Cloudification and Slicing in 5G Radio Access Network

Chia-Yu Chang

Sorbonne University

Doctoral School of
Informatics, Telecommunications and Electronics of Paris

EURECOM

Cloudification and Slicing in 5G Radio Access Network

Presented by Chia-Yu CHANG

Dissertation for Doctor of Philosophy in
Information and Communication Engineering

Directed by Navid NIKAEIN

Publicly presented and defended on 29 November 2018

A jury committee composed of:

Prof. Raymond KNOPP	EURECOM	President of the jury
Prof. Akihiro NAKAO	University of Tokyo	Reporter
Prof. Mahesh K. MARINA	University of Edinburgh	Reporter
Prof. Rami LANGAR	UPEM	Examiner
Prof. Thi-Mai-Trang NGUYEN	Sorbonne University	Examiner
Prof. Thrasyvoulos SPYROPOULOS	EURECOM	Examiner
Prof. Anna TZANAKAKI	University of Athens	Examiner
Prof. Navid NIKAEIN	EURECOM	Thesis director

Sorbonne Université

École Doctorale
Informatique, Télécommunications et Électronique de Paris

EURECOM

Cloudification et découpage des réseaux d'accès radio de cinquième génération

Par Chia-Yu CHANG

Thèse de doctorat en
Sciences de l'Information et de la Communication

Dirigée par Navid NIKAEIN

Présentée et soutenue publiquement le 29 novembre 2018

Devant un jury composé de:

Prof. Raymond Knopp	EURECOM	Président du jury
Prof. Akihiro Nakao	Université de Tokyo	Rapporteur
Prof. Mahesh K. Marina	Université d'Édimbourg	Rapporteur
Prof. Rami Langar	UPEM	Examinateur
Prof. Thi-Mai-Trang Nguyen	Sorbonne Université	Examinateur
Prof. Thrasyvoulos Spyropoulos	EURECOM	Examinateur
Prof. Anna Tzanakaki	Université d'Athènes	Examinateur
Prof. Navid Nikaein	EURECOM	Directeur de thèse

Abstract

The forthcoming fifth generation (5G) mobile networks is ambitious to evolve a broad variety of capabilities for the intended usage scenarios. To facilitate this vision, a paradigm shift is provided in 5G to establish a flexible and customizable communication system. Essentially, the 5G architecture shall be designed with a certain level control flexibility via incorporating the principles of softwarization and virtualization to turn a physical network into multiple customized end-to-end (E2E) logical network tailored to multiple service requests. Correspondingly, two key pillars are highlighted and investigated for the 5G network: slicing and cloudification. These two techniques are mostly challenging in the radio access network (RAN) domains due to its stringent requirements among real-timeliness, computing complexity and energy performance. To this end, this thesis is organized in tow parts to investigates these two techniques on the RAN domain.

In the first part, we investigate the cloudification of the RAN, i.e., cloud RAN (C-RAN), in which the centralized RAN processing in a cloud can support efficient resource multiplexing and joint multi-cell processing. Despite its appealing, the C-RAN concept faces challenges in terms of the severe capacity and latency requirements of the fronthaul (FH) interface that connects distributed remote radio unit (RRU) toward the centralized baseband processing unit (BBU). We first investigate the impacts among several factors, such as functional split, packetization and packet scheduling, on the Ethernet-bsaed FH transportation to increase multiplexing gain. Further, two implemented functional splits over the OpenAirInterface (OAI) platform are presented and compared in terms of several KPIs to justify the C-RAN applicability. We also present a model and an analytical framework for the E2E RAN service delivery to maximize the network spectral efficiency by considering multiple design factors.

In the second part, we focus on the RAN slicing to not only provide different levels of isolation and sharing to each slice but also enable the customization across control plane (CP), user plane (UP), and control logic (CL). Hence, we propose a flexible execution environment, denoted as the RAN runtime slicing system, in order to (a) virtualize slice service instances over the underlying RAN modules, and to (b) abstract radio resource to increase the multiplexing gain. Additionally, a number of new interfaces are identified for the communications between the RAN runtime and the orchestration system in support of slice-based multi-service chain creation and chain placement, with an auto-scaling mechanism to increase the performance. Finally, the runtime software development kit (SDK) is on top of the RAN runtime introduced to facilitate the development of control applications. These control applications can compose sophisticated and customized control logics that can be chained across different domains and planes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Evolution toward 5G mobile networks

Over the last few decades, a gradual yet steady advancement of mobile wireless network is witnessed by the world from the second, third and fourth generation (2G/3G/4G) wireless networks. This evolution already brought several technology enablers, such as multi-user multiplexing, high-order digital modulation, ultra-broadband communications, heterogeneous frequency bands deployment, multiple antenna techniques, and packet-based Internet accessibility, to serve a variety of services ranging from voice calls, text messages, Internet browsing to multimedia streaming. Nevertheless, the continuous growth of network statistics requests an ever-evolving technology landscape. For instance, according to Cisco's visual networking index forecast in [2], the overall mobile data traffic will grow to 49 exabytes per month by 2021, showing a seven-fold increase since 2016. Such high demands is driven by the surge of smartphones devices as well as machine-to-machine (M2M) connections. To this end, one natural question comes up in our minds: *what will fifth generation (5G) be* [3]?

To properly answer this question, we first review the required 5G capabilities from the global stakeholders perspective in Figure 1.1, provided by the International Telecommunication Union radiocommunication sector (ITU-R). Like the International Mobile Telecommunications-2000 (IMT-2000) [4] and IMT-Advanced [5] requirements respectively for the 3G and 4G systems, the IMT-2020 vision [1] requests significant improvements among several capabilities in 5G. These broad variety of capabilities is tightly coupled with the intended usage scenarios and applications. Specifically, three key usage scenarios can be pointed out [6, 7] as:

1. Enhanced mobile broadband (eMBB)[1],
2. Ultra-reliable and low latency communication (uRLLC)[2], and
3. Massive machine-type communication (mMTC).

First, the eMBB scenario demands a higher amount of importance among the highlighted six capabilities in Figure 1.1 (i.e., peak data rate, spectrum efficiency, user experienced data rate, area traffic capacity, network energy efficiency and mobility); however, not all of them have equal importance simultaneously in all cases. For instance, a higher user experienced data

[1]It is also known as extreme mobile broadband (xMBB) [8].
[2]It is also known as ultra-reliable MTC (uMTC) [8]

