ACCEPTED FROM OPEN CALL

Experimental Testbed for Edge Computing in Fiber-Wireless Broadband Access Networks

Bhaskar Prasad Rimal, Martin Maier, and Mahadev Satyanarayanan

The authors design capacity-centric FiWi broadband access networks enhanced with edge computing as well as resulting fiber backhaul sharing and computation offloading capabilities. More specifically, they introduce the concept of FiWi enhanced two-level edge computing at the access edge cloud and metro edge cloud. To guarantee low end-to-end latency, they propose a TDMA based polling scheme for resource management.

Abstract

Recently, edge computing has emerged as a promising computing paradigm to meet stringent quality-of-service requirements of an increasing number of latency-sensitive applications. The core principle of edge computing is to bring the capability of cloud computing in close proximity to mobile devices, sensors, actuators, connected things and end users, thereby supporting various types of services and applications at the network edge. In this article, we design capacity-centric FiWi broadband access networks enhanced with edge computing as well as resulting fiber backhaul sharing and computation offloading capabilities. More specifically, we introduce the concept of FiWi enhanced two-level edge computing at the access edge cloud and metro edge cloud. To guarantee low end-to-end latency, we propose a TDMA based polling scheme for resource management. Furthermore, given the vital importance of experimentally demonstrating the potential and practical limitations of edge computing, we develop an experimental testbed for edge computing across converged FiWi broadband access networks. The proof-of-concept demonstration of the testbed is studied in terms of response time and response time efficiency of both edge clouds, including their respective energy consumption.

Introduction

The concept of *edge computing* has recently emerged as a new computing paradigm to meet stringent quality-of-service (QoS) requirements such as low latency, ultra-high reliability, and security/privacy. Edge computing may be viewed as a concept similar to cloudlets [1], fog computing [2], micro-data centers (MDC), and mobile edge computing (MEC) [3]. The fundamental principle of edge computing is to bring cloud computing capabilities to the edge of networks in close proximity to end devices (e.g., sensors, actuators, smartphones), thereby not only reducing end-to-end latency but also offering a variety of novel edge services and applications as well as creating new business value chains for application developers, operators, and content providers. Further, edge computing is considered one of the important computing paradigms for future 5G networks to help achieve low latency and ultra-high reliability [4, 5].

In general, optical fiber technologies may not be feasible to be deployed everywhere due to geographical constraints or when mobility is a necessity. Existing wireless access technologies (e.g., WiFi, 4G LTE/LTE-A), on the other hand, may provide user mobility but they require a reliable high-capacity backhaul to meet the high-capacity requirements of bandwidth-hungry applications. In addition, emerging holographic imaging, immersive experiences such as virtual and augmented reality, and telemedicine applications will continue to increase the demand for high-speed wireline connectivity. This trend is also in line with the so-called Edholm's Law of Bandwidth [6], which states that a unified (or converged) optical wireline and wireless network is required for providing users with both fixed and mobile services. In light of this, the convergence of fiber and wireless networks gives rise to bimodal fiber-wireless (FiWi) broadband access networks [7]. FiWi networks combine the reliability, robustness, and high capacity of optical fiber networks with the ubiquity, flexibility, and cost savings of wireless networks (e.g., 3G, WiFi, 4G LTE/LTE-A, 5G). Further, FiWi networks form a powerful platform for the support and creation of emerging as well as future unforeseen applications and provide broadband services to not only fixed subscribers but also mobile users that help stimulate innovation, generate revenue, and improve the quality of our every-day lives. In this work, FiWi access networks are further enhanced with edge computing capabilities, giving rise to *FiWi enhanced edge computing*.

A number of important issues need to be addressed in FiWi enhanced edge computing. One of the key issues when talking about FiWi enhanced edge computing or edge computing, in general, is to define the location of the edge. The choice of the edge location does not solely depend on performance, but on many additional factors. The choice criteria may vary from non-technical (e.g., CAPEX/OPEX) to technical ones (e.g., stringent QoS). Second, since it is important to ensure that different edge services (e.g., intermittent, delay-tolerant, delay-sensitive) or devices have access to network resources at the edge, designing a scheduling and bandwidth allocation scheme to meet differentiated QoS requirements is challenging [4, 8]. Note that the majority of existing studies on edge computing do not take the backhaul into account for evaluating the end-to-end performance of the network.

Digital Object Identifier:
10.1109/MCOM.2018.1700793

B. P. Rimal is with the University of New Mexico; M. Maier is with the Institut National de la Recherche Scientifique; M. Satyanarayanan is with Carnegie Mellon University.

Third, one of the critical and most important aspects of edge cloud development is the use of applications developed via open interfaces and standardized protocols. Fourth, from an end-to-end network perspective, an experimental testbed is vital to gain hands-on experience with the deployment of edge-based applications and fully understand the benefits of edge computing by means of proof-of-concept demonstration. Note that there does not exist any similar experimental testbed in the literature. The number of state-of-the-art research studies comparable to ours is rather limited in this new research area and include our previous studies [4, 9]. Other recent studies on edge computing focused on the concept of "cloudlet" [1, 10], fog computing by Cisco [2], and the so-called Mobile-Edge Computing Initiative by ETSI [3]. Note, however, that none of them provide comparable solutions and ideas on realizing edge computing in FiWi networks by means of experimental testbed demonstrations.

