Communication and Computation Offloading for Multi-RAT Mobile Edge Computing

Kate Ching-Ju Lin, Hao-Chen Wang, Yuan-Cheng Lai, and Ying-Dar Lin

Abstract

The next generation of mobile networks, 5G, aims at supporting lower end-to-end latency, higher reliability and higher throughput, which can be improved by MEC and multi-RAT offloading, respectively. With MEC, a base station in 5G can be equipped with computing power, which can be called as an edge in MEC. With the assistance of the edges, traffic with computational tasks can be directly executed in local servers, without forwarding the tasks to the cloud or core network. Edge computing hence reduces the latency and the traffic load significantly. Conventional multi-RAT offloading decides based only on either communication resources or computing resources. In this work, we argue that, to better utilize the communication and computing resources, neighboring edges should share their resources and cooperatively offload the requests from their clients. To this end, we introduced a double offloading mechanism, called LCCOP, to offload incoming traffic to the best pair of radio and edge subject to the end-to-end latency of the requesting connections. We conduct simulations to compare LCCOP with the conventional offloading schemes. The results show that LCCOP's double offloading can significantly improve the satisfaction ratio by up to 83 percent and 143 percent, respectively, as compared to pure computation offloading and pure communication offloading.

Introduction

Recent research has been devoted to novel applications such as instantaneous cloud service, tactile Internet, enhanced vehicle-to-everything (eV2X), Internet of things (IoT) and communication with drones and robots. Those emerging applications demand high bandwidth and low latency. To meet such requirements, 5G has recently been proposed to deal with the exponentially increasing number of mobile and IoT devices and support extremely low-latency, reliable and scalable communications.

To improve reliability and scalability, one of the key techniques in 5G networks is heterogeneous radio access technologies (RATs), which allows an eNB to be equipped with multiple radios, for example, LTE and WiFi, so as to enable elastic access control. Seamless handover among diverse radio types helps improve load balance and better distribute traffic demands across all the available radio resources. To reduce latency, another technique, called Mobile Edge Computing (MEC), has also been standardized to move computing power from the cloud or core to edges. By doing this, a base station not only supports communications but also computation capability. Hence, MEC avoids unnecessary data forwarding, and thereby can significantly shorten the end-to-end latency. Traffic load can be shared by multiple edges inter-connected via X2 interfaces, which support both the control plane and user plane based on the X2-AP protocol [1].

By further integrating multi-RAT and MEC, a system flexibly offloads communication and computational demands from the default radio and edge to the best or all the available resources across nearby connected edges. In particular, a user equipment (UE) can exchange data through a radio of an edge but execute its computational task using the computing power of a different edge. By decoupling communication and computation resources, traffic and computing load can be elastically distributed in the entire system.

Such decoupling, however, makes efficient offloading a challenging problem as communication and computing resources now should be jointly considered. Existing literature mainly focuses on traffic offloading based on either only communication resources [2–5] or only computation resources [6, 7]. However, we notice that each connection usually has diverse demands, ultra-low latency for eV2X, high-bandwidth low-latency for video streaming and heavy computing requirement for virtual reality and augmented reality (VR/AR). We hence promote that the system should perform double offloading, that is, assigning each connection the best pair of radio and computing unit that may belong to different edges, so as to support as many requests as possible. Our design has the following advantages:

- It explicitly estimates the end-to-end delay of a connection request, thoroughly considering the communication and computation time, for identifying feasible pairs of radio and computing unit subject to its latency requirement.
- It assigns each accepted connection a suitable double offloading pair in order to best utilize all the communication and computation resources.
- It also includes an inspector that monitors in real-time the system and prevents any newly incoming request from violating the QoS demand of the existing accepted connections.

Digital Object Identifier:
10.1109/MWC.001.1800603

Kate Ching-Ju Lin, Hao-Chen Wang, and Ying-Dar Lin with National Chiao Tung University; Yuan-Cheng Lai is with National Taiwan University of Science and Technology.

	Computing offloading	Communication offloading	Decision maker	Requirement consideration	Objective
CUBI [2]	No	Yes	eNB	Latency	Min service blocking and max throughput
TO/O [3]	No	Yes	UE	Data rate	Min access cost
RBSE [4]	No	Yes	UE	Data rate	Min handover events
FLC [5]	No	Yes	UE	Latency	Min handover events and max throughput
ePSwH [6]	Yes	No	eNB	Seamless access	Min latency
MASL [7]	Yes	No	UE	Network resources	Min computational cost
ours	Yes	Yes	eNV	Data rate and latency	Max number of connections

TABLE 1. Comparison of related works.

The rest of this work is organized as follows. The next section summarizes the related work about 5G and MEC. We then present the considered double offloading problem, followed by the detailed description of our design. Emulation results for various demand distributions are then provided, followed by our conclusions and a discussion of future work.