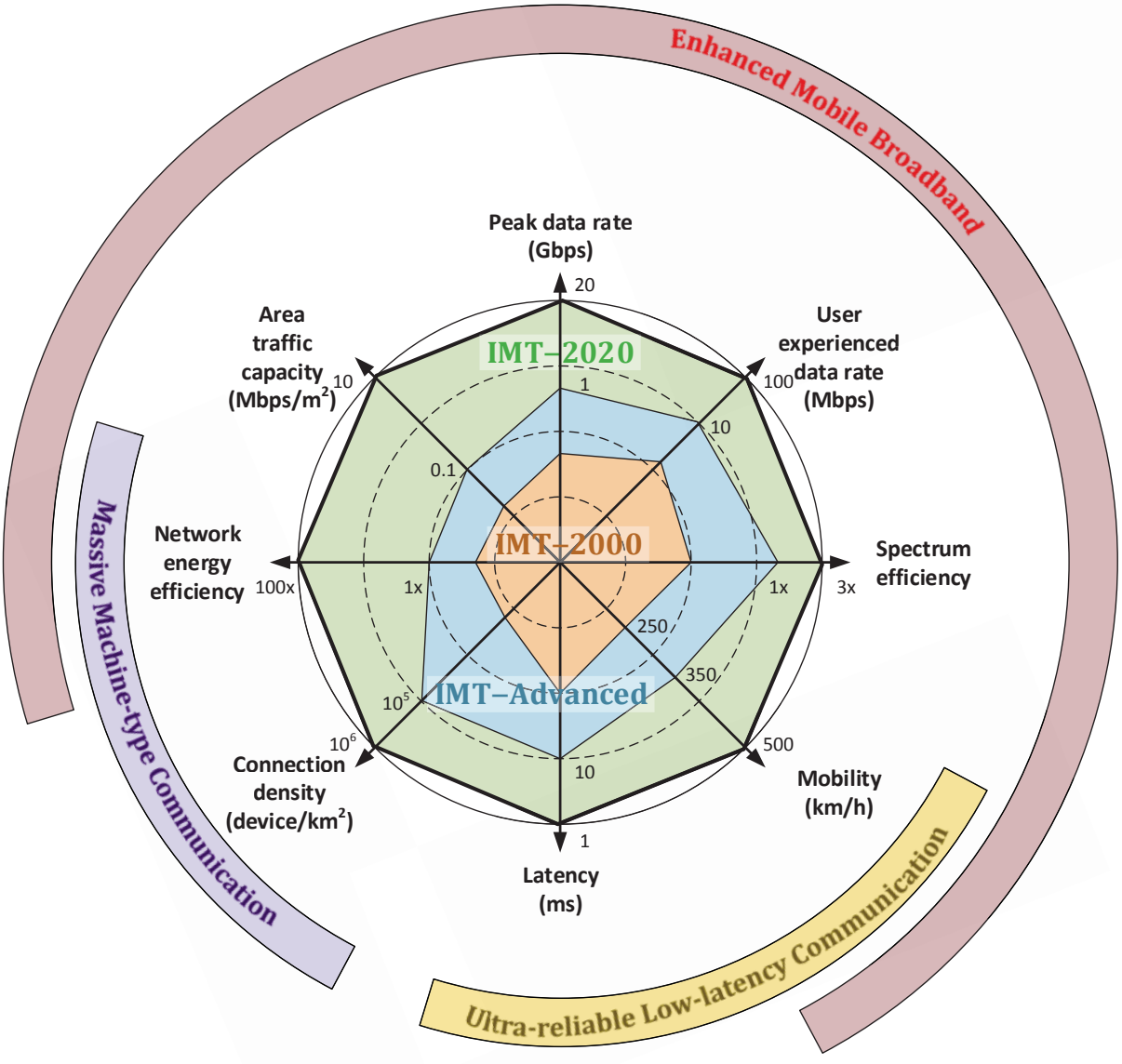


Figure 1.1: IMT-2020 vision on 5G

rate with pedestrian mobility case is shown for the hotspots, when compared with a wider coverage area like the rural region. As for the uRLLC scenario, the latency shall be reduced by a factor 10 and is of the highest importance to enable the safety critical applications, such as intelligent transport systems, smart grids, wireless industrial automation, and remote tactile interaction. Also, such capability is required in a high mobility case, e.g., up to 500 kilometer per hour (km/hr), while the data rates may be less important. Finally, the mMTC scenario anticipates a high connection density to support a tremendous number of devices that may transmit occasionally with a relatively low bit rate and non-delay-sensitive data under few or zero mobility. Note that the network energy efficiency is also vital for mMTC to enable the deployments of low cost devices with long operational lifetime.

Table 1.1: IMT-2020 radio interface minimum technical performance requirements [1]

Requirements	**Indoor Hotspot eMBB**	**Dense Urban eMBB**	**Rural eMBB**	**Urban Macro uRLLC**	**Urban Macro mMTC**
Peak date rate	DL: 20 Gbps, UL: 10 Gbps			-	-
Peak spectral efficiency	DL: 30 bps/Hz, UL: 15 bps/Hz (8 spatial layers in DL and 4 spatial layers in UL)			-	-
User experienced data rate	-	DL: 100 Mbps UL: 50 Mbps	-	-	-
5^{th} percentile user spectral efficiency	DL: 0.3 bps/Hz UL: 0.21 bps/Hz	DL: 0.225 bps/Hz UL: 0.15 bps/Hz	DL: 0.12 bps/Hz UL: 0.045 bps/Hz	-	-
Average spectral efficiency	DL: 9 bps/Hz UL: 6.75 bps/Hz	DL: 7.8 bps/Hz UL: 5.4 bps/Hz	DL: 3.3 bps/Hz UL: 1.6 bps/Hz	-	-
Area traffic capacity	DL: 10 Mbps/m^2	-	-	-	-
User plane latency	1 ms			4 ms	-
Control plane latency	20 ms transition from the Idle state to the Active state (Encourage to consider 10 ms control plane latency)				- -
Connection density	-	-	-	-	10^6 device/km^2
Energy efficiency	1. Low energy consumption when there is no data 2. Continuous sleep duration shall be sufficient long			-	-
Reliability	-	-	-	1-10^{-5} success probability for 32-byte Layer 2 packet data unit	-
Mobility (UL link data rate)	10 km/h (1.5 bps/Hz)	30 km/h (1.12 bps/Hz)	120km/h (0.8bps/Hz), 500km/h (0.45bps/Hz)	-	-
Mobility interruption time	0 ms				-
Bandwidth	At least 100 MHz, support up to 1 GHz in higher frequency bands (e.g., above 6 GHz)				

In order to cope with the aforementioned 5G requirements, the proposals for 5G are to be propounded to ITU-R and will be evaluated in five respective test environments [9] before the first half of 2020, i.e., Indoor Hotspot-eMBB, Dense Urban-eMBB, Rural-eMBB, Urban Macro-uRLLC, and Urban Macro-mMTC, as summarized in Table 1.1. Afterwards, the ITU-R recommendation is expected to be finalized in the second half of 2020 [10]. In correspondence, the standardization process of the third generation partnership project (3GPP) is alongside the ITU-R schedule. In practical, the 3GPP standardization process for 5G is decomposed into 2 phases allowing the expected deployments in the 2020 timeframe: *Phase 1* aims to provide the initial 5G standard as Release 15 by September 2018 for both non-standalone and standalone operations [11], and *Phase 2* will continue a detailed specification and support more services (e.g., communications in vertical domains and vehicle-to-everything [V2X]) by the end of 2019 as Release 16. Note that the non-standalone operation can provide 5G in combination with the existing 4G long term evolution (LTE)/LTE-Advanced systems to maintain the stable coverage areas, while the standalone operation shows a clean slate replacement for current 4G system (i.e., like the transition from 3G to 4G). Furthermore, the Institute of Electrical and Electronics Engineers (IEEE) also undertakes a 5G track to enhance existing and new IEEE technologies, such as IEEE 802.11ax/ay (WLAN), IEEE 1588 (Precision time protocol [PTP]), IEEE P1914.3 (Radio over Ethernet [RoE]) and IEEE P1918.1 (Tactile and haptic networking); however, the timelines for completion may vary in different specification groups [12].

Several key technologies are now becoming widely surveyed to enable the anticipated pathway toward the 5G as summarized in [7, 13–16], in which the three most crucial ones, including millimeter-wave (mmWave), network densification, and massive multiple-input multiple-output (MIMO), are concluded by [7]. These technologies can be utilized together within the 5G architecture but will incur some correspondingly challenges. For instance, the network densification seems promising to provide improvements among area traffic capacity and connection density; however, the corresponding challenges arise straightforwardly, such as (1) the proper coordination among densely-deployed base stations (BSs), and (2) the efficient resource utilization to support multiple network services (i.e., eMBB, uRLLC and mMTC) with the acceptable network capital expenditure (CAPEX) and operating expense (OPEX). To deal with these challenges, the 5G architecture shall be designed with a certain level of flexibility via incorporating two essential principles [17, 18]: *softwarization* and *virtualization*. The former can decouple control plane (CP) processing from user plane (UP) processing, while the latter facilitates the instantiation of several network functions over a common infrastructure. Examples of these two principles are broadly known as software-defined networking (SDN) [19] and network function virtualization (NFV) [20] techniques. These two principles are also highlighted by several standardization bodies and industry forums for establishing the 5G architecture, like the fifth generation public-private partnership (5GPPP) [21], 3GPP [22, 23], next generation mobile network (NGMN) alliance [24] and ITU telecommunication standardization sector (ITU-T) [25].

According to the aforementioned design principles, we can observe that 5G will provide a paradigm shift beyond just several technologies to establish a flexible and customizable communication system. Unlike several previous generations, 5G not only provides specific radio access technology but also utilizes the available authorized spectrum bands (e.g., mmWave, sub-6 GHz, sub-1 GHz) and access technologies (e.g., Wi-Fi, 4G, 5G) in a heterogeneous manner via leveraging the cloud computing, SDN and NFV technologies [26, 27]. In this sense, multiple network services can be provided on top of the 5G architecture via composing each customized end-to-end (E2E) logical network in a flexible manner, including endpoints (e.g., user equipment), cloud and edge infrastructures (e.g., x86 infrastructure, software-defined radio), physical and virtual network functions (PNFs/VNFs) across multiple domains (e.g., radio access network [RAN], core network [CN], transport network [TN]), and the management and orchestration functionalities (e.g., 5G operating system [OS] [26]).

An overview of 5G architecture is provided in Figure 1.2. Via applying the NFV and SDN techniques, the underlying clouds and infrastructures can be virtualized and softwarized to compose the customized UP and CP processing across several domains. Moreover, the control logic (CL) of each service can be programmable in a customized manner through the control applications at the application plane to provide the E2E quality of experience (QoE). To facilitate this QoE provisioning, the E2E logical network shall be orchestrated and managed dynamically according to the service requirements (from the service providers), the monitored key performance indicators (KPIs), and hte underlying managed infrastructures. In summary, one paramount capability of 5G system is to provide an E2E framework tailored to the requirements of a multitude of novel network services.