To address the aforementioned issues, this article introduces the concept of two-level edge computing in FiWi access networks, where we realize the edge at two different locations of the FiWi access network. The first edge is located at the optical-wireless interface in remote optical network units (ONUs) integrated with cloudlets, henceforth referred to as *Access Edge Cloud*. Note that in certain cases, very low latency may not be required, though we may need a big amount of computing power. Toward this end, it is reasonable to locate the second edge at one level higher in the network hierarchy. More specifically, in our second scenario, we define the second edge at the aggregation point of the metro/core network level, henceforth referred to as *Metro Edge Cloud*. Both types of edge cloud are discussed in greater detail in the following section. Further, to guarantee low end-to-end low latency performance in such integrated networks, there is a clear need for coordination between the access edge cloud, metro edge cloud, and intermediate passive optical network (PON) to simplify network management and improve cost efficiency. We introduce a resource management scheme based on time division multiple access (TDMA) and polling, similar to [4, 9], where edge devices are dynamically allocated transmission subslots in a deterministic manner. Given the importance of experimental measurements to fully explore the benefits of edge computing, we develop a FiWi enhanced two-level edge computing testbed and examine the implementational aspects of it.

The contributions of this article are threefold. First, we propose the concept of two-level edge computing in FiWi access networks, where applications with very low latency and ultra-high reliability requirements can be executed on the access edge cloud. Conversely, delay-tolerant applications that need more storage may benefit from the metro edge cloud. Second, a resource management scheme based on TDMA and polling is introduced to provide guaranteed QoS. Toward this end, computation offloading operations are integrated into the underlying FiWi dynamic bandwidth allocation (DBA) process. Third, an experimental testbed for FiWi enhanced edge computing is developed. To the best of our knowledge, this is the first comprehensive experimental testbed that incorporates edge computing into the optical fiber backhaul networks. In order to investigate the performance of our proposed solution, we also design an edge application and perform computation offloading onto both access and metro edge clouds. The performance is experimentally demonstrated and validated by means of analysis.

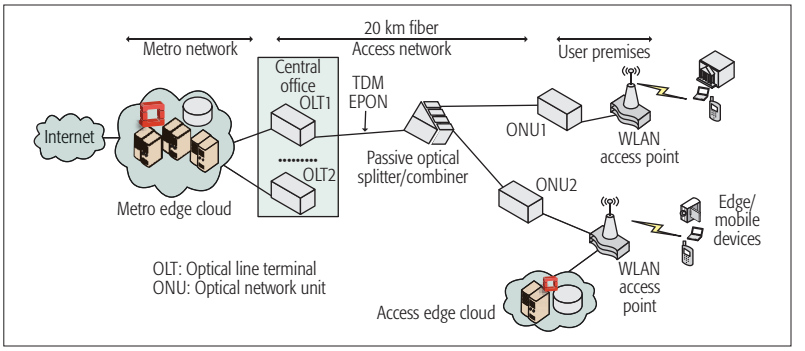

Figure 1. FiWi enhanced two-level edge computing: Communications infrastructure based on EPON, access edge cloud, metro edge cloud, and next-generation WLAN access points.

The remainder of this article is structured as follows. The following section describes the concept of FiWi enhanced two-level edge computing in technically greater detail. Then we explain our developed experimental testbed. Following that we present our proposed resource management scheme with computation offloading capabilities. Then we discuss our obtained experimental results. Finally, we draw conclusions and outline future research directions.

FiWi Enhanced Two-Level Edge Computing

Figure 1 illustrates our proposed FiWi enhanced two-level edge computing network architecture. In this network, a conventional tree-based topology is deployed, where the central optical line terminal (OLT) is located at the root of an IEEE 802.3ah Ethernet PON (EPON)/10G-EPON and the remote ONUs reside at a distance of 20 km from the OLT. Each ONU connects to an IEEE 802.11n access point (AP), referred to as "ONU-AP." An ONU-AP operates in a fully decentralized fashion, that is, it schedules transmissions and dynamically allocates upstream bandwidth to its associated wireless end devices. The central OLT is responsible for scheduling upstream and downstream transmissions and dynamically allocating upstream bandwidth to all ONU-APs.

There is no particular rule where to deploy the edge cloud in the network. Depending on the given use case, the edge may be placed at multiple levels of the network. For instance, in use cases with ultra-high reliability and very low latency requirements, it makes sense for the edge to be realized at the optical-wireless interface of FiWi access networks, whereby edge computing capabilities are integrated in the ONU-AP, which we defined above as Access Edge Cloud (also Fig. 1). This scenario is applicable to many types of applications for which latency and reliability are essential, for example, augmented reality/virtual reality (AR/VR), hospital applications (robot sur-

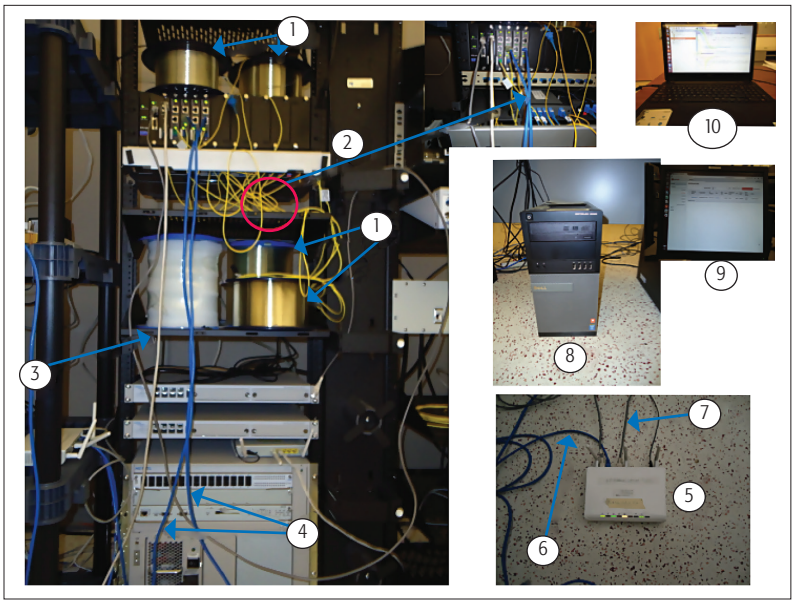

Figure 2. Experimental testbed: 1) optical fiber loops; 2) passive splitter at remote node of EPON; 3) cable connecting metro edge cloud; 4) cable connecting AP; 5) WLAN access point; 6) Ethernet connecting ONU; 7) Ethernet cable connecting cloudlet; 8) cloudlet server hosting OpenStack++ platform; 9) running VM instance in OpenStack++; 10) edge/mobile device running edge application.aa

gery, patient data processing, medical image processing), home assistance robot, real-time control and coordination of multiple devices, and body area network (BAN) applications, among others.