Related Work

We classify the related works into two categories: communication offloading and computation offloading.

Communication Offloading

Several works focus on multi-RAT offloading, which considers only communication performance such as the signal strength, channel bandwidth and traffic load. Combined UE and BS Information (CUBI) [2] designs a two-round decision method, which allows each UE to prioritize its connected radios and make the base station decide RAT allocation based on the priority of all UEs. Traffic Offloading/Onloading (TO/O) [3] enables a UE to initially connect to the WiFi access point with the highest received signal strength, but handover to another one if its requirements cannot be satisfied. Reference Base Station Efficiency (RBSE) [4] introduces a quality indicator that jointly considers the required data rate of a UE and the transmit power and the load of a base station for radio selection. In Fuzzy Logic Controller (FLC) [5], a UE monitors the signal strength, mobility pattern, the load of a radio and even its backhaul network. That information is then input into a fuzzy logic controller that selects the suitable radio based on some predefined fuzzy logic rules. The above studies assume that the computing power of edges will never be a bottleneck, which may not be realistic for users with intensive computational demands. The aim of this work is to further take edge computing power into consideration.

Computation Offloading

ePSwH [6] assumes that each UE is served by a virtual machine (VM), which is initially installed in the serving eNB but can be migrated to another. The goal of ePSwH is to predict the placement of VM based on UE mobility so as to provide seamless service. However, the work does not explicitly consider the computing load of each edge, and hence may overwhelm an edge. MASL [7] allows a UE to execute its computing tasks locally but offload to the cloud as necessary. It then determines the offloading decision using game theory so as to minimize the global computation cost. However, reducing the computing cost may not always guarantee that the requirement of each UE can be satisfied. Some studies [8, 9] further control the transmission power so as to reduce the transmission delay and further partition users evenly to two cooperatively edge servers. However, those approaches do not explicitly consider the heterogeneous computational capability of servers and diverse demands of users. Hence, neighboring edges cannot cooperatively share the computational load.

In Table 1, the above previous solutions are compared in terms of considered resources, decision maker, decision constraints and objectives. Many literatures ask UEs to locally make the offloading decision, which is usually hard to ensure performance guarantee as UEs do not have global information about the entire system. More importantly, most of the works perform either merely communication offloading or merely computation offloading. We hence aim at proposing a framework that globally optimizes both communication and computation resource utilization while satisfying the requirement of as many requests as possible.

Problem Description

The main goal of our design is to offload connection requests with explicit consideration of both communication and computation resources. We consider a group of connected edges, each of which has a set of radios (e.g., LTE and WiFi interfaces) and computational units with a limited capability (instructions per second). Any two edges are interconnected by an X2 interface. Each UE initially associates with its default eNB, which is typically the one with the best channel quality, and sends a connection request to the associated eNB. Each UE requests for building a connection using a message including the information about the mean packet length, mean packet arrival rate, required computing power (instructions per packet), required data rate and the required latency. Note that different radios typically support heterogeneous service ranges (e.g., WiFi covering a much smaller area than LTE). Hence, for an edge to prioritize the available radios, each UE esti-

Our system considers the end-to-end latency, which is the sum of the communication delay and the required computational time. In our system, we ask each edge to perform prediction locally. To this end, we let edges monitor the utilization of its associated radios and computing unit and share this information periodically with neighboring edges.