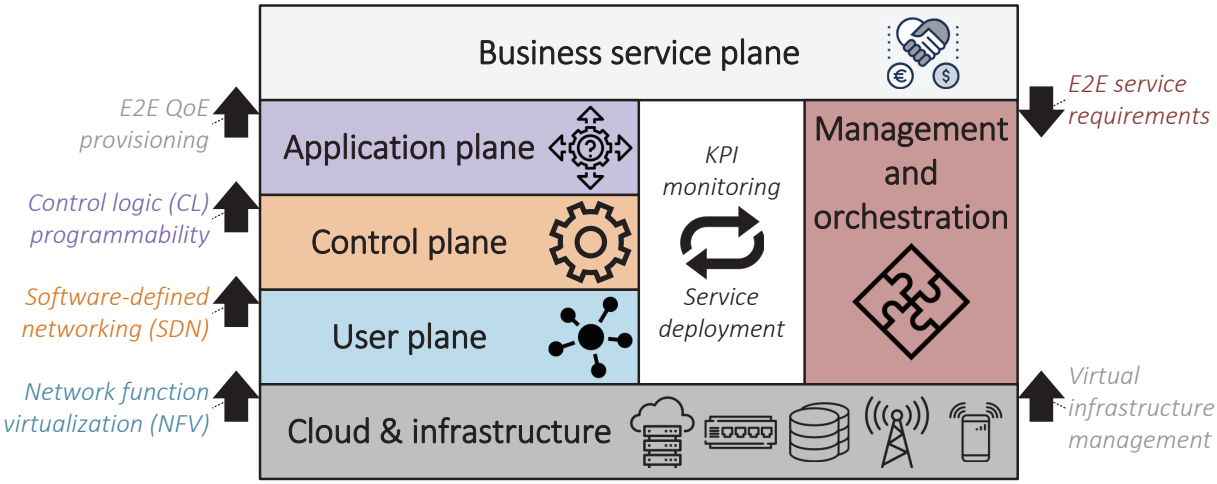


Figure 1.2: Overview of 5G architecture

1.2 Challenges and related work

Based on the aforementioned evolution toward the 5G mobile network, the overall E2E network is built across several domains in respect to the corresponding services. More practically, multiple logical network are established from the virtual network on top of the underlying physical infrastructures as shown in Figure. 1.3, compared with the monolithic solution provided by the previous generations (e.g., 4G). In more details, these virtual networks are built using the NFV technology to reuse the underlying physical infrastructures and to offer the network deployment flexibility, while the logical network can use the SDN technology to decouple the CP functionality from the UP data transportation and to customize service-specific control and management functionalities. Also, the network can be used efficiently and independently via crafting the logically-separated space for each service, hereafter termed *network slice*, in an E2E manner from the endpoint devices, network-edge infrastructures, to distributed and/or centralized clouds and infrastructures. Note that each logical network can leverage the *cloudification* technique to deploy its network functions in a flexible cloud environment tailored to the specific service requirements, e.g., the data centers at edge, local and distant region in Figure 1.3.

To go one step further, we can notice that the above two techniques, i.e., cloudification and slicing, are especially challenging in the RAN domain due to the following observations: (i) there exist some stringent time-critical functions at the RAN domain which will influence the service satisfactory, (ii) a number of compute-intensive RAN operations, such as large matrix inversion and channel decoder, needs dynamic and efficient resource provisioning, and (iii) several novel coordinated/centralized RAN processing can be applied to significantly improve user and network performances like the coordinated multi-point (CoMP) processing. Based on the above observations, this thesis focuses on both cloudification and slicing techniques over the RAN domain.

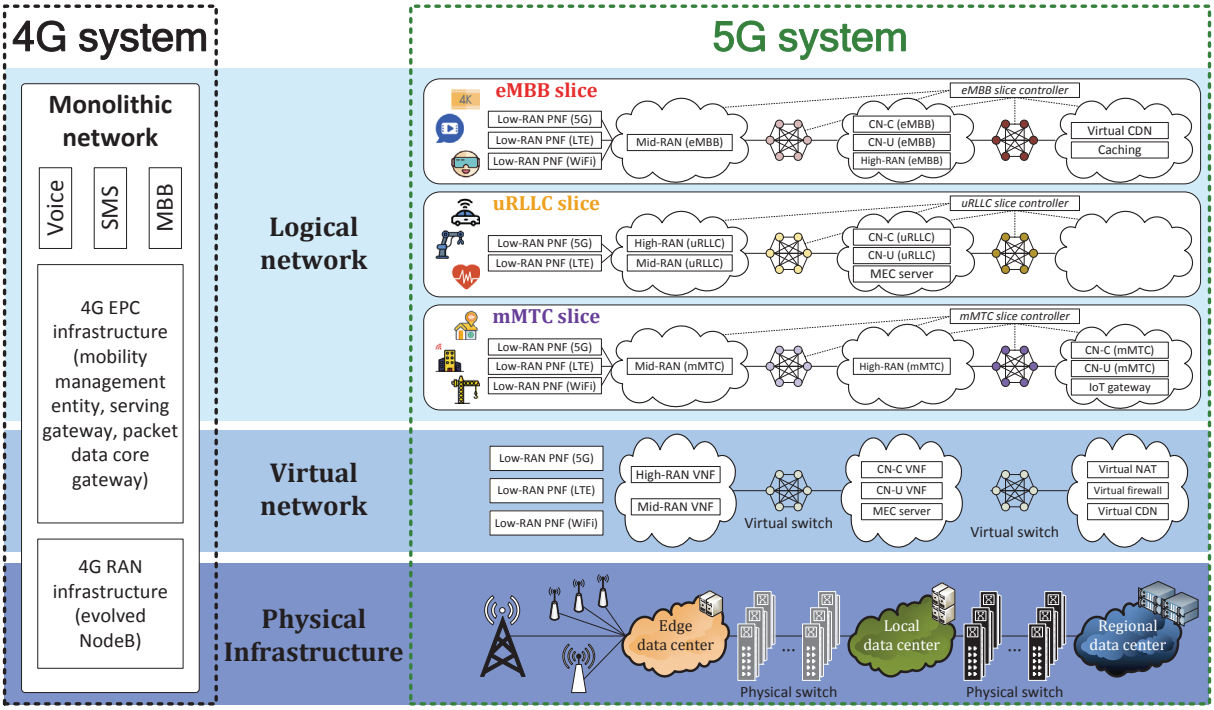


Figure 1.3: Mapping from physical infrastructure, virtual network to logical networks

1.2.1 Challenge 1: Cloudification of radio access network

The idea to cloudify the RAN is promoted by China Mobile write paper [28] as cloud RAN (C-RAN) to replace the monolithic BSs of the legacy distributed RAN (D-RAN) with the centralized processing as the baseband unit (BBU) pool and several distributed antennas equipped by remote radio heads (RRHs). Note that the centralized processing can reduce the required equipments at the cell sites and also enable the cooperation among geographically distributed RRHs to achieve a higher spectral efficiency. Furthermore, via utilizing the cloud infrastructures, several centralized processing at BBU pool can be flexibly aggregated and dynamically allocated to reduce the power consumption and increase the infrastructure utilization ratio. Hence, C-RAN can enable the operators not only to meet the service requirements but also to provide new services with lower cost and simpler upgrade. In summary, C-RAN provide the following advantages: (1) cost reduction for both CAPEX and OPEX, (2) lower energy consumption, (3) higher spectral efficiency, (4) smooth evolution to support future standards, and (5) easy support new revenue generating services.

Despite its aforementioned appealing advantages, three significant challenges emerge at the fronthaul (FH) interface that connects the RRH to the BBU. The first challenge is on the overwhelming capacity requirements due to the raw time-domain samples that shall be transported on the FH interface. For example, 1 Gbps of capacity requirement on the FH link even we only support a mere uplink 75 Mbps radio access rate [29], not to mention the further MIMO layers or carrier aggregation. The second challenge is on the FH latency requirements, e.g., the one-way delay shall be less than 250 μs as suggested by the small cell forum (SCF) in [30] to support coordinated physical layer processing. Last but not least, the FH topology in the original C-RAN

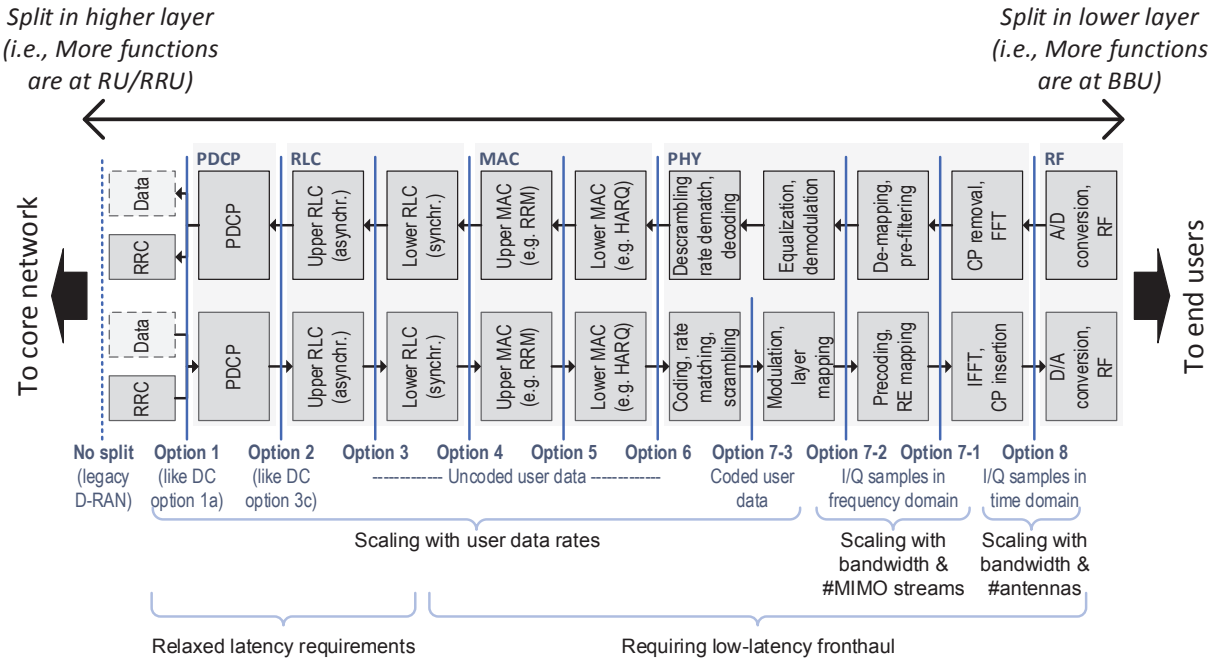


Figure 1.4: Functional split options defined by 3GPP

deployment is not cost-effective since the dedicated connection is needed in a point-to-point manner between each pair of RRH and BBU. In this sense, the functional split concept is introduced in [31], where some baseband or protocol processing can be moved back from the centralized BBU to the distributed RRH according to the applied functional split between them. By doing so, aforementioned challenges can be remedied. More specifically, a number of functional split options is defined by 3GPP in [32], as shown in Figure1.4, aiming to decompose and distribute the overall processing between RRH and BBU. To this end, the RRHs will be evolved from the passive elements to the active ones that can host some processing capability and is termed as (remote) radio unit (RU/RRU), while the BBU can also be further decomposed into a centralized unit (CU) and distributed unit (DU)[3]. Several other split definitions are also introduced by other organizations, e.g., next generation fronthaul interface (NGFI), NGMN alliance, SCF, and common public radio interface (CPRI) forum[4].