In the second scenario, edge computing is realized at the metro/core aggregation point, which in turn is connected to the OLT, as shown in Fig. 1. Recall from above that we refer to this type of implementation as Metro Edge Cloud. The metro edge cloud has the potential to transform the metro edge into a service and revenue generating hub, which is not fully realized yet because service providers typically view the metro edge as a predicament. In general, the metro edge cloud is more powerful than the access edge cloud to serve metro-scale user demands. It may also provide better performance-cost trade-offs than conventional remote clouds. Some examples of applications that can be run at the metro edge cloud include photo editing, optical character recognition (OCR), large-scale IoT, local live news broadcasting, and other edge applications. Note that the metro edge cloud is not envisioned to replace or compete with conventional content delivery networks (CDNs), but rather complement them, thus helping utilize network resources more efficiently and carrying traffic where CDN nodes are not deployed or are not considered cost-effective.

Note that our proposed FiWi enhanced two-level edge computing architecture offers a variety of potential advantages. For instance, from a business point of view, by sharing existing PON based backhaul infrastructures, both CAPEX and OPEX are reduced significantly. From a technical point of view, converged communications infrastructures may be exploited for multiple purposes (e.g., fixed/mobile convergence, access edge cloud, metro edge cloud, and centralized cloud). One of the most promising deployment scenarios of FiWi enhanced edge cloud computing is for hospital and industry automation, where both delay-sensitive and delay-tolerant applications are widely used. Further, emerging Tactile Internet applications [11] will benefit from FiWi enhanced edge computing given that its underlying control communication paradigm requires very low latency on the order of 1 ms and carrier-grade reliability (i.e., 99.999 percent availability).

Experimental Testbed for FiWi Enhanced Edge Computing

This section describes the experimental testbed in further detail. It consists of the following networking equipment, which provides the communications and computing infrastructure for different possible application scenarios.

TDM 1G-EPON: The FiWi enhanced edge computing testbed (Fig. 1) consists of one Sun Telecom GE8100 series OLT and four Sun Telecom GE8200 series ONUs located 20 km from the OLT. An ONU interfaces with a ZyXEL NWA570N next-generation WLAN (IEEE 802.11b/g/n) access point to realize the integrated ONU-AP.

Access Edge Cloud: The OpenStack++ platform is hosted in a Ubuntu 14.04 Desktop (64-bit) that serves as the edge cloud, as shown in Fig. 2. It is implemented in a Dell OptiPlex 9020 Mini Tower with Intel (R) Core (TM) i7-4790 Processor (Quad-core HT, 3.60GHz Turbo) and 32 GB RAM. It has three cache levels (L1, L2, and L3) with 256 KB, 1 MB, and 8 MB, respectively. OpenStack++[1] is the extension of OpenStack (https://www.openstack.org/; accessed on March 1, 2017) offering additional features, including cloudlet discovery, rapid provisioning, and VM handoff. OpenStack is a widely used open source platform for creating private and public clouds. An ONU-AP connects to the access edge cloud via wired Ethernet, as shown in Fig. 2. Table 1 summarizes the specifications of the hardware and software platform used in our experimental testbed.

Metro Edge Cloud: We set up a desktop-class machine (see Table 1 for specifications) that runs OpenStack, which serves as the metro edge cloud. The OLT connects to the metro edge cloud via wired Ethernet, as shown in Fig. 2.

Edge/Mobile Device: We use a Dell Inspiron 3521 laptop with integrated camera as the wireless edge device, whose computing power is approximately equivalent to that of state-of-the-art smartphones.

To evaluate the performance of our developed testbed, we designed a resource management scheme, developed an application, and performed computation offloading onto both edge clouds, as explained in greater detail next.

Resource Management and Computation Offloading

This section describes our proposed resource management scheme and presents a computation offloading mechanism for our developed experimental testbed.

Resource Management Overview: Our proposed TDMA-based resource management poll-

[1] OpenStack++ was developed by Carnegie Mellon University, Pittsburgh, PA, USA. For further details please visit: http://elijah.cs.cmu.edu/development.html.

	Edge/mobile device	Access edge cloud	Metro edge cloud
Model	Dell™ Inspiron 3521	Dell™ OptiPlex 9020 Mini Tower (210-AATM)	HP Phoenix h9-1130
CPU	Intel® Pentium® CPU 2127U@1.9 GHz, Dual Core	Intel® CoreTM i7-4790@3.6 GHz Turbo, Quad-core (4 VCPU for VM)	AMD FX™ -8120@3.1 GHz, eight core CPU
RAM	4 GB DDR3L SDRAM	32 GB 1600 MHz DDR3 (10 GB VM RAM)	8GB DDR3-1600 MHz
Disk	500 GB HDD	1 TB HDD (50 GB VM disk)	2 TB HDD
Network	802.11b/g/n WiFi	Broadcom NetXtreme 10/100/1000 PCIe Gigabit Ethernet	802.11b/g/n, RTL8171EH-CG Gigabit Ethernet
OS	Ubuntu 14.04 LTS Desktop 64 bit	Host: Ubuntu 14.04 LTS Desktop 64 bit Guest: Ubuntu 12.04 LTS Desktop	Host: Ubuntu 14.04 LTS 64 bit Guest: Ubuntu 12.04 LTS Desktop
Virtual machine manager (VMM)	–	QEMU/KVM-2.0.0	QEMU/KVM-2.0.0
Cloud platform	–	OpenStack++	OpenStack
Application platform	OpenCV 3.2.2	OpenCV 3.2.2	OpenCV 3.2.2
OLT	1 × Sun Telecom's SUN-GE8100 series: – one 10/100/1000Base-T RJ45 and one 1000Base-X (PON-OLT) interfaces – one network management card (one console port (RS232) and one management port (10/100Base-TX-RJ45))		
ONU	4 × Sun Telecom's SUN-GE8200 series: – one 10/100/1000Base-T, one 10/100Base-Tx, and one 1000Base-X (PON-ONU) interfaces		
AP	ZyXEL NWA570N 802.11b/g/n Ethernet wireless access point with 4 10/100 Base-TX (RJ-45) port		
Splitter	1 × 4 planar lightwave circuit (PLC) splitter (P/N: PLCSB-0104-X-SC)		
Fiber	20 km fiber length from the OLT to an ONU-AP		

