Dynamic Allocation of Processing Resources in Cloud-RAN for a Virtualised 5G Mobile Network

Yi Zhang[1], Federico Barusso[2,1], Diarmuid Collins[1], Marco Ruffini[1] and Luiz A. DaSilva[1]
[1]CONNECT centre, Trinity College Dublin, Ireland
[2]Politecnico di Torino, Italy
emails: {zhangy8, collindi, dasilval}@tcd.ie, federico.barusso@studenti.polito.it, marco.ruffini@scss.tcd.ie

***Abstract*—One of the main research directions for 5G mobile networks is resource virtualisation and slicing. Towards this goal, the Cloud Radio Access Network (C-RAN) architecture offers mobile operators a flexible and dynamic framework for managing resources and processing data. This paper proposes a dynamic allocation approach for processing resources in a C-RAN supported by the concept of Network Function Vitualisation (NFV). To achieve this objective, we virtualised the Baseband Unit (BBU) resources for Long Term Evolution (LTE) mobile network into a BBU pool supported by Linux Container (LXC) technology. We report on experiments conducted in the Iris testbed with high-definition video streaming by implementing Software-Defined Radio (SDR)-based LTE functionality with the virtualised BBU pool. Our results show a significant improvement in the quality of the video transmission with this dynamic allocation approach.**

***Index Terms*—C-RAN, NFV, container, 5G, testbed**

I. Introduction

C-RAN is a mobile network architecture where the BBUs of a group of base stations are separated from the Remote Radio Heads (RRHs), and processing is moved to a centralised BBU pool in a cloud-based computing centre [1]. NFV functionality can be implemented at the C-RAN central office, with the support of dynamic resource allocation. This enables the flexible assignment of different amounts of computational resources to different BBUs. In this way, computational resources for each BBU can be tailored to the requirements of applications in real time, in order to optimize overall resource utilisation. It is hard to achieve this dynamic allocation with Application-specific Integrated Circuits (ASICs). Therefore, more flexible BBU implementations in software have emerged to bring flexibility and dynamicity.

In this paper we design and implement an SDR-based prototype to dynamically allocate computational resources in a BBU pool for C-RAN. The objective is to demonstrate its benefit in terms of an increase in the performance of high throughput applications, e.g. High Definition (HD) video streaming. We use LXC technology to achieve NFV on a Linux-based BBU-pool system.

The connection between BBUs and RRHs in a C-RAN is referred to as the mobile *fronthaul* [2]. The mobile fronthaul in C-RAN can be implemented by a wireless-optical integrated network, where the optical access network (in particular, Passive Optical Networks (PON) [3]) is used for connecting the BBUs and RRHs. The optical fibre fronthaul system is capable of providing more bandwidth than traditional copper systems, meeting the high bandwidth-consuming I/Q sample transmission for the C-RAN.

This paper reports on an experiment that illustrates how dynamic allocation of processing resources improves the performance of C-RAN, especially for high-intensity processing applications. We use an HD video stream via an LTE base station as an example of a processing-intensive application. In order to transmit HD videos over the air, the BBUs for the LTE base station require a higher Modulation and Coding Scheme (MCS) index, corresponding to higher spectral efficiency and throughput. The higher MCS index also brings higher computational complexity to the BBUs. Our dynamic allocation scheme can assign more CPU resources to the BBU pool in real time without breaking the LTE transmission links. Furthermore, when the traffic demand decreases, we can release computational resources originally assigned to BBUs, and reduce power consumption.

This paper is organized as follows: Section II discusses the use cases and the background for this work, including a brief literature review. Section III introduces the Iris testbed at Trinity College Dublin, which is one of the testbeds participating in the FUTEBOL project and supports this experiment. Section IV discusses the experimental results for our proposed dynamic resource allocation approach. Section V concludes this paper and discusses potential future work.

II. Use cases and background

Virtualisation and resource sharing are expected to play important roles for flexible usage of network resources in next generation mobile networks [4], adopting the concept of NFV [5]. To fulfil the system requirements of different verticals (i.e., different use cases in terms of network design and planning), multiple isolated virtualised network functions have to be selected and connected together. Therefore, research problems such as resource allocation and load balancing emerge to address the challenge of high computational requirement and limited amount of resources.

The centralisation of BBUs allows for the flexible control and management of virtualised resources. When the BBU processing is moved away from the antennae, the traditional backhaul of the mobile base station system changes from transmitting backhaul data to transmitting I and Q samples that are obtained/sent directly from/to antennae after/before AD/DA conversion. These changes, which take place on the transmissions between BBUs and RRHs for C-RAN, represent a new transmission system called the mobile *fronthaul* [2].