Several general surveys of the C-RAN can be found in [33–37]. More specifically, numerous C-RAN studies are conducted along a variety of directions, and thus we can classify these related works into four main categories according to Figure 1.5. Also, we explicitly position our contributions in the first part of this thesis, i.e., from Chapter 2 to Chapter 4, among these four categories in Figure 1.5. More comprehensive technical contents will be given in the corresponding chapters. In the following paragraphs, the related works of these four categories are introduced in much more details.

[3]A thorough summary among the applied C-RAN entity terminologies from different standardization bodies and industry forums will be given in Chapter 2.

[4]Likewise, a comparison of the functional split notations defined by different organizations will be given in Chapter 2.

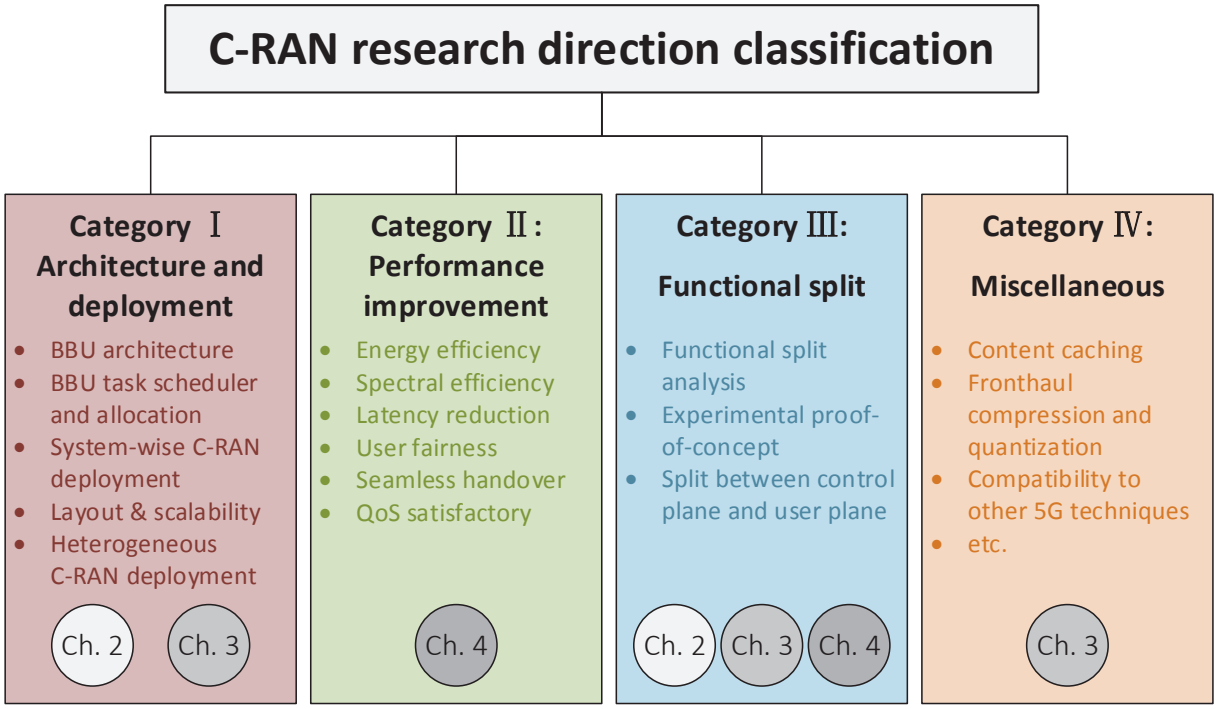


Figure 1.5: Four categories of C-RAN research direction

1.2.1.1 Category I: C-RAN architecture and deployment

The first category of current studies focuses on the C-RAN architecture and deployment. Following works highlight the **BBU architecture design** and the **BBU task scheduler and allocation**. In [38], the authors propose the BBU pool architecture using the general purpose processors (GPPs) for a efficient and scalability deployment. In [39], the implementation of a real-time downlink CoMP architecture for multiple RRHs at the BBU is provided over the GPP platform. The authors of [40] propose a modular architecture for BBU pool to provision the RAN into virtual network operators to enable the multi-tenancy paradigm. The CloudIQ framework provided in [41] can efficiently manage centralized compute resource via (1) partitioning the set of BSs into groups for simultaneous processing and (2) scheduling based on the real-time constraints. A C-RAN prototype it provided in [42] with a scheduling framework called RT-OPEX as a flexible computing task scheduling to dynamically migrates parallelizable tasks to idle computing resources at runtime. The problem of allocating multiple computational resources at BBU is studied in [43] to minimize the number of active BBUs to serve all users. The trade-off between the latency and energy is conducted in [44] for the BBU computational multiplexing of C-RAN in both long- and short-term timescales. A two-level scheduling for both air-interface (i.e., user association and beamforming) and computation (e.g., computing task) is investigated in [45] at the BBU pool. An optimization framework provided in [46] survey the power-performance trade-off to maximize the profit via jointly scheduling resources in FH capacity and BBU pool. The facility location algorithm is provided by [47] to assign both primary and backup functionalities to BBU pools aiming to minimize the number of active BBU pools and provide full coverage to all RRHs.

Moreover, some related works investigate the **C-RAN system-wise deployment issues**, such as the FH impacts and RRH-BBU mapping. A modular C-RAN system is implemented in [48] integrating the optical fronthaul, a software defined front-end and a digital radio compression algorithm. The FluidNet framework in [49] deploys a logically re-configurable FH to apply the appropriate transmission strategies in different parts of network for maximizing the traffic demand satisfaction while optimizing the compute resource usage in the BBU pool. Further, the work of [50] claims that the one-to-one mapping between BBUs to RRHs is sub-optimal and the FH network shall be re-configurable to allow a flexible RRH-BBU mapping. In [51], the C-RAN is investigated with the virtualized passive optical network FH to enable the dynamic RRH-BBU association, and a joint resource allocation of radio, optical network and baseband processing is proposed. An evaluation methodology to analyze the BBU-RRH fast switching scheme is provided in [52] by taking backlogs, idle resources and switching cost into account. The authors of [53] provide an optimization framework to first learn the temporal dependency and spatial correlation among BSs' traffic patterns for the traffic prediction and then cluster complementary RRHs to BBUs to optimize capacity utility and deployment cost.

To go one step further, some academic works study the **layout and scalability issue** and consider several **heterogeneous C-RAN** deployment. The optimization problem of C-RAN infrastructure deployment and layout planning is considered in [54] to minimize the deploying cost. An self-organized C-RAN is considered in [55] to accommodate traffic via scaling the BBUs and RRHs as well as maintaining the balanced BBU-RRH mapping to increase network throughput and to reduce blocked users. In [56], a load prediction model is further considered for proactive network scaling of BBUs and RRHs. Moreover, the heterogeneous C-RAN deployment is suggested by [57] with RRHs at the macro level and standalone BSs with high peak capacity at micro level utilizing the microwave radio as the backhaul to cover some hotspots. A joint resource allocation and admission control framework is provided in [58] considering both macrocell with RRHs of heterogeneous C-RAN. The authors of [59] investigate the soft fractional frequency reuse in a two-tier heterogeneous C-RAN to increase network throughput and conclude that allocating more frequency resources to the RRHs can improve the energy efficiency. A complementary networking architecture for C-RAN is surveyed in [60], aiming to explore the trade-off between the area spectral efficiency, the mean delay, and the system cost. Two following works introduce different types of RRH. The author of [61] investigates the deployment of two different types of RRH to minimize the deployment cost considering the FH capacity and system traffic demands. Another new type of RRH is suggested by [62] using the extra radio connected through the existing RRHs as its FH link and the cooperation strategy between them is also introduced.

1.2.1.2 Category II: C-RAN performance improvement

In the second category, several works study the **energy efficiency** advantages provided by the C-RAN with respective focuses: (1) C-RAN operation power (2) RRH activation power, (3) user-centric power consumption In [63], an energy-efficient C-RAN design aims to select RRH and minimize beamforming power via the group sparse beamforming method. Moreover, the joint optimization over the RRH selection, user association and beamforming vector is done in [64] to minimize the total C-RAN power consumption with imperfect channel state information. The resource allocation scheme that optimizes the C-RAN energy consumption is proposed in [65] considering the allocation of bandwidth and power at RRH as well as the number of active BBUs. A joint C-RAN computation and transmission power minimization problem is studied in [66] to

reach the energy efficiency over different time scales. The authors of [67] exploit the interplay between the transmit beamforming at RRH and the processor sleeping at the data center of BBU pool to minimize the total system power. Moreover, the energy saving approach in implemented in [68] by switching BBUs from sleep mode to operational mode according to the dynamic traffic demand, using the wake-on-LAN packets sent by the RRH or the controller in the BBU pool. To be specific on the RRH activating power, the authors of [69] propose a stochastic game-theoretic algorithm for traffic forecasting and achieving energy efficiency by switching off (and on) RRHs during off-peak hours. The RRH switching problem is further studied in [70] to achieve a trade-off between the system energy saving and the load balancing among RRHs to avoid excessive user handover due to simultaneous switching on/off many RRHs. The authors of [71] consider the sleep mode operation for RRHs and employ RRH clustering techniques to choose active RRHs close to the hotspots. Further, the work of [72] aims to optimize the user-centric quality energy efficiency, considering the quality of experience and the energy consumption. The authors of [73] design the user-centric RRH clustering scheme to maximize the network energy efficiency while satisfying the signal to interference plus noise ratio (SINR) and power constraints.