Table 1. Specification of hardware and software platform used in experimental testbed.

ing protocol is designed similar to [9]. Note, however, that different from [9] this work focuses only on edge cloud traffic instead of coexistent broadband access and cloud traffic. The OLT and ONU-APs communicate with each other by exchanging control messages in compliance with the IEEE 802.3ah multi-point control protocol (MPCP). More specifically, the OLT dynamically allocates non-overlapping upstream timeslots to each ONU-AP in a centralized fashion. An ONU-AP polled by the OLT can transmit its data packets to the OLT during its assigned timeslot. In the upstream direction, given that the medium is shared among all ONU-APs, TDMA is used to avoid data collisions among transmitting ONU-APs. Traffic in the downstream direction is broadcast by the OLT to all ONU-APs. Since the logical link ID (LLID) field defines the intended destination ONU-AP, the ONU-AP filters received frames based on the LLID in the frame header.

Since in the proposed scheme ONU-APs are decentralized entities, they are able to allocate bandwidth and schedule transmissions of their associated end devices. There exist basically two possibilities for making a computation offloading decision about where to offload (i.e., access edge cloud or metro edge cloud) in FiWi enhanced edge computing. The decision can be made by either a given ONU-AP based on the QoS requirements of its associated end devices (i.e., network-initiated offloading) or independently the edge device itself (i.e., user-driven offloading, see Fig. 3). Unlike [4, 9], in this work, we apply the user-driven offloading approach to help reduce the workload of ONU-APs (e.g., computation offloading decision update process). To incorporate this information, we modify the WLAN PS-Poll and MPCP REPORT messages by using their reserved bits. The end devices send their computation offloading tasks to their associated ONU-APs during their assigned subslots. Figure 3 illustrates the protocol's signaling exchange of control messages and computation offloading procedure in more detail, whereby the control messages are used as follows.

PS-Poll: An extended WLAN control frame is used by a wireless station to inform the ONU-AP of its upstream bandwidth request. The PS-Poll frame also contains an offload flag to notify the ONU-AP about its offload request.

Beacon: An extended WLAN control frame is used by an ONU-AP to grant bandwidth to its associated end devices in the next polling cycle. The beacon frame contains the ONU-AP's timeslot and subslot start time and duration of all its associated ends devices.

REPORT: An ONU-AP generates a REPORT message to inform the OLT of its upstream aggregated bandwidth request. The REPORT also contains additional information (offloading) needed by the OLT to schedule the next timeslot of an ONU-AP. The OLT polls the ONU-AP and receives the REPORT.

GATE: The OLT generates a GATE message and sends it to an ONU-AP to inform its granted bandwidth (i.e., timeslot) in the next cycle.

Note that TDMA scheduling helps reduce the signaling overhead compared to contention-based access protocols in the front-end of the FiWi network, while the PON backhaul does not entail extra control overhead compared to a conventional EPON [4]. Further, in dynamic net-

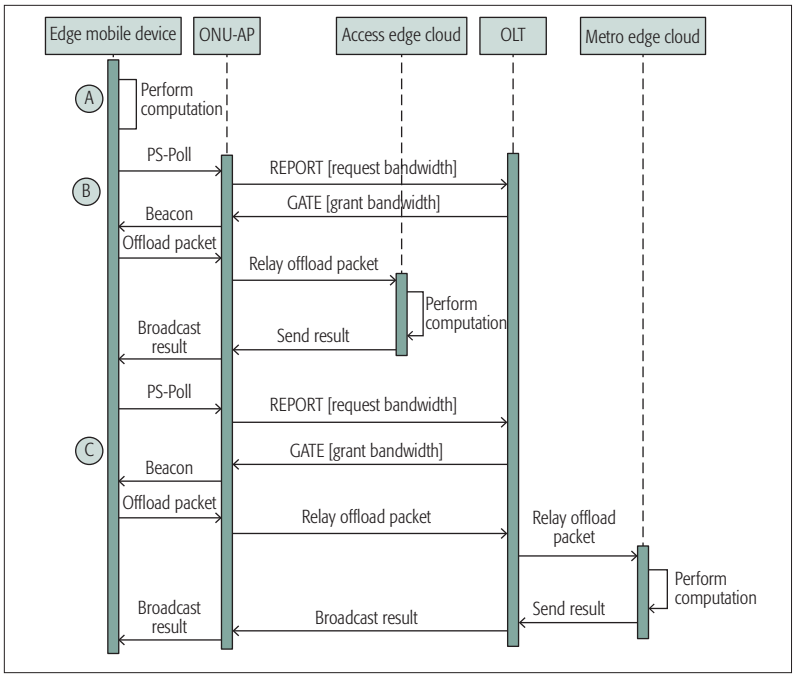

Figure 3. Space-time diagram of control messages with user-driven computation offloading: A) non-offloading scenario; B) access edge cloud offloading scenario; C) metro edge cloud offloading scenario.

work environments, it is challenging to synchronize network nodes for TDMA-based scheduling. Given that we have not considered user mobility, we utilized the well known timestamp mechanism specified in EPON/10G-EPON standards for synchronization, whereby all network devices assign their local clocks to the OLT global clock once they receive a timestamped downstream control message.