In this paper, we investigate the adoption of the NFV concept for C-RAN processing. The BBU pool is virtualised and each individual BBU can be implemented in software and encapsulated inside a container. Multiple containers share the computational resources of one physical machine. These containers are supported by a cloud-based virtual machine system and an LXC hypervisor, implemented in the SDR Iris testbed in Trinity College Dublin [6]. The containerised BBUs form a pool that can be controlled by a central controller. The controller is capable of dynamically reallocating resources to each BBU container running on the same physical machine.

These LXC containers, whose computational resources are capable of being reassigned dynamically (as introduced in Section IV), can also support the migration from one virtual machine to another (or even from one physical machine to another) in real time. This functionality, called live migration of containers, can support dynamic network function exchange. The live migration of BBU elements is challenging to implement in real time due to the strict latency requirement for baseband processing in the LTE standard. However, live migration of Evolved Packet Core is feasible, enabling flexible management of the LTE backhaul.

Several recent works in the literature address virtualisation and resource allocation in C-RAN. For instance, authors in [7]–[10] all focus on the resource allocation for C-RAN, especially virtualised computational resources for a BBU pool. However, none of these works has performed an actual implementation-based investigation into the virtualisation and dynamicity of resource allocation required. In this paper, in contrast, we have built an SDR-based C-RAN prototype to demonstrate and evaluate the performance of dynamic allocation of computing resources in response to the requirements of each BBU.

III. FUTEBOL TESTBED INFRASTRUCTURE

The FUTEBOL project [11] focuses on experimental research in the area of wireless-optical network convergence. Another important goal of the project is to make advanced wireless and optical research testbeds across European and Brazilian institutions federated and open to external scientific researchers. Towards these goals, the project is developing a control framework that supports the orchestration and management of physical and virtualised network resources in both the wireless and optical domains, such as virtual machines, SDN switches, optical devices, and USRP reconfigurable radios in multiple federated testbeds in parallel. In the Iris testbed, at Trinity College Dublin, we are contributing to this objective by exploring the dynamic reallocation of available computational resources to containers on demand.

A. Iris Testbed

Iris, the reconfigurable radio testbed at Trinity College Dublin, offers virtualised radio hardware to support the experimental investigation of resource allocation and orchestration in future wireless networks. This facility pairs flexible radio equipment and computational resources with various hypervisors in the form of software-defined radio frameworks to realize different testing configurations. These platforms are connected to a computational cloud, allowing users to deploy an array of computational environments.

Iris can be thought of as having five functional layers, as illustrated in Fig 1. These include:

- Functional elements: The bottom layer provides the physical resources, such as servers, switches, USRPs, storage, CPU, memory, and so forth.
- Virtualised experimental resources: The next layer corresponds to the virtualized testbed resources allocated to the experimenter for a specific time frame. It is composed of virtual machines images, real server resources, and physical radio equipment.
- Hypervisor: To expose the functionality of physical equipment for applications, Iris employs a variety of hypervisors including the Kernel-based Virtual Machine (KVM), and LinuX Containers (LXC). This layer provides support for open source SDR elements such as GNU Radio and srsLTE [12], an open-source implementation of the 3GPP LTE base stations and User Equipments (UEs). This layer also provides support for lightweight LXC, LXD, and Docker containers.
- Slices: This layer abstracts the slicing taking place on the network by SDN controllers, at the radio by the SDRs, and at the operating system by hypervisors.
- Experiments: The final layer sits at the top and is defined by the experimenter. It provides the experiment definition that will utilise the resources offered by the lower layers.

Iris's Cloud Based Testbed Manager (CBTM) allocates experimentation units across these functional layers, by supporting the creation of virtual machine images in physical rack servers. The principal responsibility of the CBTM is to instantiate available virtual machines and USRP radios in servers, where users can run experiment-specific software. Furthermore, Iris's CBTM facility offers support for other operating-system-level virtualization frameworks. This is achieved by providing experimenters with a virtual machine image with hypervisor capabilities.

For example, experimenters can use the LXC hypervisor to run lightweight Linux containers, which is available as a virtual machine image at Iris. Experimenters have complete control over features including the LXC Linux c-groups (control groups) kernel, which isolates, limits, and provides accounting for resource usage, and the namespace Linux kernel feature, which supports the complete isolation of virtualised system resources from containers including the process tree, filesystems, hostnames, process IDs, and network access. Both are fundamental for the support of containers on Linux.