Moreover, considering the centralization benefits brought by the C-RAN, following works target the **spectral efficiency**. The RRH clustering scheme is investigated in [74] to maximize the network capacity via adaptive selecting the RRH subset and considering both the antenna processing gain and the reference symbol overhead. The authors of [75] design a joint dynamic radio clustering and cooperative beamforming scheme to maximize the weighted sum rate system utility. The user grouping problem is surveyed in [76] to maximize the average achievable sum rate using the joint transmission scheme. A joint problem that associates users to RRHs and BBUs, and allocates power and antennas with the aim of maximizing the system sum rate is conducted in [77]. Further, several works jointly evaluate the power efficiency and spectral efficiency. In [78], the focus is on the optimization of the economical spectral efficiency that takes both spectral efficiency and energy efficiency into account. The authors of [79] consider the user-centric RRH clustering mechanism in C-RAN corresponding to the two respective objectives of improving the area spectral efficiency and energy efficiency.

Further, some works focus on **other C-RAN advantages**, such as latency reduction, user fairness, seamless handover, and quality of service (QoS) satisfactory. In [80], the user utility is formulated using a convex delay cost function, and then a two-step user grouping and resource scheduling approach is designed to maximize the system utility. The idea of adaptively assigning different channel estimation algorithms at BBU to different users is surveyed in [81] to minimize the time required to acquire the channel information and to reduce the E2E latency. An interesting work of [82] studies the impact of the hybrid automatic repeat request (HARQ) report delay and the user throughput due to the FH transportation. In terms of the fairness criteria, a joint RRH beamforming and user association problem is studies in [83] to maximize the minimum user SINR. The author of [84] jointly design the RRH selection and beamforming vectors to maximize the minimum weighted sum rate among users to reach the fairness goal during multicasting. The authors of [85] examine one C-RAN benefit in terms of the synchronous handover without random access procedure. The authors of [86] propose a joint beamforming design scheme among multiple RRHs to maximize user aggregated weighted QoS using a sigmoidal function. To cater to the traffic demand, the work in [87] jointly consider the wireless resource allocation, transmission power minimization, and BBU-RRH assignment.

1.2.1.3 Category III: Functional split

The third category studies following work study the impact of **functional split**. The work of [31, 88] first provides the concept of functional split between RRH and BBU in order to strike a balance between the centralization benefits and strict timing constraints. An overview for a converged fronthaul/backhaul architecture of C-RAN is given in [89] featuring the flexible functional split to not only reduce the FH requirements but also introduce multiplexing gain. The work of [90] analyzes the trade-off between the coordination capability and the FH data rate reduction for some functional splits. The authors of [91] evaluate different functional splits with respect to the energy efficiency and the resource multiplexing gains in terms of FH link and BBU pool. A game-based C-RAN baseband functions splitting and placement problem is surveyed in [92] to balance the trade-off between fronthaul cost and C-RAN cooperation in different deployment scenarios. In [93], the modeling approach and analytical work are provided to minimize the RAN costs by jointly selecting the functional splits and the FH routing paths. Further, several works focus on the experimental approach and the proof-of-concepts. In [94], the authors use the Ethernet as the FH to examine the feasibility of the functional split between medium access control (MAC) and physical layers. The experimental results are provided in [95] to examine the FH performance of two different functional splits when applying different transport protocols. The authors of [96] propose the WizHaul solution to route the NGFI traffic of several functional splits while maximizing the centralization degree of C-RAN through both proof-of-concept and simulations. Likewise, some related works survey the **split between the control plane and user plane** over the C-RAN. The authors of [97] survey the split between control and user planes across RRHs and BBUs on the network sum rate as a function of the FH latency. In [98], the decoupling of control and user planes for HARQ process is considered, in which the retransmission decision is made by RRHs, while the data decoding is done by BBUs.

1.2.1.4 Category IV: Miscellaneous

Apart from the above three categories, several other C-RAN studies are conducted. A joint caching problem in a hierarchical C-RAN is surveyed in [99], where both BBUs and RRHs are considered as possible caching locations. The authors of [100] provide a layered hierarchical caching scheme in C-RAN for the scalable video coding based hypertext transfer protocol (HTTP) adaptive streaming service. Moreover, some works study the techniques of FH compression and quantization. The authors of [101] survey the effective compression algorithms for the FH links and highlight the gains of multi-terminal compression and structured coding. The compressive sensing technique is used in [102] to apply a distributed FH compression scheme at RRHs and the signal recovery at BBU considering the multi-access fading. Some RAN functionalities are surveyed by [103] considering the impacts from both quantize-and-forward and detect-and-forward schemes. Furthermore, some works jointly consider the multi-access edge computing (MEC) concept with C-RAN. In [104], the edge computing capability of MEC is viewed complementary to C-RAN and the Fog RAN (F-RAN) notion is provided to offload parts of UP traffic to the network edge. A hybrid C-RAN architecture with MEC is considered in [105] to assign RRH, BBU and MEC server to each user for a minimized request refusal ratio. Last but not least, the compatibility to other 5G technologies are also studied. The authors of [106] examine the full-duplex communication in C-RAN considering different network- and user-centric clustering schemes. The applicability of full dimensional MIMO within C-RAN is surveyed in [107] via leveraging the low-rank channel structure to reduce the fronthaul overhead.

1.2.2 Challenge 2: Slicing in radio access network

In the second challenge, we analyze the *RAN slicing*, as the natural evolution from the RAN sharing concept introduced since 3G/4G era. Also, as the origination of the RAN slicing, the idea of network slicing aims to consider a collection of logical overlay networks over a physical network [108]. Practically, the actions taken by one slice will not negatively affect other slices [109] even they share the same physical infrastructure. We can notice that the this isolation characteristic provided by the network slicing is an ideal match to the multi-service aspect of 5G presented in Figure 1.1. Hence, several standardization bodies and industry forums highlight the E2E service architecture for multiple services, e.g., ITU-T [110], NGMN alliance [111], 5GPPP [21] and global system for mobile communications association (GSMA) [112]. Moreover, 3GPP mentions the RAN slicing realization principles in [32,113], which can be enabled through the software-defined RAN (SD-RAN) concept that decouples the CP processing from the UP processing. Further, another enabler for RAN slicing highlighted by the open networking foundation (ONF) in [114] is the RAN virtualization and disaggregation, through which the overall RAN service can be composed with high flexibly and scalability to serve multiple innovative services from underlying virtualized and disaggregated RAN entities. To conclude, there are four technology enablers for the RAN slicing, i.e., RAN sharing, E2E service orientation, software-defined RAN, and RAN virtualization and disaggregation, as shown in Figure1.6. In the following, four categories of related works are surveyed according to the above four technology enablers.

1.2.2.1 Category I: RAN sharing

The RAN sharing concept is originated from the idea of *network sharing*, like the gateway core network (GWCN) defined by 3GPP in [115], via sharing the RAN and some parts of CN. Additional network sharing models are surveyed and summarized in [116–118]. To concentrate on the RAN domain, two particular RAN sharing model are widely known as multi-operator RAN (MORAN) and multi-operator CN (MOCN). The MORAN approach shares the same RAN infrastructure but with dedicated frequency bands for different operators, while MOCN allows to also share the spectrum among operators as standardized by 3GPP in [115]. Some detailed investigations on these approaches can be found in [119] and [120] related to the deployment/operation and economics, respectively. Moreover, these approaches can efficiently utilize available radio resources which are surveyed widely as network virtualization substrate (NVS) in [121–123] that can virtualize radio resources for different resource provisioning approaches in order to coexist several mobile virtual network operators (MVNOs) in a single physical RAN. The NetShare approach proposed in [124] extends the aforementioned NVS approach and it applies a central gateway-level component to ensure resource isolation and to optimize resource distribution for each entity. In [125], the authors propose the CellSlice architecture as a gateway-level solution that can indirectly impact individual BS scheduling decision for slice-specific resource virtualization. Authors of [126] provide the AppRAN as the application-oriented framework that defines a serial of abstract applications with distinct QoS guarantees.