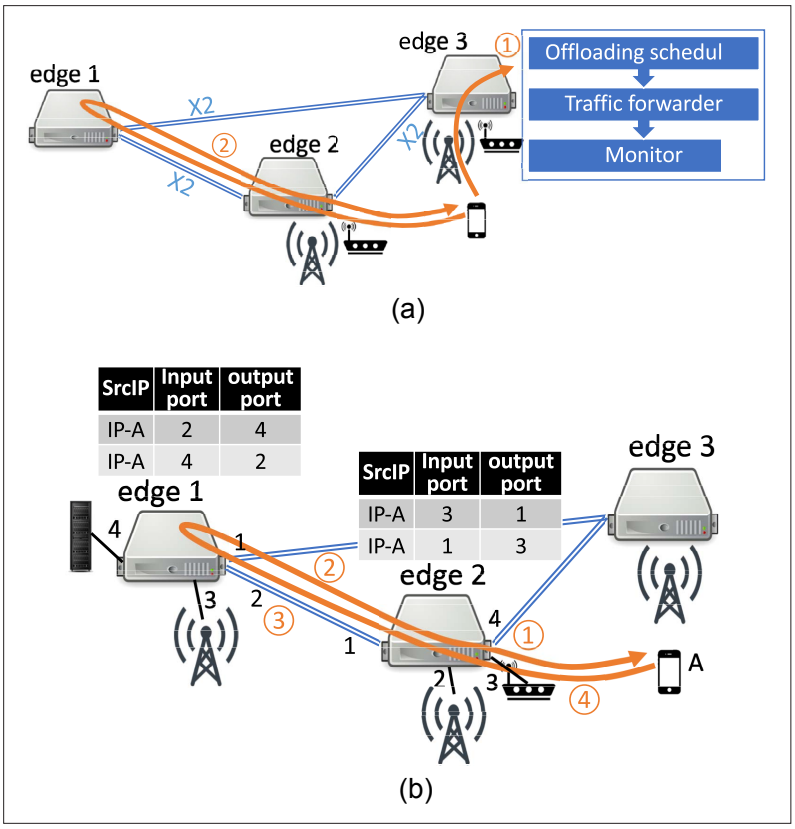

FIGURE 1. Joint communication and computational offloading: a) system architecture; b) traffic forwarding.

mates the signal-to-interference-noise ratio (SINR) of different radios by overhearing their signals and embeds those SINR measures in the request. A UE's request can be offloaded to any radio of nearby edges that can provide a sufficient channel quality.

We let each edge make real-time offloading decisions in a distributed manner, with consideration of both communication and computation resources. That is, each edge has a scheduler that handles the request from a UE. The offloading scheduler in an edge should immediately assign it a pair of radio and computing unit that can satisfy both the bandwidth and latency requirements of the request; otherwise, if no such pair can be found, the connection should be rejected. Note that the assigned radio and computing unit could belong to different edges. To this end, the edges should know how to monitor the available communication and computation resources and estimate the achievable data rate and end-to-end latency accordingly. With those estimates, the objective of offloading is to accept as many connections as possible subject to the constraints of connection requirements. After offloading, UEs will be redirected to their assigned edges. Then, each edge can further allocate communication and computing resources to its associated UEs based on some algorithm, such as round robin or MAX C/I, which is out of the scope of this work.

Joint Traffic Offloading

The framework of our design is illustrated in Fig. 1a. Each UE initially associates with its default eNB and sends a connection request to the associated eNB. Each edge (eNB) consists of three modules: scheduling module, forwarding module and monitoring module. When a new request arrives, the scheduling module predicts the achievable data rate and latency for any pair of radio and computing unit and identifies a pair that can satisfy the requirements of the request. For simplicity, for each edge, we call a radio of the edge *home radio*, while calling a radio of a neighboring edge *remote radio*. Similarly, a *home and remote computing unit*, respectively, indicates the computing unit of a home edge and a neighboring edge. If the assigned radio and computing unit are remote, the forwarding module (illustrated in Fig. 1b) initiates handover of the UE and redirects the computing tasks to the assigned computing unit. Finally, the monitoring module continuously monitors the system and makes sure that the requirements of the accepted connections can consistently be satisfied. By admission control in the scheduling module and resource update in the monitoring module, our system can flexibly manage the available resources of cooperative edges.

Offloading Scheduling

The default edge of a UE should make a real-time offloading decision when it receives a connection request. To identify a set of offloading choices that satisfy the requirements of a request, the edge should predict the achievable throughput and latency of any given pair of radio and computing unit. Note that our system considers the end-to-end latency, which is the sum of the communication delay and the required computational time. In our system, we ask each edge to perform prediction locally. To this end, we let edges monitor the utilization of its associated radios and computing unit and share this information periodically with neighboring edges.

Communication Delay: The end-to-end communication latency is the summation of propagation delay, transmission delay, queueing delay and forwarding delay. That is, the communication latency of a requesting connection c served by radio r of edge e (forwarded from its home edge e') can be by $T^{comm}_{c,e,r} = T^{prop}_{c,e,} + T^{tx}_{c,e,r} + T^{queue}_{e,r} + T^{fwd}_{e',e}$. The propagation delay $T^{prop}_{c,e,}$ is typically very small and negligible. The transmission delay $T^{tx}_{c,e,r}$ between a UE and a radio r depends closely on its link quality, that is, SINR. Hence, for a radio, different UEs would obtain heterogeneous transmission delays. In most wireless protocols, the optimal data rate can be achieved by proper modulation and coding scheme (MCS) selection, for example, [10–12] for LTE and [13–15] for WiFi. As the bit error rate of a modulation scheme could vary with SINR, those MCS adaptation protocols usually leverage SINR to select the best MCS that achieves the highest effective throughput, that is, the data rate of the MCS multiplied by the packet delivery ratio. As a UE has embedded the SINR measures of its neighboring radios in the request, the system can determine its optimal MCS of any radio and the corresponding achievable data rate. Then, the offloading scheduler directly predicts the transmission delay of different radios (even remote radios) accordingly.