The dynamic processing resource allocation research problem investigated in this paper illustrates some of the benefits of reallocating system resources to containers by modifying c-group rules on the LXC hypervisor.

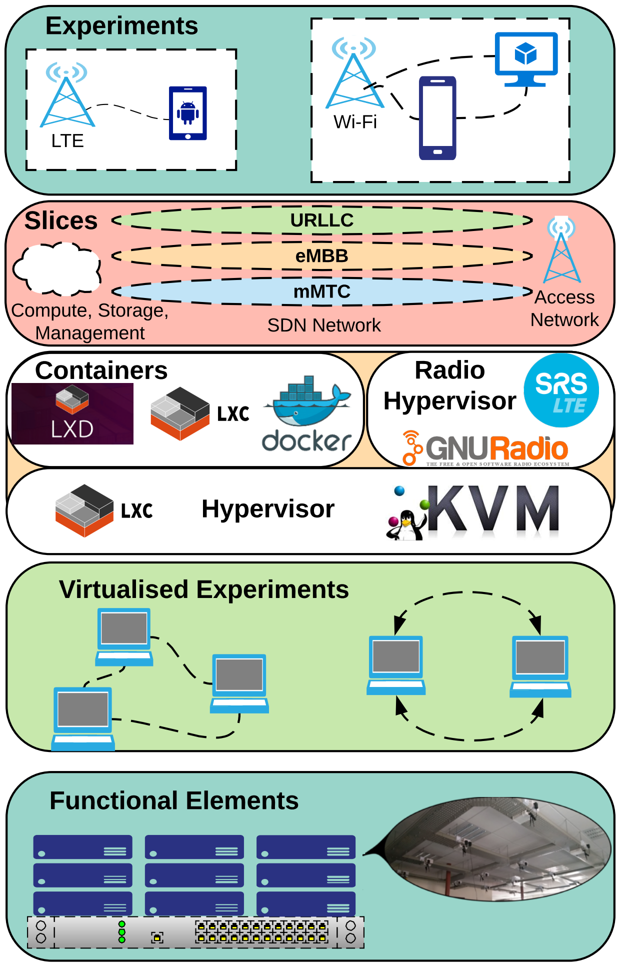

Fig. 1. Iris testbed functional layers.

IV. Experiment and Results

We have designed and implemented a video-streaming experiment running in the Iris SDR testbed to evaluate the benefit of dynamic resource allocation for a virtualised BBU pool. The BBU pool is installed in containers and the computational resources (Central Processing Unit (CPU) cores, or CPUs for short in this paper) of the BBU pool can be dynamically allocated. As shown in Fig. 2, we have implemented two SDR-based LTE BBUs inside an LXC container as a prototype for experimentation. The SDR-based LTE BBU adopts the open-source srsLTE software, which provides physical layer downlink data transmission from eNodeB to UE. The two RRHs are implemented in USRP B210s [13], which are programmable and reconfigurable radios. Two RRHs work in different frequency channels (2.4 GHz and 2.48 GHz), with a sufficiently wide guard band to eliminate interference that might affect the results. The physical transmission link between the BBUs and RRHs is USB 3.0, playing the part of the fronthaul. The throughput of USB 3.0 can reach up to 5 Gbps, meeting the LTE sampling rate requirement. The USB physical links can potentially be replaced by an optical fibre transmission system, in order to increase the fronthaul bandwidth. This is planned as future work.

As shown in Fig. 2, we stream high definition video over the air to evaluate the performance of the container-based BBU system. The video file is converted into a data stream by the software “avconv” [14], and sent through a local TCP port to the srsLTE software (i.e., the LTE BBU). The BBU is a single thread process and cannot be parallelised to utilise two processors. In other words, in our system, one BBU process can be assigned to only one CPU. Therefore, in order to observe the effect of under-loading/over-loading of computational resources, in our experiment we develop two BBUs inside one LXC container, and assign one or two CPUs to this container to observe the performance changes. During our experiment, we have observed that, once an extra CPU is assigned to the container, one out of the two BBU processes that are running on the previous CPU will automatically move to the new assigned CPU, thanks to the fairness policy that we set for load balancing of each CPU. The whole procedure only takes 12 milliseconds.

On the other side of the testbed, two UEs are also implemented by SDR with srsLTE software and USRP B210s. The software “avplay” is installed on the hosting machines of the UEs to play the received video stream, as well as to collect the statistics of video streaming, e.g., received data, frames dropped, etc. [15].