Application Layer Overview: From an edge application perspective, three use case scenarios are considered for computation offloading in the following (also Fig 3). Computation offloading or cyber foraging is a technique for improving the execution efficiency and battery life of mobile devices by moving computation from a mobile device to a powerful computer (i.e., surrogate of mobile device) [12].

Non-Offloading Scenario: In this scenario, the end device executes the computation task locally. More precisely, as shown in Fig. 4, the view controller on the edge/mobile device captures image frames from the live camera and sends the captured frames to the face detection client module for computation. The face detection client module executes the computation task using the OpenCV[2] library and then returns the result. Finally, it records the performance metrics. We note that the considered face detection represents only one interesting example application. Other compute-intensive applications may be considered in our developed testbed as well, such as real-time face recognition on cloudlets [10] and emotion recognition applications, to name a few.

Access Edge Cloud Offloading Scenario: For illustration, Fig. 4 depicts the application layer system diagram of computation offloading in greater detail for the access edge cloud scenario. In this scenario, the edge/mobile device establishes a reliable communication via transmission control protocol (TCP) with the access edge cloud, where an instance of a multithreaded face detection server module is running on top of OpenStack++. Once the connection is established successfully, the edge/mobile device offloads the computation task onto the access edge cloud. Upon receiving a request, the face detection server module executes the offloaded computation task using the OpenCV library. Afterward, the server compresses the result frames and sends them back to the face detection client module upon completion. Finally, the view controller receives the results from the face detection client module and visualizes the result.

Metro Edge Cloud Offloading Scenario: This scenario works in a similar fashion as the access edge cloud scenario. More precisely, the face detection client module establishes a connection with the metro edge cloud, which runs an instance of the multithreaded server module on top of OpenStack. After receiving the offloaded task, the server module executes the computation. The server module then compresses the result frames and sends them back to the edge device upon completion.

Experimental Results

This section presents the results and findings obtained from our experimental testbed evaluation of the aforementioned scenarios. For validation, the experimental results are compared with analytical results. Note that the focus of this study is on investigating the performance of edge computing in an integrated end-to-end FiWi network, which plays a more critical role in practical deployments than non-FiWi networks.

Computation offloading should be performed if the time to execute a task on the end device locally is longer than the response time of offloading that task onto an access/metro edge cloud. This response time difference is called *offload gain*. The response time of the access edge cloud is the sum of computation task transmission time, packet queueing delay, execution time at access edge cloud, and result transmission time. Conversely, the response time of the metro edge cloud is the sum of computation task transmission time, packet queueing delay, execution time at metro edge cloud, result transmission time, and round-trip propagation time between a given ONU-AP and the OLT. Since only one edge device is considered in the experiment, queueing delays are assumed to be negligible.

The response time efficiency is defined as the ratio of the offload gain and the response time of a task that is locally executed at an end device [9]. To evaluate the accuracy of our obtained results, we used the cumulative distribution function (CDF) of the average error metric. The average error is defined as the difference between the experimental and analytical values divided by the latter values. Further, to measure the energy consumption of the end device in both non-offloading and computation offloading scenarios, we used the open-source jRAPL [13] API. jRAPL is based on Intel's widely used Running Average Power Limit (RAPL) interfaces.

The analytical values presented in the following results are computed by using our recently developed analytical framework [9]. The FiWi

[2] OpenCV is a widely used open source computer vision library. For further information please visit: http://opencv.org/ for more information about OpenCV.

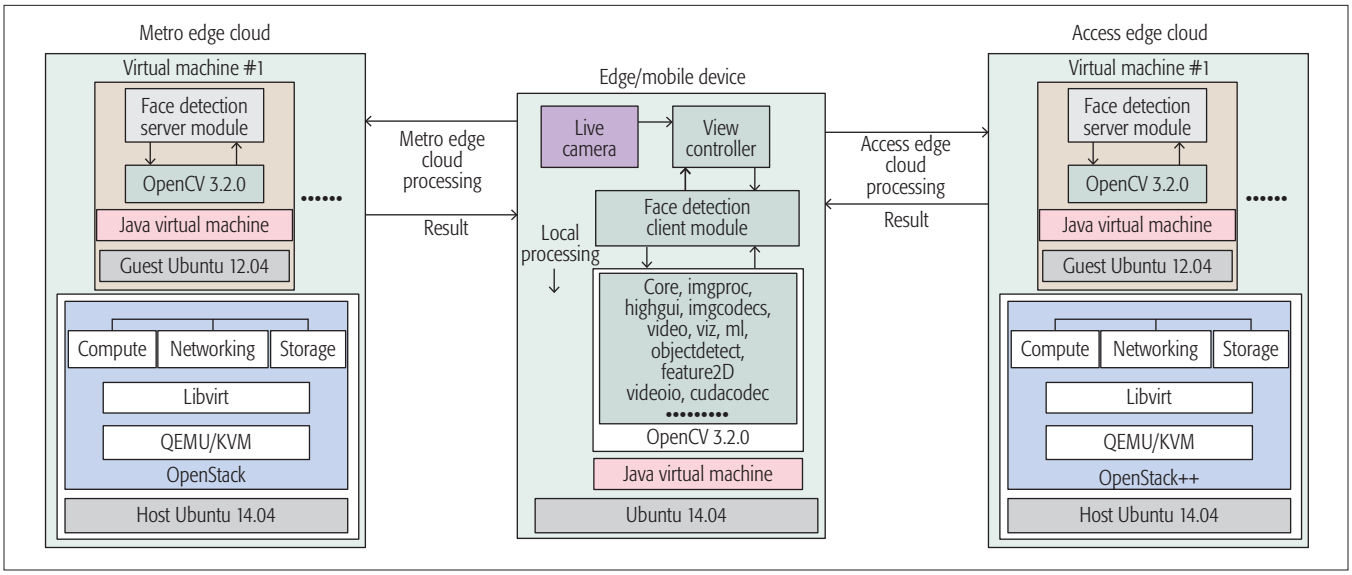

Figure 4. Application layer system diagram: Access edge cloud offloading scenario.

