Distributed Apps," *4th IEEE Int'l. Conf. Mobile Cloud Computing, Services, and Engineering*, Oxford, U.K., Mar. 29–Apr.1, 2016, pp. 21–30.

[12] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and Mobility: User Movement in Location-Based Social Networks," *Proc. 17th ACM SIGKDD Int'l. Conf. Knowledge Discovery and Data Mining*, San Diego, CA, Aug. 21–24, 2011, pp. 1082–90.

[13] X. Sun and N. Ansari, "PRIMAL: PRofIt Maximization Avatar pLacement for Mobile Edge Computing," *Proc. IEEE ICC*, Kuala Lumpur, Malaysia, May 23–27, 2016.

[14] X. Sun and N. Ansari, "Green Cloudlet Network: A Distributed Green Mobile Cloud Network," *IEEE Network*, to appear; also available in Computing Research Repository (CoRR), arXiv:1605.07512, 2016.

Biographies

Xiang Sun [S'13] received his B.E. degree in electronic and information engineering and M.E. degree in technology of computer applications from Hebei University of Engineering, China. He is currently working toward his Ph.D. degree in electrical engineering at the New Jersey Institute of Technology (NJIT), Newark. His research interests include mobile edge computing, big data networking, green edge computing and communications, cloud computing, and the Internet of Things.

Nirwan Ansari [S'78, M'83, SM'94, F'09] (nirwan.ansari@njit.edu) received his B.S.E.E. (summa cum laude with a perfect GPA) from NJIT, his M.S.E.E. from the University of Michigan, Ann Arbor, and his Ph.D. from Purdue University, West Lafayette, Indiana. He is Distinguished Professor of Electrical and Computer Engineering at NJIT, which he joined in 1988. He has also assumed various administrative positions at NJIT and has been Visiting (Chair) Professor at several universities. He has also (co-)authored more than 500 technical papers, over 200 in widely cited refereed journals/magazines. He has guest edited a number of Special Issues covering various emerging topics in communications and networking. His current research focuses on green communications and networking, cloud computing, and various aspects of broadband networks. He has served on the Editorial Boards and Advisory Boards of over 10 journals, including as a Senior Technical Editor of *IEEE Communications Magazine* (2006–2009). He was elected to serve on the IEEE Communications Society (ComSoc) Board of Governors as a Member-at-Large (2013–2015). He has chaired ComSoc Technical Committees, and has actively organized numerous IEEE international conferences/symposia/workshops, assuming various leadership roles. He frequently delivers keynote addresses, distinguished lectures, tutorials, and invited talks. Some of his recognitions include some Best Paper awards, several Excellence in Teaching Awards, the Thomas Alva Edison Patent Award (2010), New Jersey's Inventors Hall of Fame Inventor of the Year Award (2012), NCE Excellence in Research Award (2014), Purdue University Outstanding Electrical and Computer Engineer Award (2015), and designation as an IEEE Communications Society Distinguished Lecturer (2006–2009). He has also been granted over 30 U.S. patents.