The queueing delay of each radio $T^{queue}_{e,r,}$ is harder to predict since each edge can only know the load of its home radios. Also, the queueing

delays for uplink and downlink transmissions are different. As the traffic load usually fluctuates, we allow each edge to estimate the queueing delay of its radio either empirically or analytically. For empirical estimation, each edge can estimate the queueing delay of a packet as the total time required to send all the packets queued in its buffer. The uplink queueing delay is harder to estimate as the edge cannot monitor the queue of each UE. We could alternatively estimate the uplink queueing delay of a connection by measuring the duration required for receiving all the uplink packets arrived during the average inter-packet time of that connection. Alternatively, for analytical estimation, an edge can formulate a queueing model based on the traffic arrival rate of UEs and derive the mean queuing delay. In our implementation, we adopt empirical estimation; however, this design option can be decided by the network operator. We then ask each edge to exchange this information with neighboring edges for local offload scheduling. The traffic forwarding delay across different edges $T^{fwd}_{e',e}$ can be predicted in a similar way and exchanged among cooperative edges. With this information, the offloading scheduler of the default edge can derive the end-to-end communication latency of any radio nearby the requesting UE.

Computation Delay: The computation time of a packet of a connection consists of the execution time and the waiting time. The execution time can be directly calculated by the required computing power divided by the computing capability of an edge. The waiting time is, however, determined by the computational load of an edge, which can be estimated by the aggregated traffic arrival rate and their required computing power. Specifically, it can be estimated by summing up the execution time multiplied by the packet arrival rate of all the accepted connections. That is, the computational time can be estimated by $T^{comm}_{c,e,} = T^{exec}_{c,e,} + T^{wait}_{c,e,} = D_c/C_e + \Sigma_{c' \in \mathcal{A}} D_{c'}/C_e * r_{c'}$, where D_c is the required computing power of a requesting connection c, C_e is the computing capability of edge e, $\mathcal{A}$ is the set of accepted connections and $r_{c'}$ is the packet arrival rate of an accepted connection c'.

With the above estimation, we can analyze the end-to-end delay (i.e., communication delay plus computation time) of each pair of radio and computing unit and filter the candidate pairs subject to the requirement constraint. Specifically, we estimate the end-to-end latency for any pair of radio and edge and identify the pair (e, r) that minimizes the end-to-end delay for connection c, that is, $T^{comm}_{c,e,r} + T^{comm}_{çe}$. If the minimum delay still exceeds the requested delay constraint, the system will reject this request since none of the edges is capable of serving it.

Handover and Traffic Forwarding

Once the scheduler determines the offloading pair of radio and computing unit, the default edge asks the traffic forwarding module to install the connection. The forwarder first triggers handover (if necessary) and then configures the forwarding rules for the offloading decision. Before the connection is built, the default edge will trigger handover for the requesting UE from the default radio to the assigned radio, which can belong to a different edge. The connection is then built after handover. For decoupling the communication and computation resources, we should configure forwarding rules in the assigned radio and the edge switch of the assigned computing unit for data forwarding, illustrated as the yellow block of Fig. 2.

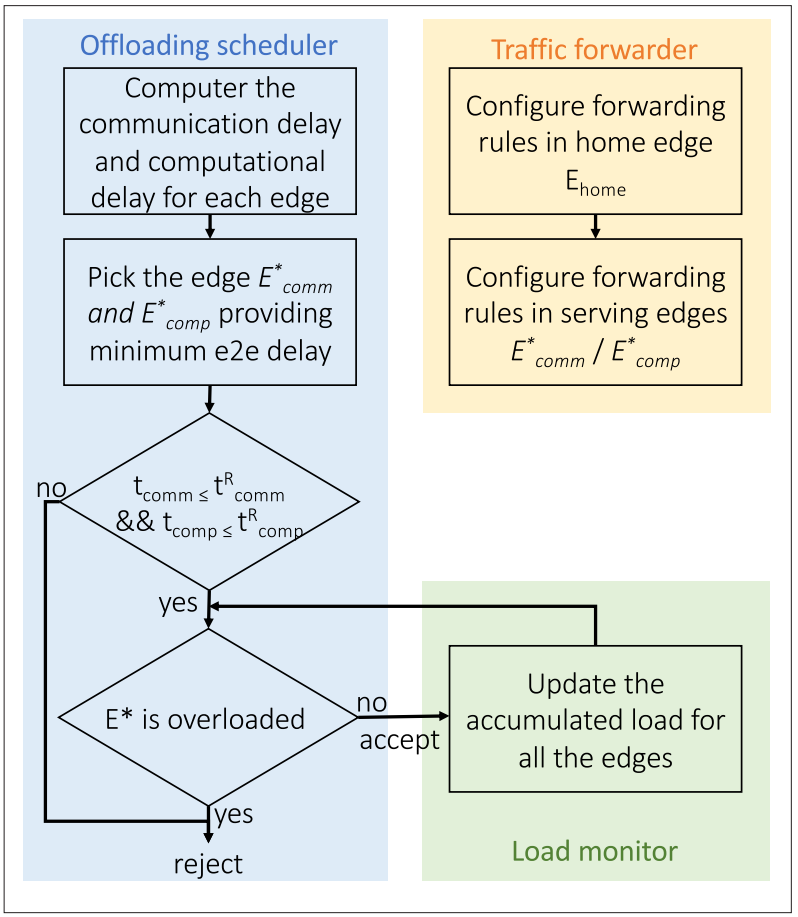

FIGURE 2. Flow chart.