enhanced edge computing system of an integrated ONU-AP associated with an end device is set up for performance measurement. The transmission capacity of both uplink and downlink is set to 1 Gb/s, with a 20 km fiber reach between OLT and ONU-APs. The EPON polling cycle time and guard time between consecutive frames are set to 60 ms and 1 μs, respectively. We assume Poisson traffic that is generated by mobile users, similar to [9]. The application is partitioned into fine-grained tasks, similar to [14]. The offloaded computation task is associated with some data (e.g., images) and a set of operations (code) that perform a certain type of execution on the data (e.g., image processing, sorting algorithm). A data size of 640×427 pixels RGB color mode is assumed. Traffic denotes the length of the packet (data load) moving from the source node across the network at any given time. The traffic load (intensity) is calculated similar to [9] and is varied between 0.12 and 0.91 Erlang.

Figure 5a illustrates the response time performance for the non-offloading scenario. The figure clearly shows that the response time is a function of the traffic load. We observe that the response time remains rather small at low traffic loads, but rapidly increases for growing traffic loads. For a traffic load of 0.12 (low), the average execution time equals 865.370 ms, which may be improved by means of computation offloading onto the access/metro edge cloud.

Figure 5b shows the variation of the average response time for the edge cloud offloading scenario as a function of the offloaded traffic load. At a traffic load of 0.12, a response time value of 18.29 ms is measured experimentally. Note that this value is 47.31 times less than that in the non-offloading scenario. More importantly, we observe that decreasing the response time yields an improved battery life of the end devices. Clearly, this indicates that access edge cloud offloading is more effective for achieving a low response time. Furthermore, Fig. 5b also provides insight into the theoretical upper bound of the permissible offload traffic load for any given delay limit. For instance, for a given end-to-end delay limit of 50 ms, the permissible traffic load must not exceed 0.3, as shown in Fig. 5b. Computation offloading is not preferable when the traffic load exceeds 0.85 due to longer response time. Similarly, Fig. 5c depicts the average response time of metro edge cloud offloading for different traffic loads. At a traffic load of 0.12, a response time of 28.60 ms is obtained, which is 29.28 times less than that in the non-offloading scenario and 1.56 times more than that in the access edge cloud offloading scenario. Many applications with a delay requirement of less than 100 ms may benefit from metro edge cloud offloading, which is not achievable with conventional cloud computing.

Next, let us compare the experimental values of the maximum achievable response time efficiency of access and metro edge cloud offloading for different traffic loads in Fig. 5d. We notice that the response time efficiency of the access edge cloud is consistently slightly higher than that of the metro edge cloud. Interestingly, note that both edge clouds are able to achieve a response time efficiency of more than 96 percent at all traffic loads under consideration, which translates into a response time reduction of at least 96 percent with respect to the response time obtained in the non-offload scenario.

To illustrate the good match between our experimental and analytical results, we show the CDF of the average error in the following. Figure 5e shows the CDF of the average error between the experimental and analytical values of the response time in the access and metro edge cloud scenarios. The Pearson product-moment correlation coefficient is equal to 0.99 for the experimental and analytical values. That indicates a perfect linear association between the experimental and analytical results. Note that our obtained experimental results in Fig. 5 are slightly higher than the analytical results. This is mainly due to the presence of interference and overhead of the hypervisor. For completeness, we mention that in our presented results each data point was obtained by averaging over 100 runs. Using larger numbers of observations may further reduce this variation.

Finally, Fig. 5f shows the energy consumption versus traffic load. We observe from Fig. 5f that the obtained energy efficiency gains reflect the gains in terms of offloaded computation. More specifically, on average, an energy efficiency gain of 14.61 percent can be achieved in the considered scenarios compared to non-offloading. Note that this gain may be further increased by employing power-saving mode scheduling at the end device [4].

Although not shown in the results above, note that the FiWi polling cycle time, offloaded traffic load, bandwidth, execution time at Access Edge Cloud, and execution time at Metro Edge Cloud are other important factors that have an impact on the performance of the Access Edge Cloud and Metro Edge Cloud. Furthermore, we note that there are specific hardware requirements to properly execute OpenStack++, as specified in Table I, using a similar specification of the cloudlet developed at Carnegie Mellon University. Alternatively, Raspberry Pi could be used instead of expensive servers for edge computing. Also note that there are no vendor-specific requirements of the EPON equipment (e.g., OLT, ONU, splitter) deployed in our experimental testbed.

Conclusions and Outlook

This article introduced the concept of FiWi enhanced two-level edge computing. A resource management scheme was proposed to cope with network integration and deterministic QoS support. An experimental testbed was developed and the performance of the scheme was evaluated through experimentation and validated by means of analysis. The obtained results show that our proposed solution and developed testbed help enhance today's mobile broadband experience by reducing latency and increasing energy efficiency. The proposed architecture and testbed offer several benefits. From an economic viewpoint, the proposed architecture helps reduce the capital and operational expenditures by sharing existing fiber and wireless infrastructures. From