1.2.2.2 Category II: E2E service architecture

To serve multiple services, the network slicing approach can be applied as outlined in [127–129]. In [130], the authors summarize the two importance requirements for network slicing from a variety of industry and standardization resources, i.e., “flexibility” and “customization”. The

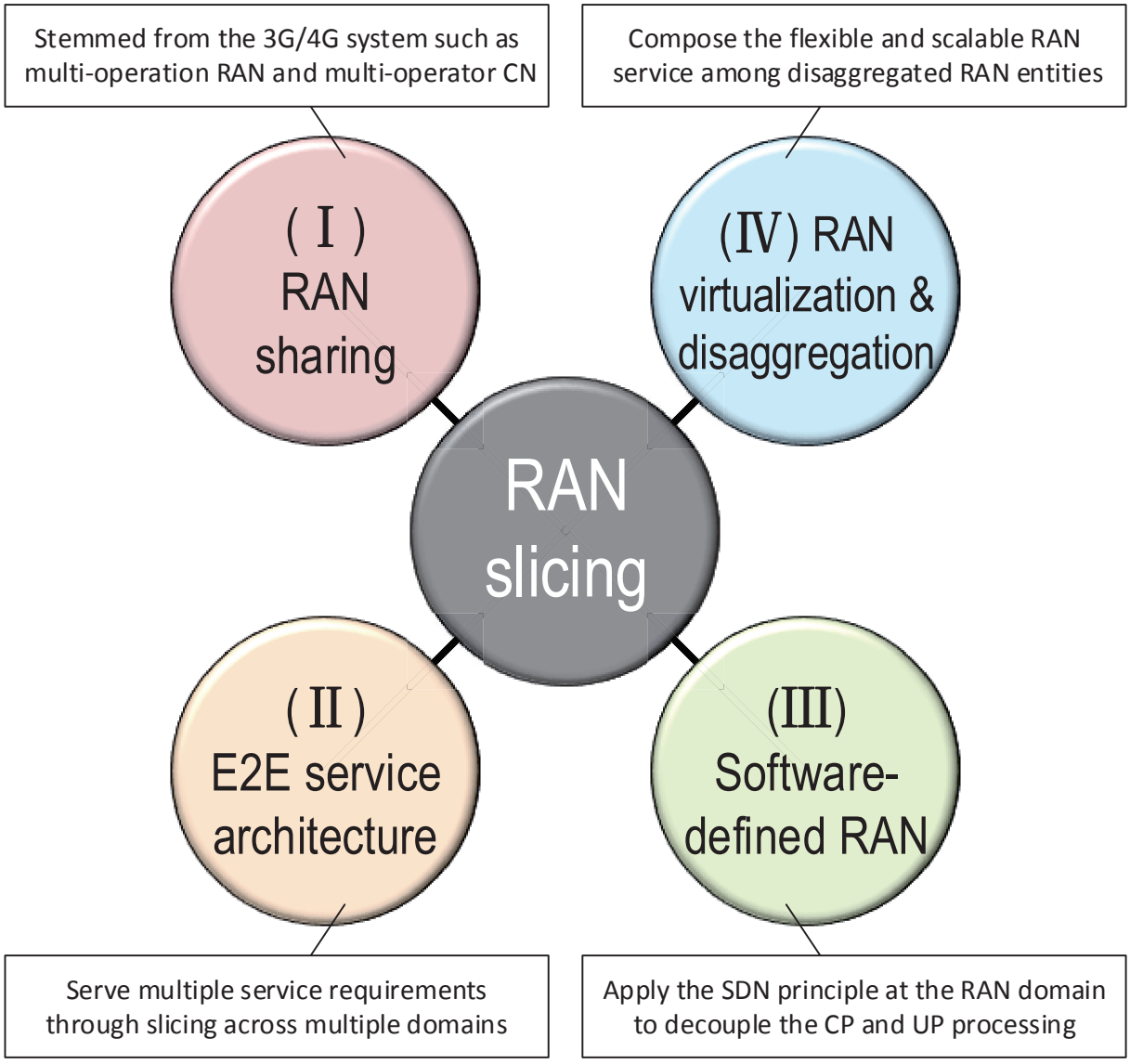


Figure 1.6: Technology enablers for RAN slicing

discussion on the needs of network customization at different granularity levels and a high-level architecture solution are provided in [131]. In [132], the proposed modularized network architecture is composed of several building blocks, each with various sub-functions to customize the functionalities on per slice service. The generic slice as a service model is presented in [133, 134] aiming to orchestrate customized network slice as a service with the mapped network functions based on the service level agreement (SLA). The authors of [135] propose cloud-native network slicing approach to devise network architectures and deployments tailored to the service needs. The network slice broker notion is provided in [136] to enable the on-demand multi-tenant slice resource allocation and is further investigated in [137] with the traffic forecasting scheme to optimize the network utilization and SLAs.

Moreover, several works study some specific services across different domains for an E2E service architecture. In [138], the authors present the E2E network slicing architecture and elaborate on the mobility management and resource allocation mechanisms for three major 5G service types, i.e., eMBB, uRLLC and mMTC. Specifically, the solution of customized slice design for V2X services is provided in [139] via partitioning the RAN and CN resources, as well as the configuration of vehicular end-device functionality. An E2E architecture converging from optical to wireless is outlined in [140] to enable the cross-domain network slicing. Also, a joint RAN and transport network slicing approach facilitating the programmable control and orchestration plane is provided in [141]. The authors of [142] realize the application-driven E2E slicing across wired and wireless networks guaranteeing the E2E bandwidth over isolated VNFs. The aims of [143] is to achieve the E2E reliability of the mission-critical traffic via leveraging multiple radio access technologies (RATs) and dynamic resource orchestration across both RAN and CN over its prototype implementation. The POSENS solution in [144] provides an E2E solution with slice-aware shared RAN and dedicated CN that can achieve isolation and customization capabilities for multiple tenants.

1.2.2.3 Category III: Software-defined RAN

To enable the RAN slicing paradigm, several 5G RAN design requirements elaborated in [8] shall be fulfilled leveraging the high-level design patterns mentioned in [145], e.g., cloud computing, SDN/NFV and software engineering. Also, the RAN slicing realization principles mentioned by 3GPP in [32, 113], such as RAN awareness slicing, QoS support, resource isolation, SLA enforcement among the others, can be enabled through the aforementioned SD-RAN concept. Hence, based on this SD-RAN concept, several related works argue the level of centralization of CP functionalities. The fully centralized architecture is proposed such as OpenRAN in [146], and SoftAir in [147] that may face the challenge of real-time control given the inherent delay between the controller and underlying RAN. While the SoftRAN architecture in [148] statically refactors the control functions into the centralized and distributed ones based on the time criticality and the central view requirement. Also, the hierarchical control approach in the SD-RAN can also be found as CMaaS in [149] and HSDRAN in [150]. Moreover, the SoftMobile approach [151] further abstracts these CP processing across several layers based on their functionalities and can perform the control functionalities through the application programming interfaces (APIs). As for the UP programmability and modularity, the OpenRadio [152] and PRAN [153] are pioneered to decompose the overall processing into several functionalities that can be chained. Last but not least, the proposed FlexRAN in [154] can realize a SD-RAN platform and implements a customized RAN south-bound API through which programmable CL can be enforced with different levels of centralization, either by the controller or RAN agent.

1.2.2.4 Category IV: RAN virtualization and disaggregation

The RAN virtualization is stemmed from the NFV technique to allows multiple virtual BSs to share the common resources for multi-tenancy. The works of [155, 156] provide functional isolation in terms of customized and dedicated CP functionalities for each MVNO. A detailed discussion on the RAN function virtualization can be found in [157] in terms of data-processing complexity, cost analysis, and implementation challenges. In [158], a slice-based "network store" architecture is proposed as a platform to facilitate the dynamic network slicing based on the VNFs on top of the underlying infrastructures. The similar idea is provided in [159] featuring

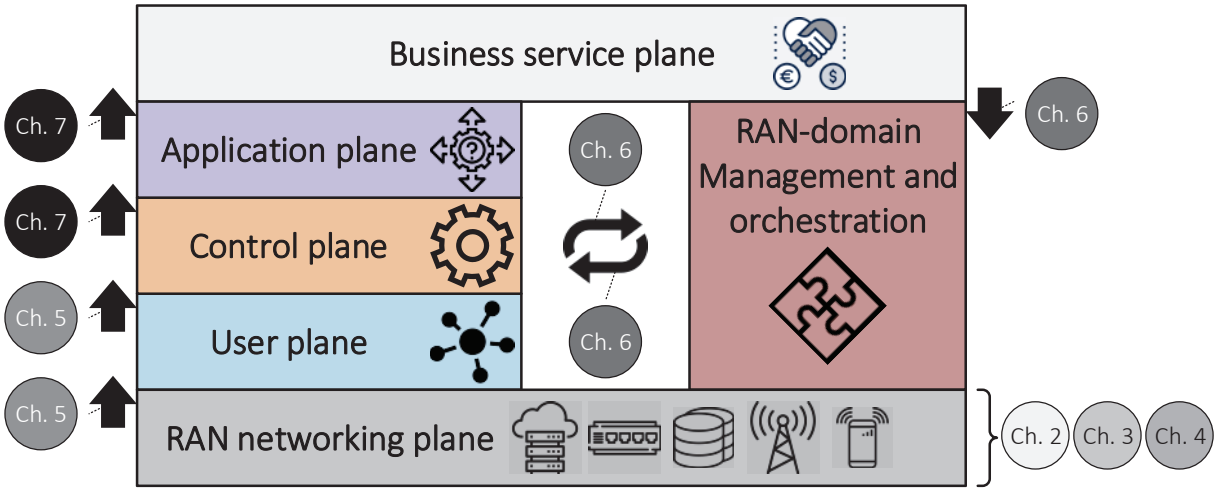


Figure 1.7: Mapping between the RAN architecture and the thesis contributions

the "network and application store", which simplifies the procedure to define each slice. One can notice that the RAN virtualization is beneficial from the C-RAN notion since a large number of RAN functions can be dynamically virtualized at the BBU pool. Some explanations between C-RAN, and RAN virtualization can be found in [160]. The CONCERT architecture proposed in [161] reaps the benefits from both functional splits of C-RAN and RAN virtualization.