In order to stream videos with high definition, more throughput is needed from the LTE network. If we fix the number of Physical Resource Blocks (PRBs) assigned to the wireless channel, one way to increase the throughput is to increase the MCS index to higher orders. According to the 3GPP LTE standards, the MCS index defines the Transport Block Size (TBS) (in bits), and the throughput of the LTE links can be derived from the TBS by the following equation [16]: $Throughput = TBS * 1000/s$.

However, different levels of MCS index also affect the computational load of the SDR-based BBU. Therefore, if the computational resources are not sufficient to support the baseband processing of high MCS, the SDR system will be overloaded and the data will not be successfully transmitted over the air interface. In this case, frame dropping will be observed at the receiver side. The srsLTE SDR system allows us to change the MCS in real time (i.e. less than a subframe length (1ms) without any data loss). Therefore the computational load of a BBU can be changed in real time by changing the MCS index, providing the opportunity for us to investigate the benefit brought by the dynamic resource allocation. The dynamic resource allocation can balance the computational load for the BBUs in a BBU pool, so that high throughput can be guaranteed without any blocking that would be caused by overloaded CPUs.

Based on the testbed setup shown in Fig. 2 and the aforementioned concepts regarding MCS index, throughput and computational resources, we have designed a video streaming experiment for a duration of 100 seconds to test the performance of dynamic resource allocation. Table I describes the procedure for the experiment. From 0 to 60 seconds,

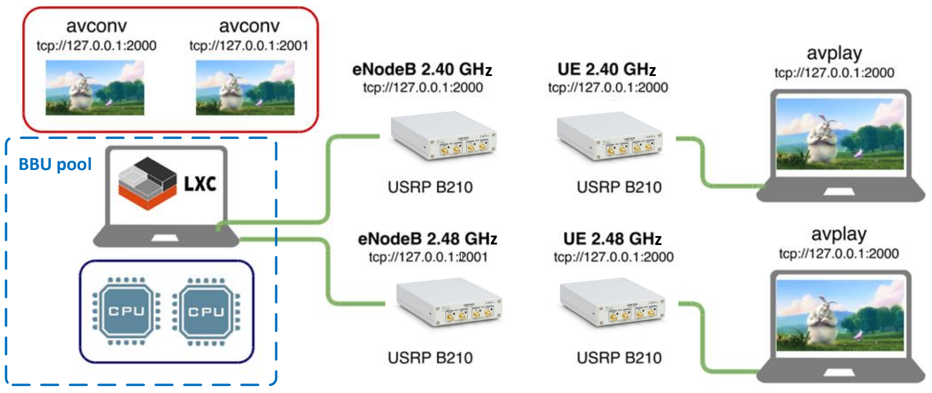

Fig. 2. The experiment setup for the dynamic resource allocation of BBU pool.

we gradually increase the MCS index every 20 seconds and therefore the throughput increases. However, the CPU that is assigned to these two BBUs becomes overloaded and buffering interruptions start occurring on the video stream. Then we add an extra CPU to the container at the 80th second to determine whether the performance can be improved.

The block diagram of the experiment for dynamic processing resource allocation is shown in Fig. 3. We implement the timer that defines the timing for changing the allocation of CPUs to the container. It talks to the CPU resource scheduler that we implemented in Python and Linux shell scripts to make the changes. The scheduler is capable of reassigning different numbers of CPUs to the container. Furthermore, we have implemented the BBUs and USB 3.0 interface bridges inside the container in order to make the physical hardware interfaces accessible from the containers, so that the virtualised BBUs inside the container are capable of exchanging data with the RRHs through USB 3.0 (i.e. the fronthaul).

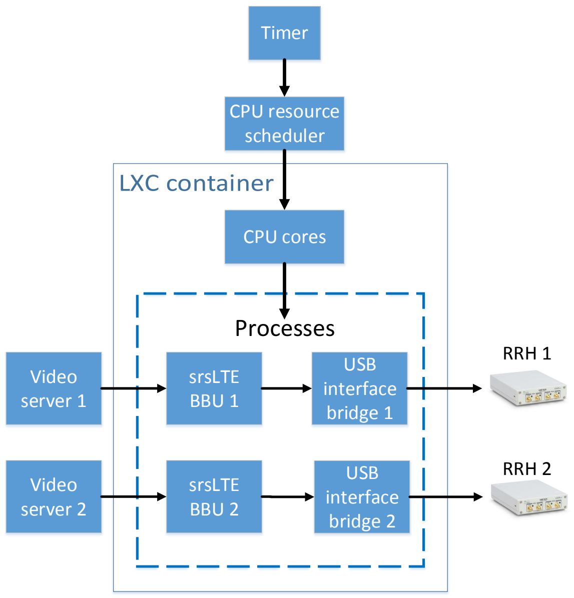

Fig. 3. Block diagram of the experiment.