Consider again the example in Fig. 1a, where the request of a UE from home edge 3 is handed over to the radio of edge 2 and relayed to edge 1 for executing the computational task. To enable such offloading, we should install the forwarding tables as shown in Fig. 1b. In particular, the offloading scheduler should insert a forwarding rule in the flowtable of the edge switch of the assigned radio (e.g., edge 2 in Fig. 1b) such that the packets can be redirected to the assigned computing unit (e.g., edge 1 in this case) through the X2 interface. Then, it also has to insert the forwarding rule in the edge switch of the remote edge to deliver the tasks to the assigned computation unit and also respond the computing results back to the associated radio of the UE, for example, the table of edge 1 in Fig. 1b. The edge of the assigned radio then relays the response to the UE through the assigned radio. By such forwarding configuration, we can separate the control plane in the default edge and the data plane of the edge switches and radios. The default edge is only in charge of identifying the suitable pair of radio and computation unit. After that, the UE can directly send its traffic through the assigned radio to the allocated computation unit without the involvement of its default edge. This makes the signaling overhead of offloading negligible. The blue

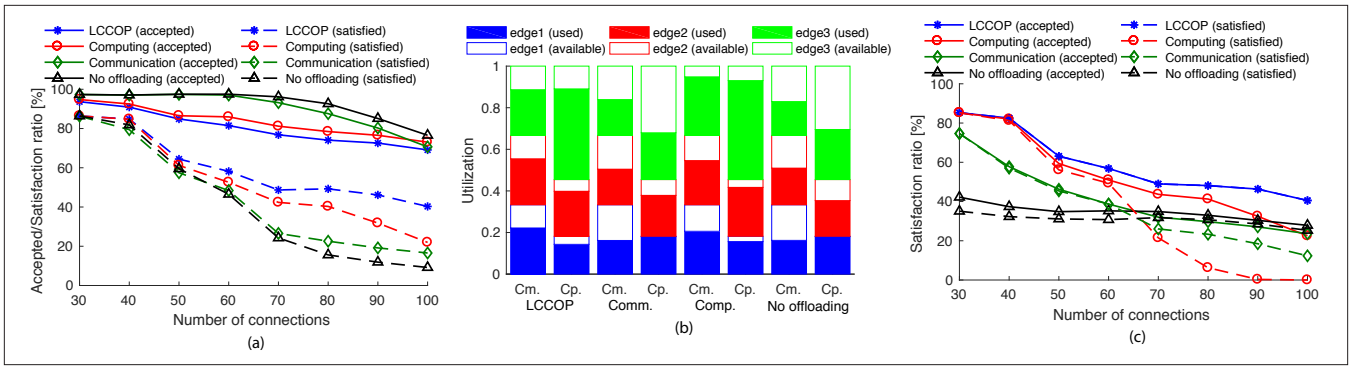

FIGURE 3. Performance comparison for various number of connection requests: a) performance comparison; b) resource utlization; c) monitoring.

block of Fig. 2 illustrates the flow chart of LCCPP's offloading scheduling module.

Connection Monitoring

After admitting the connections, the system should continuously track resource utilization such that the performance of each accepted connection can consistently be guaranteed. If an accepted connection cannot obtain the requested provision of service, the system should re-identify a feasible pair of radio and edge for it and continue service provision. However, the cost of tracking each accepted connection over time and connection migration is fairly expensive. To avoid this complexity, we alternatively adopt a one-time decision strategy. To be more specific, before accepting a connection, we further examine whether admitting this connection will violate the provision of the previously accepted connections. If so, we decline this request even if the current system can satisfy it. That is, we never admit a connection that may make other accepted connections become unsatisfied.