To go one step forward, the deployment of virtualized RAN shall consider the disaggregated resource. This RAN disaggregation is surveyed in the mobile central office re-architected as a datacenter (M-CORD) project of ONF to provide the solution of virtualization and slicing out of many disaggregated components. It allows to create a common pool of resources that can be independently selected and combined to compose the initiated RAN service on-demand. As highlighted in [162], the RAN disaggregation can offer the potentials of flexibility, scalability, upgradability and sustainability via adopting the service chaining notion. The authors of [163] allocate the RAN processing functions to disaggregated resources depending on the processing requirements for a better resource utilization and higher energy efficiency.

1.3 Thesis contributions and structure

Based on the aforementioned related works and the technology enablers, we can notice that the cloudification and slicing are critical in the future RAN to enable the 5G vision in terms of flexibility, control and performance. In this sense, the goal of this thesis is to answer these two raised challenges in Section 1.2, and the current manuscript can be divided into two different parts: (i) RAN cloudification (Chapter 2, Chapter 3, and Chapter 4), and (ii) RAN slicing (Chapter 5, Chapter 6, and Chapter 7). A graphical mapping between the RAN architecture (modified from Figure 1.2 for the RAN domain) and the thesis contributions can be found in Figure 1.7. As we can observe, the focus of the first part of the thesis is to investigate several factors for enabling of RAN cloudification. In the second part of the thesis, our aim is to explore the RAN slicing notion across different planes via the aforementioned four technology enablers mentioned in Section 1.2.2.

In the first part, our focus is on the C-RAN, in which the centralized RAN processing in a cloud can support joint multi-cell processing. As mentioned beforehand, this replaces traditional monolithic BSs with distributed (passive) radio elements, called RUs, with much smaller footprints than legacy BS, and centralized (remote) pools of baseband processing units, called DU and CU, where baseband and protocol processing for many BSs takes place. Despite its appealing of coordinated multi-cell processing and multiplexing gain via exploiting the processing load variations, such C-RAN concept still face challenges in terms of severe capacity and latency requirements of the FH interface that connects RU and DU. To better depict these challenges, our investigations is conducted in Chapter 2 among different factors such as functional split between RU and DU/CU, and packet processing (e.g., packetization, scheduling) for the RoE solution. Further, two different implemented functional splits over the OpenAirInterface (OAI) platform [164] are presented and compared in Chapter 3 in order to justify the C-RAN applicability in real life considering the deployment requirements (e.g., resources, performance) and applicable techniques (e.g., radio sample compression). We also present a model and an analytical framework in Chapter 4 for the E2E RAN service delivery problem (i.e., from user toward the centralized processing) in order to clearly understand the impacts from different factors and their correlations.

In the second part, the importance of RAN slicing is highlighted in terms of providing different levels of isolation and sharing to allow a slice owner to customize its service across CP, UP, and CL while increasing the resource utilization of RAN infrastructure. Note that a slice can either be completely isolated from other slices down to the different sets of spectrum and cell site (as in most of current 3G and 4G deployments), or be shared across several types of resources including radio spectrum and network functions (e.g., all network layers of protocol stack), or be customized for a subset of UP and CP processing with an access to a portion of radio resources in a virtualized form. To facilitate these options, we propose a flexible execution environment, denoted as the RAN runtime slicing system, in Chapter 5 to host slice service instances over the underlying RAN modules and resources and to allow flexible service composition and customization across UP, CP, and CL. Moreover, to efficiently provide radio resource and exploit potential multiplexing benefits among different slices, we present a new set of radio resource abstractions in Chapter 5 for the inter-slice resource partitioning and accommodation approaches. Additionally, a number of new interfaces are identified in Chapter 6 for the communications between the RAN runtime and a slice orchestration system to enable the slice life-cycle management for increasing the service acceptance ratio in the disaggregated ultra dense RANs. Finally, the runtime software development kit (SDK) is also provided on top of RAN runtime to simplify the design, development and update of control applications, e.g., spectrum management application described in Chapter 7. These control applications can leverage the two-level abstraction concept presented also in Chapter 7 to be flexibly chained across different domains (e.g., technology and administrative domains) and to compose sophisticated and customized control logics.

Specifically, the chapters of the thesis are organized as following:

Chapter 2 — Impact of Functional Split, Packetization and Scheduling on C-RAN Performance: First of all, this chapter studies the impact of different functional splits on the FH capacity for representative scenarios. We propose the use of a packet-based FH network and study the joint impact of different packetization methods and functional splits on the FH rate and latency. Based on this study, we provide the potential multiplexing benefits in terms of the number of RRUs one could support over an Ethernet-based FH network. Moreover,

we provide the packetization algorithm over the FH links and analyze various packet scheduling policies applied at the aggregated switch. The numerical results show the multiplexing gain considering both packetization and packet scheduling also in the maximum number of RRUs that can be supported. The works related to this chapter are:

- C.-Y. Chang, R. Schiavi, N. Nikaein, T. Spyropoulos, and C. Bonnet, "Impact of packetization and functional split on C-RAN fronthaul performance," in *Proceedings of 2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1-7.

- C.-Y. Chang, N. Nikaein and T. Spyropoulos, "Impact of packetization and scheduling on C-RAN fronthaul performance," in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016, pp. 1-7.

Chapter 3 — Flexible Functional Split Framework over Ethernet Fronthaul in C-RAN: In this chapter, we propose a unified RRU/BBU architectural framework for C-RAN that can support both a flexible functional split and a FH transport protocol over Ethernet. Furthermore, we experimentally evaluate the main KPIs of an operational C-RAN network built based on the OAI platform, under two functional splits and different deployment scenarios. Last but not least, we also investigate the impacts of the FH compression scheme for these two functional splits. The work related to this chapter is:

- C.-Y. Chang, N. Nikaein, R. Knopp, T. Spyropoulos, and S. S. Kumar, "FlexCRAN: A flexible functional split framework over Ethernet fronthaul in Cloud-RAN," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1-7.

Chapter 4 - Flexible C-RAN Centralization for End-to-end RAN Service: In this chapter, we focus on the level of C-RAN centralization to provide the E2E RAN service across centralized BBU pools, FH network, distributed RRUs toward the end-users. Our provided modeling approach and analytical framework aim to maximize the network spectral efficiency by jointly selecting the user association, RRU clustering, functional split, BBU anchoring and FH network routing. Finally, we provide one efficient algorithm and show the numerical results to justify its performance. The work related to this chapter is:

- C.-Y. Chang, N. Nikaein and T. Spyropoulos, "Flexible centralization level of Cloud-RAN," to be submitted to ACM MOBIHOC 2019.

Chapter 5 — RAN Runtime Slicing System for Flexible and Dynamic Service Execution: In this chapter, we propose a RAN runtime slicing system through which the operation and behavior of the underlying RAN could be customized and controlled to meet the slice requirements. Further, a detailed approach for radio resource virtualization is proposed, leveraging different resource abstraction types. Based on it, we formulate the problem of inter-slice resource partitioning and allocation, and propose an efficient algorithm. Finally, we present a proof-of-concept prototype of the proposed RAN runtime slicing system for LTE on top of OAI platform, assess its feasibility and potentials, and demonstrate the isolation, sharing, and customization capabilities among several use cases. The works related to this chapter are:

- C.-Y. Chang and N. Nikaein, "RAN runtime slicing system for flexible and dynamic service execution environment," *IEEE Access*, vol. 6, pp. 34018-34042, 2018.

- C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Radio access network resource slicing for flexible service execution," in *Proceedings of IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Apr. 2018, pp. 1-6.

Chapter 6 — Slice Orchestration for Multi-Service Disaggregated RAN: In this chapter, the devised multi-service RAN runtime is capable of supporting slice orchestration procedures and enabling flexible customization of slices as per tenant needs. The presented architecture concentrates on the orchestration and management systems and a number of new interfaces are identified to enable a customized dedicated orchestration logic for each slice. Finally, we present the results for a disaggregated UDN deployment where the RAN runtime is used to support the slice-based multi-service chain creation and chain placement, with the applied auto-scaling mechanism to increase the acceptance ratio. The work related to this chapter is:

- C.-Y. Chang, N. Nikaein, O. Arouk, K. Katsalis, A. Ksentini, T. Turletti, and K. Samdanis, "Slice orchestration for multi-service disaggregated ultra dense RANs," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 70-77, Aug. 2018.

Chapter 7 — Enabling Control Applications for Multi-Service RAN Programmability: In this chapter, we highlight the runtime SDK that can facilitate the development of agile control applications able to monitor, control, and program the underlying RAN modules. For instance, the implemented SMA is an efficient tool to manage and process different policies and rules defined by various stakeholders, as an open-source and clean-slate replacement for the legacy platform-dependent spectrum management solutions. Also, we can chain single- and cross-domain control applications to form the application plane and to implement sophisticated control logics. A prototype of the proposed runtime SDK is provided based on the OAI and Mosaic5G [165] platforms to demonstrate how slicing and programmability can be achieved in several use-cases. The works related to this chapter are:

- C.-Y. Chang and N. Nikaein, "5G Control Apps: Enabling Multiservice Programmability in a Disaggregated Radio Access Network," *IEEE Vehicular Technology Magazine*, vol. 13, no. 3, Dec. 2018.