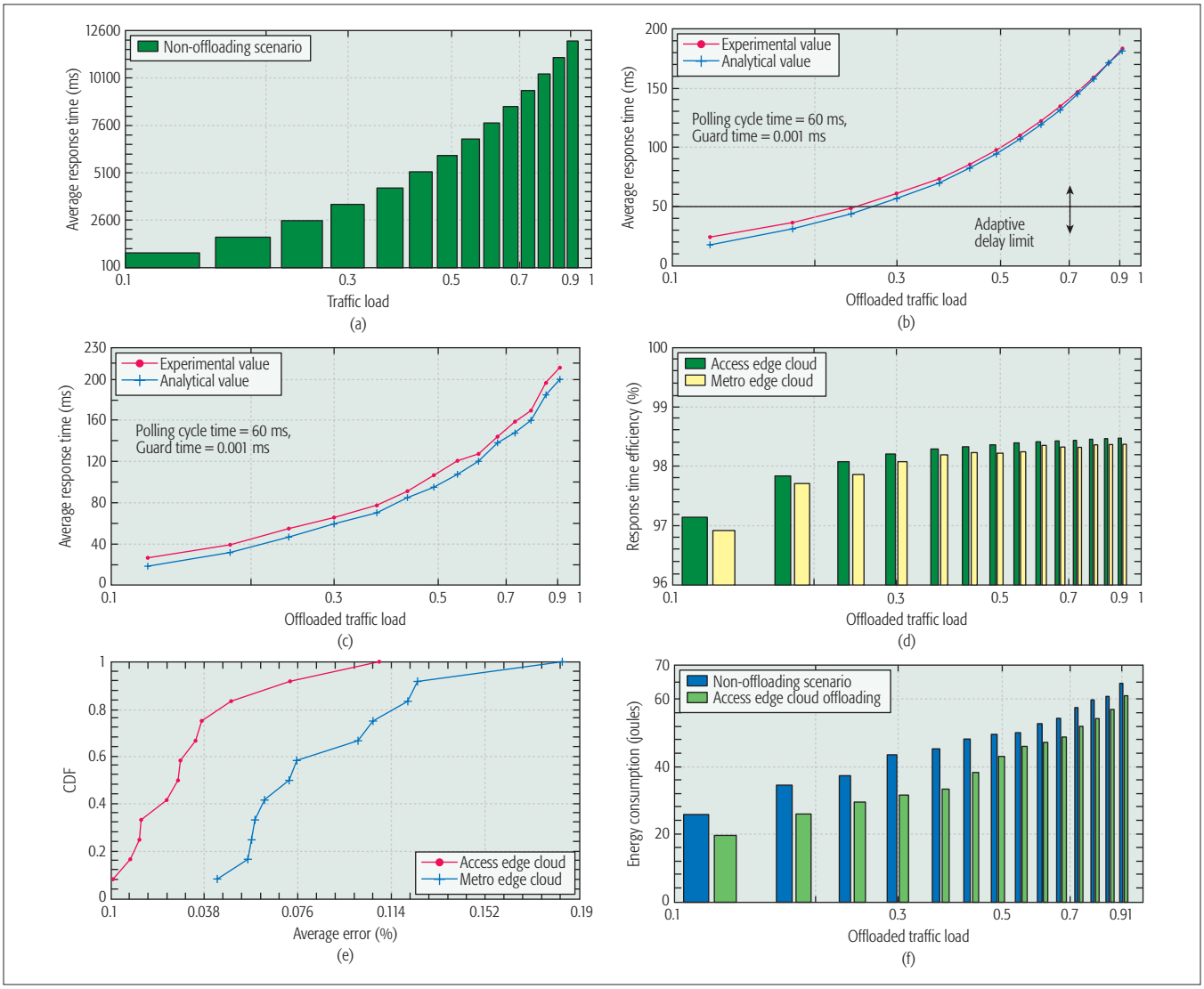

Figure 5. Experimental testbed results: a) average response time vs. traffic load in non-offloading scenario; b) average response time vs. offloaded traffic load in the access edge cloud offloading scenario; c) average response vs. offloaded traffic load in the metro edge cloud offloading scenario; d) comparison of response time efficiency of access edge cloud and metro edge cloud; e) CDF of the average error between experimental values and analytical ones; e) average energy consumption of the edge device in non-offloading vs. access edge cloud offloading scenarios.

a technical perspective, a unified resource management mechanism can be designed to operate such an integrated network efficiently.

The developed testbed enables researchers to analyze and compare various protocols and assess the performance of different emerging edge computing applications. It enables the development of new multi-disciplinary skills necessary to keep pace with the rapidly changing fields of power engineering, communications, and networking, for example, emerging Tactile Internet applications, wearable cognitive-assistance based on augmented, virtual, or mixed reality, vehicular communications, video analytics, among others. In particular, interesting research areas of FiWi enhanced edge computing include human-to-robot (H2R) communications. Since the respective merits of humans and robots are generally different, the involved tasks should be categorized accordingly. For example, monitoring, supervision, and recovery are crucial capabilities for controlling H2R communications. Machine learning helps automatize those capabilities, particularly non-verbal communications and imitation learning mechanisms for robots. Since machine learning requires significant resources (e.g., CPU, RAM), such resource intensive functionalities may be offloaded onto the access edge cloud or metro edge cloud. Another possible future research area is to exploit software-defined networking (SDN) and network function virtualization (NFV) in FiWi enhanced edge computing in order to reduce the complexity of network management. Given that in SDN networks low control plane latencies are important for achieving high network performance [15], the unification of SDN/NFV in FiWi enhanced edge computing is anticipated to play an important role.

References

[1] M. Satyanarayanan *et al.*, "The Case for VM-based Cloudlets in Mobile Computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, Oct. 2009, pp. 14–23.

[2] F. Bonomi *et al.*, "Fog Computing and Its Role in the Internet of Things," *Proc. ACM 1st Edition of the MCC Workshop on Mobile Cloud Computing*, 2012, pp. 13–16.

[3] ETSI Industry Specification Group (ISG), "Mobile-edge computing — introductory technical white paper," *ETSI*, Sept. 2014, pp. 1–36.

[4] B. P. Rimal *et al.*, "Mobile Edge Computing Empowered Fiber-Wireless Access Networks in the 5G Era," *IEEE Commun. Mag.*, vol. 55, no. 2, Feb. 2017, pp. 192–200.

[5] M. Satyanarayanan, "The Emergence of Edge Computing," *Computer*, vol. 50, no. 1, Jan. 2017, pp. 30–39.