To achieve this goal, ideally we should re-estimate the latency of all the accepted connections with the assumption of admitting a new request. However, this solution does not scale as the number of connections grows rapidly. Hence, we adopt a more efficient examining mechanism that alternatively calculates system utilization. We observe that the latency of an accepted connection usually can stay stable if the system is not overloaded. Hence, if we can make sure that a newly accepted connection does not overload the system, the previous accepted connections should still obtain their requested provision. The system load can be estimated by the proportion of the allocated resources to the total capacity of each edge, as illustrated as the green block of Fig. 2. As this estimation only needs the information about communication and computing requirements of each connection and edge capability, it does not introduce any additional signaling overhead and has a lightweight computational cost, which is much smaller than periodically re-calculating the end-to-end latency of every accepted connection. To summarize, we admit the connection when there exists a pair of radio and edge that can satisfy its latency requirement, and the overall system load can be no larger than the edge capability after it joins. We will evaluate later the achievable satisfaction rate of connections by such lightweight distributed admission control.

Performance Evaluation

The performance of the proposed joint offloading method is evaluated via Network Simulator 3 (NS-3) [16]. We consider a scenario of three neighboring edges interconnecting with each other by a 10Gb/s X2 interface. The distance between any two edges is set to 5 kilometers, which is roughly the same with the LTE coverage range. In our implementation, each edge is equipped with an LTE and WiFi interface (both supporting the bandwidth of 20 MHz) and has its own computing power. A number of static clients are randomly distributed in the simulation environment. Each UE picks its closest base station as its home edge. Since the coverage ranges of LTE and WiFi are different, some UEs may only be covered by LTE radios, but can be offloaded among multiple nearby LTE base stations. We simulate different types of connections, including:

- Latency sensitive (which needs 3M–4M instructions and a latency requirement of 20–40 ms).
- Heavy computing (which needs 25M–35M instruction and a latency requirement of 70–90 ms).
- Normal (which needs 8M–10M instructions and a latency requirement of 100–120 ms).

The order of the latency and computing power is set similar to the configurations used in [6, 7]. Each UE is randomly assigned one of the above connection types and generates requests following a Poisson process with a mean arrival rate of 10 packets per second.

As a connection may be accepted but does not achieve its requested latency or bandwidth requirement, we hence evaluate our design in terms of both the acceptance ratio, that is, the percentage of connections accepted, and the connection satisfaction ratio, that is, the ratio of the satisfied connections to all the requests. Our simulations are designed to check the effectiveness of joint offloading and the impact of connection distribution. We compare LCCOP with communication offloading, computing offloading and no offloading, respectively. For all the comparison schemes, we perform our monitoring check, that is, preventing the system from being overloaded.

Effectiveness of Joint Offloading

To check the impact of heterogeneous computing resources and UE distribution, we configure the computing power of the three edges to 4,000, 6,000 and 20,000 MIPS, respectively, and distrib-

ute 60 percent, 20 percent and 20 percent of UEs to the area of edges 1, 2 and 3, respectively. The UEs allocated to each edge are then uniformly randomly deployed within the coverage area of that edge. In such a configuration, we test a more challenging scenario where edge 1 is a hot-spot (i.e., covering most of the UEs) but has a limited computing power. Figure 3a shows the acceptance ratio and the satisfaction ratio for various numbers of connections. The acceptance ratio of communication offloading and no offloading decreases as the number of connections grow since it does not consider heterogeneous computing resources, but the allocated edge could reject the requests in the step of monitoring check if the computing power is insufficient. Though computing offloading accepts more connections than our LCCOP, it however achieves a lower satisfaction ratio since the assigned edge might introduce a long communication latency and, as a result, violate the end-to-end latency requirement. In other words, it accepts more users but fails to finish their computing tasks, which hence significantly wastes the resources of the edges. Overall, LCCOP improves the satisfaction ratio by 83 percent and 143 percent, respectively, as compared to computing offloading and communication offloading when the system loading is high, that is, 100 connections.

To take a closer look at how LCCOP improves the satisfaction ratio, we further plot the resource utilization of the three edges for the comparison schemes in Fig. 3b. The figure shows that, for communication offloading, Edge 1 underutilizes its communication resources but saturates its computing resources. This is why the connections offloaded to Edge 1 become congested. Our LCCOP can jointly consider both resources and hence better balance the utilization of both the communication and computing resources, thereby improving the overall satisfaction ratio. We then check the effectiveness of the satisfaction ratio with and without the monitoring check. Figure 3c shows that, without checking the system loading, both communication and computing offloading could accept some connections whose requirement can be satisfied but would disturb the previous accepted connections. Hence, the satisfaction ratio could drop significantly without monitoring, for example, dropping to nearly 0 percent for computing offloading. This shows that the lightweight checking based on the system load can effectively protect the accepted connections and avoid the need of periodically monitoring the achievable latency of the existing connections.