TABLE I
THE EXPERIMENTAL PROCEDURE OF CHANGING MCS INDEX AND NUMBER OF CPUS ASSIGNED TO THE BBU POOL

Time (s)	MCS index	Modulation	TBS(bit)	No. of CPUs
0	4	QPSK	1800	1
20	10	16QAM	4008	1
40	17	64QAM	7736	1
60	24	64QAM	13536	1
80	24	64QAM	13536	2

Fig. 4 shows, as a function of experiment time, the amount of data successfully received by one of the two UEs through the air interface, while Fig. 5 shows the number of frame drops for the same UE. We show results for only one UE, as the results for the other are nearly identical. As described in Table I, the parameter changing period is every 20 seconds (s). We use 1080p, 60 fps video for streaming. The air interface utilises 25 PRBs.

Results show that during the 0-10 seconds period, the video data is successfully transmitted, received and buffered at the receiver side. The amount of data buffered at the UE during the 0-10 seconds period is enough for the video player to play the whole 0-20 seconds period without any frame drops: the video playing is smooth. Therefore, during the 10-20 seconds period there is no data received by the UE but the video keeps playing. Then we change the MCS index from 4 (QPSK) to 10 (16QAM) at t = 20 seconds: some frame drops are observed from 20-23 seconds due to the re-synchronization between UE and eNodeB when the MCS index changes in the LTE software. During the 23-40 seconds period, most of the video data is still successfully received and buffered and the video playing goes on, albeit with some discontinuities of video frames starting from t = 30 seconds. This is due to the fact that more computational resources are needed for 16QAM, compared to QPSK, and the CPU starts to become overloaded. Note that frame drops occur between 30-40 seconds. There is no frame dropping between 20-30 seconds because some data is pre-buffered during the 0-20 seconds period.

During the periods of 40-60 and 60-80 seconds we continue to increase the MCS index from 10 (16QAM) to 17 (64QAM), and then to 24 (64QAM, but with higher TBS). More intensive computation is required for the higher modulation order, and the single CPU being used for both BBUs gets heavily overloaded and the UE can barely receive any data. The video streaming almost stops during these two time periods. Finally, from t = 80 seconds onwards we assign an additional CPU to the BBU pool, so that one BBU is shifted to the new CPU, and the two BBUs are working on separate CPUs. In this case the computational load for each CPU drops significantly. Therefore, after a couple of seconds transition period, from around t = 84 seconds onwards no frames are dropped and the data transmission completes successfully. The allocation of an additional CPU resource takes only 12 milliseconds but the transition period is as long as 4 seconds, due to the re-synchronization between UE and eNodeB in the LTE software, which causes some packet loss. We will strive to shorten the transition time in our future work by improving the SDR-based LTE BBU system.

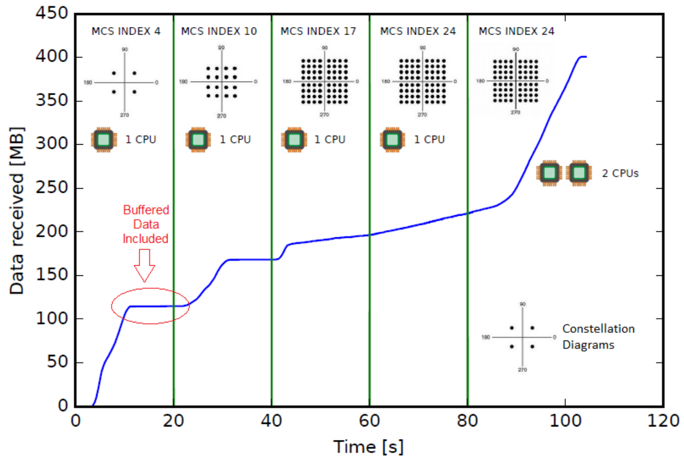

Fig. 4. Received data during the experiment.