[6] S. Cherry, "Telecom: Edholm's Law of Bandwidth," *IEEE Spectrum*, vol. 41, no. 7, July 2004, pp. 58–60.

[7] M. Maier and B. P. Rimal, "Invited Paper: The Audacity of Fiber-Wireless (FiWi) Networks: Revisited for Clouds and Cloudlets," *China Commun.*, vol. 12, no. 8, Aug. 2015, pp. 33–45.

[8] S. Davy *et al.*, "Challenges to Support Edge-as-a-Service," *IEEE Commun. Mag.*, vol. 52, no. 1, Jan. 2014, pp. 132–39.

[9] B. P. Rimal, D. Pham Van, and M. Maier, "Mobile-Edge Computing vs. Centralized Cloud Computing in Fiber-Wireless Access Networks," *Proc. IEEE INFOCOM Wkshps. on 5G & Beyond*, Apr. 2016, pp. 595–600.

[10] J. Wang *et al.*, "A Scalable and Privacy-Aware IoT Service for Live Video Analytics," *Proc., 8th ACM Multimedia Systems Conf.*, June 2017.

[11] G. Fettweis and S. Alamouti, "5G: Personal Mobile Internet Beyond What Cellular Did to Telephony," *IEEE Commun. Mag.*, vol. 52, no. 2, Feb. 2014, pp. 140–45.

[12] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges," *IEEE Personal Commun.*, vol. 8, no. 4, Aug. 2001, pp. 10–17.

[13] K. Liu, G. Pinto, and Y. D. Liu, "Data-Oriented Characterization of Application-Level Energy Optimization," *Proc. 18th Int'l. Conf. Fundamental Approaches to Software Engineering*, LNCS, vol. 9033, Apr. 2015, pp. 316–31.

[14] E. Cuervo *et al.*, "MAUI: Making Smartphones Last Longer with Code Offload," *Proc. ACM MobiSys*, 2010, pp. 49–62.

[15] A. S. Thyagaturu *et al.*, "Software Defined Optical Networks (SDONs): A Comprehensive Survey," *IEEE Commun. Surv. Tut.*, vol. 18, no. 4, July 2016, pp. 2738–86.

Biographies

Bhaskar Prasad Rimal [SM'17] (bhaskar.rimal@ieee.org) received the M.Sc. degree in information systems from Kookmin University, Seoul, South Korea, and the Ph.D. degree in telecommunications from Institut National de la Recherche Scientifique (INRS), University of Québec, Montréal, Canada. He is currently a postdoctoral fellow with the Electrical and Computer Engineering Department, University of New Mexico, Albuquerque, USA. He was a visiting scholar with the Department of Computer Science, Carnegie Mellon University, Pittsburgh, USA, in 2016. His current research interests include edge/fog computing, cyber-physical systems, Internet of Things, and 5G, and beyond. He has been serving as an associate technical editor of *IEEE Communications Magazine*, an associate editor of *IEEE Access*, an associate editor of *EURASIP Journal on Wireless Communications and Networking* (Springer), and an editor of *Internet Technology Letters* (Wiley). He was a recipient of the Doctoral Research Scholarship from the Québec Merit Scholarship Program for foreign students of Fonds de Recherche du Québec-Nature et Technologies, the Korean Government Information Technology Fellowship, the Kookmin University IT Scholarship, and the Kookmin Excellence Award as an Excellent Role Model Fellow. He is a professional member of the ACM, and member of the OSA.

Martin Maier [SM'09] (maier@emt.inrs.ca) is a full professor with the Institut National de la Recherche Scientifique (INRS), Montréal, Canada. He was educated at the Technical University of Berlin, Germany, and received M.Sc. and Ph.D. degrees (both with distinctions) in 1998 and 2003, respectively. In the summer of 2003 he was a postdoc fellow at the Massachusetts Institute of Technology (MIT), Cambridge. He was a visiting professor at Stanford University, Stanford, from October 2006 through March 2007. He was a co-recipient of the 2009 IEEE Communications Society Best Tutorial Paper Award. He was a Marie Curie IIF Fellow of the European Commission from March 2014 through February 2015. In March 2017, he received the Friedrich Wilhelm Bessel Research Award from the Alexander von Humboldt (AvH) Foundation in recognition of his accomplishments in research on FiWi enhanced networks. In May 2017, he was named as one of the three most promising scientists in the category "Contribution to a better society" of the Marie Sklodowska-Curie Actions (MSCA) 2017 Prize Award of the European Commission. He is the founder and creative director of the Optical Zeitgeist Laboratory (www.zeitgeistlab.ca).

Mahadev Satyanarayanan [F'02] (satya@cs.cmu.edu) is the Carnegie Group Professor of Computer Science at Carnegie Mellon University, Pittsburgh, PA, USA. He received the Ph.D. in computer science from Carnegie Mellon University, after bachelor's and master's degrees from the Indian Institute of Technology, Madras. His multi-decade research career has focused on the challenges of performance, scalability, availability, and trust in information systems that reach from the cloud to the mobile edge of the Internet. In the course of this work, he pioneered many advances in distributed systems, mobile computing, pervasive computing, the Internet of Things, and, most recently, edge/fog computing. He was the founding program chair of the HotMobile series of workshops, the founding editor-in-chief of *IEEE Pervasive Computing*, the founding area editor for the Synthesis Series on Mobile and Pervasive Computing, and the founding program chair of the First IEEE Symposium on Edge Computing. He was the founding director of Intel Research Pittsburgh, and was an advisor to Maginatics, which has created a cloud-based realization of the AFS vision and was acquired by EMC in 2014. He is a Fellow of the ACM.

Another possible future research area is to exploit SDN and NFV in FiWi enhanced edge computing in order to reduce the complexity of network management. Given that in SDN networks low control plane latencies are important for achieving high network performance, the unification of SDN/NFV in FiWi enhanced edge computing is anticipated to play an important role.