Impact of Connection Distribution

We next examine the impact of UE distribution and the heterogeneity in computing resources. Tables 2a and 2b summarize the configurations we have tested. Figure 4a illustrates the performance of configuration 1. The results show that UE distribution is less relevant to the achievable satisfaction ratio of the comparison offloading schemes. Only the no offloading scheme achieves a worse performance for the hot-spot scenario. This is because when the number of connections is large, that is, 80 connections in our simulation, no matter which UE distribution we test, the three edges are all saturated. Hence, proper offloading can fully utilize the balanced resources. Figure 4b illustrates the performance of configuration 2. The results show that both the communication offloading and no offloading scheme do not explicitly consider computing power, and hence may accept some connections that will be rejected in the monitoring check phase. Hence, they achieve a lower acceptance ratio. Moreover, even when computing offloading carefully considers the heterogeneous computing resources, it may allocate some edges whose communication resources have been saturated, and thus perform worse than LCCOP in terms of the satisfaction ratio. Our LCCOP can efficiently utilize the aggregated resources and hence achieve stable performance no matter how resource distribution changes.

a) UE distribution				
		E1	E2	E3
Communication resource (UEs)		30	30	30
Computing resource (MIPS)		3666	3666	3666
UE distribution	Uniform	33%	33%	33%
		40%	30%	30%
		50%	30%	20%
		60%	20%	20%
	Hot-spot	70%	20%	10%
b) Computing resource distribution				
		E1	E2	E3
Communication resource (UEs)		30	30	30
UE distributions		26	27	27
Computing resource (MIPS)	Balanced	3666	3666	3666
		3300	3300	4400
		2200	3300	5500
		2200	2200	6600
	Unbalanced	1100	2200	7700

TABLE 2. Configuration setup.

Conclusions

In this work, we have presented a multi-RAT double offloading system, called LCCOP. Our design jointly considers both the communication and computing resources in a 5G-MEC scenario. By collecting the performance requirement information from UE and the utilization of edges, our design can identify suitable edges to offload the computing and communication load separately. We believe that the future MEC architecture can allow neighboring edges to aggregate their resources and share the loading adaptively. We further develop a lightweight monitoring scheme to prevent a new incoming connection from destroying the QoS of existing connections. By combining the double offloading schedule and the lightweight monitor, LCCOP achieves a connection satisfaction ratio 83 percent and 143 per-

Even when computing offloading carefully considers the heterogeneous computing resources, it may allocate some edges whose communication resources have been saturated, and thus perform worse than LCCOP in terms of the satisfaction ratio. Our LCCOP can efficiently utilize the aggregated resources and hence achieve stable performance no matter how resource distribution changes.

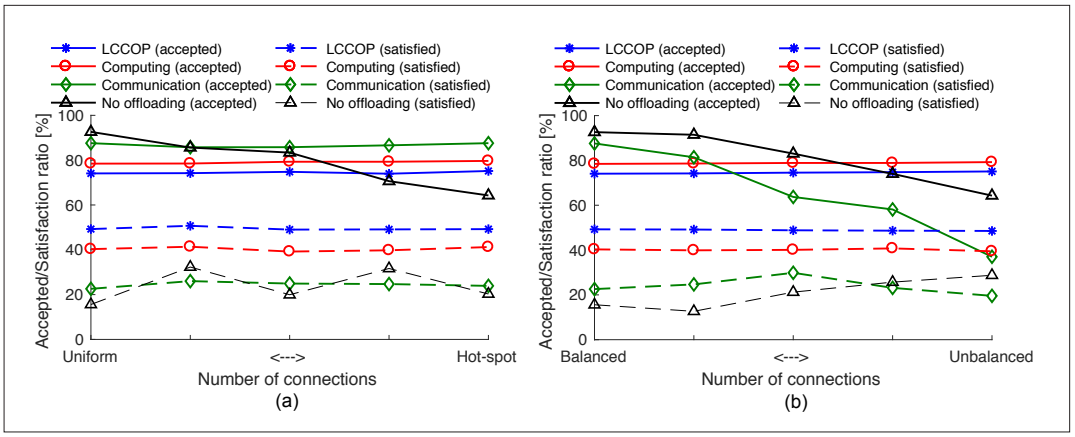

FIGURE 4. Impact of heterogeneity: a) impact of UE distribution; b) resource utilization.

cent, respectively, higher than pure computing offloading and pure communication offloading when the edge resources are unbalanced and nearly saturated.

As we now identify the pair of radio and computing units that can minimize the end-to-end latency, it is sometimes unnecessary if the required demand is not that rigid. It is worth studying how to identify the best pair among multiple feasible choices so as to maximize the admission ratio. The offloading efficiency can be improved if the system can further take UE mobility into consideration. We leave the above challenging issues to our future study.

References

[1] 3GPP, "Evolved Universal Terrestrial Radio Access Network (E-Utran); X2 Application Protocol (X2ap)," 3rd Generation Partnership Project (3GPP), TS 36.423.

[2] Y.-D. Lin *et al.*, "Wi-Fi Offloading between LTE and WLAN with Combined UE and BS Information," *Wireless Networks*, vol. 24, no. 4, 2018, pp. 1033–42.