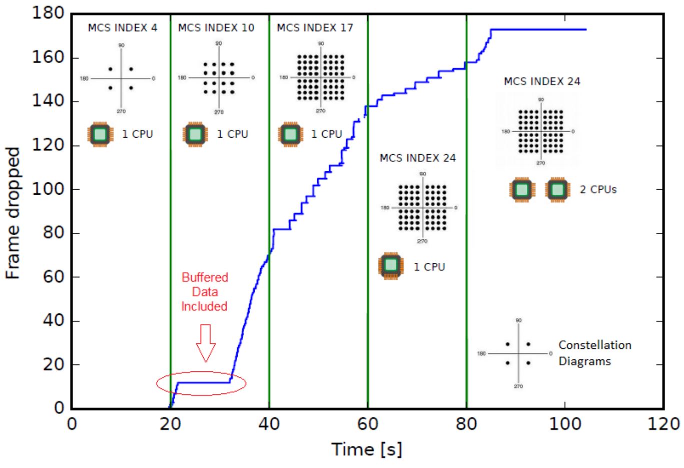

Fig. 5. Data frames dropped during the experiment.

V. Conclusions and Future Work

In this paper we describe the implementation and evaluation of dynamic processing resource allocation for the BBU pool of a C-RAN. We use the concept of NFV to manage the BBU pool by implementing the BBUs inside LXC containers, in our Iris SDR testbed. An experimental evaluation of HD video transmission using different modulation orders shows the benefit of dynamic resource allocation for balancing the load of processors.

Under the framework of the FUTEBOL project, more experiments related to network virtualisation are planned. In addition to the dynamic processing and resource allocation approach introduced in this paper, we are also investigating the feasibility and use cases for low-latency live migration of containers. The implementation of live container migration will offer additional opportunities to support NFV in more verticals.

Acknowledgment

This publication has emanated from the research supported by the European Commission Horizon 2020 under grant agreements no. 688116 (eWINE) and no. 688941 (FUTEBOL).

References

[1] A. Checko, H. L. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. S. Berger, and L. Dittmann, "Cloud RAN for Mobile Networks - A Technology Overview," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 405–426, Firstquarter 2015.
[2] T. Pfeiffer, "Next generation mobile fronthaul and midhaul architectures [invited]," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 11, pp. B38–B45, November 2015.
[3] K. Tanaka and A. Agata, "Next-generation optical access networks for c-ran," in *2015 Optical Fiber Communications Conference and Exhibition (OFC)*, March 2015, pp. 1–3.
[4] L. Doyle, J. Kibida, T. K. Forde, and L. DaSilva, "Spectrum without bounds, networks without borders," *Proceedings of the IEEE*, vol. 102, no. 3, pp. 351–365, March 2014.
[5] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "Nfv: state of the art, challenges, and implementation in next generation mobile networks (vepc)," *IEEE Network*, vol. 28, no. 6, pp. 18–26, Nov 2014.
[6] CONNECT centre, "Iris - The Reconfigurable Testbed," Accessed: Feb. 2018. [Online]. Available: https://iris-testbed.connectcentre.ie/
[7] I. Al-Samman, M. Artuso, H. Christiansen, A. Doufexi, and M. Beach, "A framework for resources allocation in virtualised c-ran," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sept 2016, pp. 1–7.
[8] F. Zhang, J. Zheng, Y. Zhang, and L. Chu, "An efficient and balanced bbu computing resource allocation algorithm for cloud radio access networks," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, June 2017, pp. 1–5.
[9] S. K. S. Tyagi, T. Lin, and Y. Zhou, "Thermal-aware dynamic computing resource allocation for bbu pool in centralized radio access networks," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, June 2017, pp. 1–5.
[10] N. Yu, Z. Song, H. Du, H. Huang, and X. Jia, "Multi-resource allocation in cloud radio access networks," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.
[11] FUTEBOL, "Federated Union of Telecommunications Research Facilities for an EU-Brazil Open Laboratory," Accessed: Feb. 2018. [Online]. Available: http://www.ict-futebol.org.br
[12] srsLTE, "Open source 3GPP LTE library," Accessed: Feb. 2018. [Online]. Available: https://github.com/srsLTE/srsLTE
[13] Ettus Research. USRP B210 USB Software Defined Radio (SDR). Accessed: Feb. 2018. [Online]. Available: https://www.ettus.com/product/details/UB210-KIT
[14] avconv, "documentation," Accessed: Feb. 2018. [Online]. Available: https://libav.org/avconv.html
[15] avplay, "documentation," Accessed: Feb. 2018. [Online]. Available: https://libav.org/documentation/avplay.html
[16] 3GPP, "LTE Release 12," Accessed: Feb. 2018. [Online]. Available: http://www.3gpp.org/specifications/releases/68-release-12