[3] C.-F. Chiasserini *et al.*, "Traffic Offloading/Onloading in Multi-RAT Cellular Networks," *IFIP Wireless Days (WD)*, IEEE, 2013.

[4] A. Orsino *et al.*, "Effective RAT Selection Approach for 5G Dense Wireless Networks," *Proc. IEEE Vehicular Technology Conf. (VTC Spring)*, IEEE, 2015.

[5] A. Kaloxylos *et al.*, "An Efficient RAT Selection Mechanism for 5G Cellular Networks," *Proc. IEEE Wireless Commun. and Mobile Computing Conf. (IWCMC)*, IEEE, 2014.

[6] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic Resource Allocation Exploiting Mobility Prediction in Mobile Edge Computing," *Proc. IEEE Personal, Indoor, and Mobile Radio Communications (PIMRC)*, IEEE, 2016.

[7] J. Zheng *et al.*, "Stochastic Computation Offloading Game for Mobile Cloud Computing," *Proc. IEEE Int'l. Conf. Commun. China (ICCC)*, IEEE, 2016.

[8] T. G. Rodrigues *et al.*, "Hybrid Method for Minimizing Service Delay in Edge Cloud Computing through VM Migration and Transmission Power Control," *IEEE Trans. Computers*, vol. 66, no. 5, May 2017, pp. 810–19.

[9] T. G. Rodrigues *et al.*, "Cloudlets Activation Scheme for Scalable Mobile Edge Computing with Transmission Power Control and Virtual Machine Migration," *IEEE Trans. Computers*, vol. 67, no. 9, Sept. 2018, pp. 1287–1300.

[10] C.-B. Chae *et al.*, "Adaptive MIMO Transmission Techniques for Broadband Wireless Communication Systems," *IEEE Commun. Mag.*, vol. 48, no. 5, May 2010, pp. 112–18.

[11] S. Nanda, K. Balachandran, and S. Kumar, "Adaptation Techniques in Wireless Packet Data Services," *IEEE Commun. Mag.*, vol. 38, no. 1, Jan. 2000, pp. 54–64.

[12] R. Fantacci *et al.*, "Adaptive Modulation and Coding Techniques for OFDMA Systems," *IEEE Trans. Wireless Commun.*, vol. 8, no. 9, Sept. 2009, pp. 4876–83.

[13] J. Bicket, "Bit-Rate Selection in Wireless Networks," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[14] W. Kim *et al.*, "An Experimental Evaluation of Rate Adaptation for Multi-Antenna Systems," *Proc. IEEE INFOCOM*, 2009.

[15] I. Pefkianakis *et al.*, "MIMO Rate Adaptation in 802.11n Wireless Networks," *Proc. ACM MobiCom*, 2010.

[16] "The ns-3 Network Simulator," 2010; available: http: // dblp.uni-trier.de/db/books/collections/Wehrle2010.html#RileyH10

Biographies

Kate Ching-Ju Lin received the B.S. degree from the Department of Computer Science, National Tsing Hua University in 2003, and the Ph.D. degree from the Graduate Institute of Networking and Multimedia, National Taiwan University in 2009. She was a visiting scholar at CSAIL, MIT from March 2007 to March 2008 and from October 2010 to March 2011. She is now the Director of the Institute of Network Engineering and a professor in the Department of Computer Science at National Chiao Tung University, Taiwan. Her current research interests include wireless systems, RF-sensing and network architecture for machine learning. She was the recipient of the K. T. Li Young Researcher Award from ACM Taipei/Taiwan in 2014, and was awarded the Research Project for Excellent Young Scholars from the Ministry of Science and Technology, Taiwan, in 2013, 2015, and 2017. She is an IEEE senior member.

Hao-Chen Wang received his B.S and M.S. degrees from the Department of Computer Science, National Chiao Tung University (NCTU), Taiwan, in 2016 and 2018, respectively. His research interests include mobile networks, wireless networks, and software define networking.

Yuan-Cheng Lai received his Ph.D. degree from the Department of Computer and Information Science, National Chiao Tung University in 1997. He joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in August 2001 and has been a distinguished professor since June 2012. His research interests include performance analysis, software-defined networking, wireless networks, and IoT security.

Ying-Dar Lin is a distinguished professor of computer science at National Chiao Tung University (NCTU), Taiwan. He received his Ph.D. in computer science from the University of California at Los Angeles (UCLA) in 1993. His research interests include network security, wireless communications, and network softwarization. His work on multi-hop cellular was the first along this line, and has been cited over 850 times. He is an IEEE Fellow, IEEE Distinguished Lecturer, and Editor-in-Chief of *IEEE Communications Surveys and Tutorials* (COMST).