- C.-Y. Chang, Ł. Kułacz, R. Schmidt, A. Kliks, and N. Nikaein, "Spectrum management application - A tool for flexible and efficient resource utilization," in *Proceedings of 2018 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018, pp.1-7.

Chapter 2

Impact of Functional Split, Packetization and Scheduling on C-RAN Performance

2.1 Introduction

The C-RAN architecture is one of the most promising technologies that will impact future 5G architecture and possibly re-shape existing mobile network environments. Unlike traditional D-RAN deployment, C-RAN can detach the baseband processing from the remote radio equipments. The baseband processing for many BSs, now called RRHs, is centralized into multiple pools of shared and dynamically allocated BBUs, offering energy and multiplexing gains. These BBU functions could be implemented on commodity hardware and be executed on virtual machines, further benefiting from the virtualization technology. Finally, the centralization of BBU functions facilitates advanced coordinated multi-cell signal processing, which is less practical in D-RAN setup due to the stringent synchronization constraints. Despite its appeal, one key obstacle in the adoption of the C-RAN architecture is the excessive capacity and latency requirements on the FH link connecting the RRH with the BBU. An example depicted in Figure 2.1 shows that shifting all baseband processing away from the BS to a remote cloud implies to support a mere 75 Mbps radio access rate, we need to transport approximately 1 Gbps of information on the FH link. If one further considers extra MIMO layers or carrier aggregation, these rates quickly become prohibitive. Furthermore, the user expects to receive the (negative) acknowledgement (ACK/NACK) response within several milliseconds (ms) to comply with the HARQ protocol [166] after its transmission, imposing also a strong latency requirement on the FH link.

To relax these excessive FH bandwidth constraints, the concept of C-RAN is revisited, and a more flexible distribution of baseband functionality between the RRHs and the BBU pool is considered [31, 88]. Rather than offloading all the BBU processing on the cloud, dividing the receiver (Rx) and transmitter (Tx) chains in different blocks, it is possible to keep a subset of these blocks in the RRH. This concept is also known as *flexible centralization*. By gradually placing more and more BBU processing at the RRH, the FH capacity requirement will be gradually reduced (e.g., redundant cyclic prefix and guard band removal). Nevertheless, flexible centralization may limit the initially envisioned benefits of C-RAN:

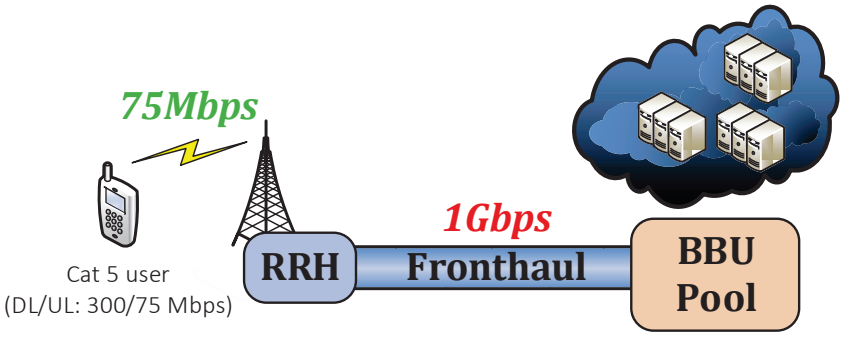


Figure 2.1: Fronthaul data rate explosion in the uplink direction

(i) RRHs become active elements performing some computing tasks, termed as RRUs, and thus more expensive,

(ii) De-centralizing the BBU processing reduces the opportunities for multiplexing gains, coordinated signal processing and advanced interference avoidance schemes.

Consequently, flexible or partial centralization provides a trade-off between what is gained in terms FH requirements and what is lost in terms of C-RAN features.

Another key question is how the information between the RRU and the BBU is transported over the FH link. A number of FH transmission protocols are already investigated, such as the CPRI [167], open radio interface (ORI) [168], and open base station architecture initiative (OBSAI) [169]. However, these have mainly been considered for carrying raw in-phase and quadrature (I/Q) samples in a traditional C-RAN architecture. In light of the different possible functional splits, different types of information are transported over the FH link, carried out by eCPRI in [170]. Given the extensive adoption of Ethernet in clouds, data centers, and the core network, RoE [171] could be a generic, cost-effective, off-the-shelf alternative for FH transport. Also, both NGFI [172] and xRAN forum [173] highlight the potentials of RoE as the FH solution. Furthermore, while a single FH link per RRH, all the way to the BBU pool, has usually been assumed, it is expected that the FH network will evolve to a more complex multihop mesh topology, requiring switching and aggregation [172]. This is further facilitated by a standard Ethernet approach and SDN-based switching capabilities.

Nevertheless, packetization over the FH interface will introduce additional concerns related to the latency and overhead. In practice, when several incoming samples arrive, they shall be packed into one or more Ethernet frames with the extra (sub-)headers for the FH transportation To ensure this overhead is small and does not waste the capacity gains obtained from functional splitting, it would thus be desirable to fill all the Ethernet payload before sending. However, waiting to fill a payload may introduce additional latency, possibly using up some of the latency budget. Hence, it is important to consider the impact of packetization on the FH performance, in conjunction with possible functional splits, to understand the feasibility and potential gains of different approaches.

Moreover, when transporting Ethernet packets between RRU and BBU, several switches within the FH network are utilized as the aggregated and multiplexing points where the packets are switched and routed. The packet scheduling policy at these switches will impact the latency and fairness for all connected RRUs. In view of the fairness, all RRUs expect to have the same

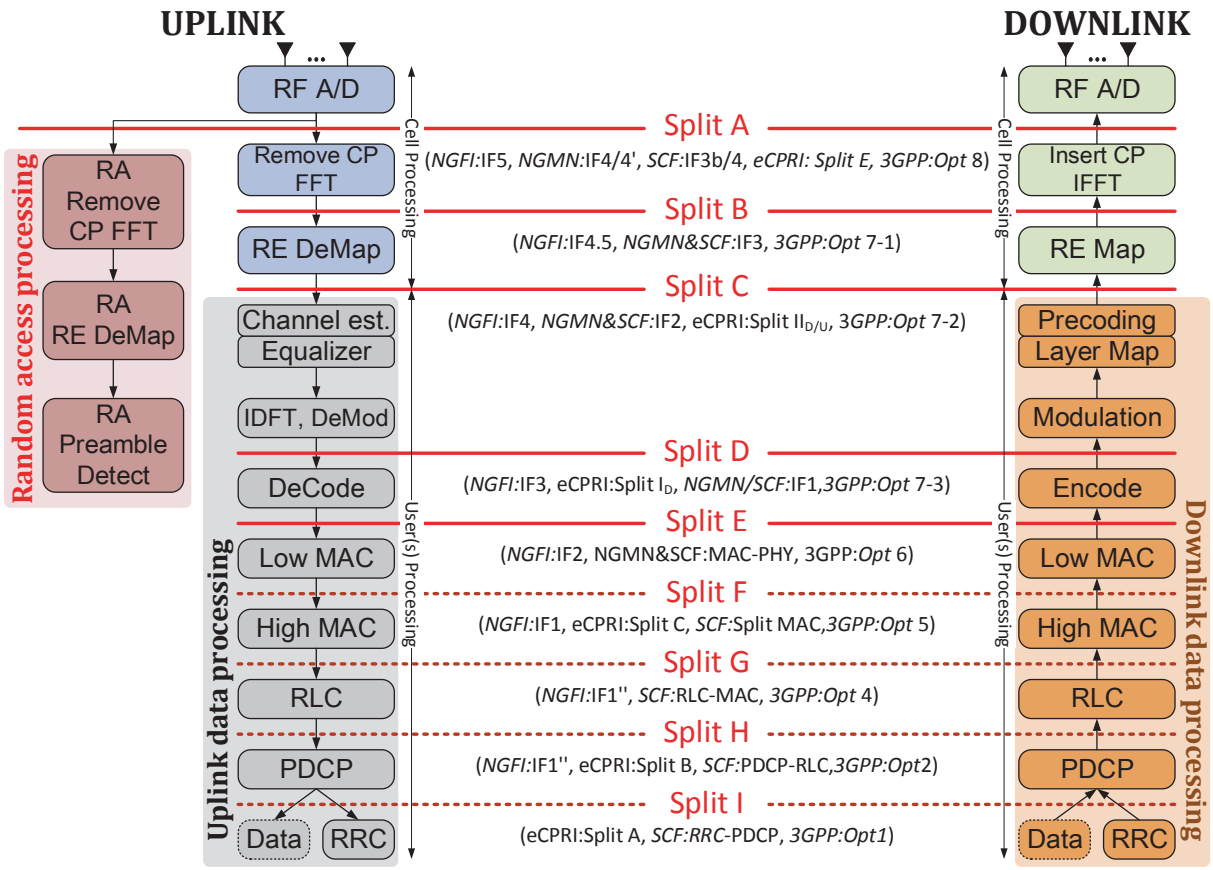


Figure 2.2: Functional splits of the uplink and downlink directions

level of latency irrelevant to their traffic characteristics. However, ensuring a unified latency implies the inflexibility on the overprovision of switching performance without considering the multiplexing gain brought by the C-RAN To this end, the packet scheduling is surveyed to better apply a trade-off between the fairness and the multiplexing gain.

2.2 System model

In this section, we thoroughly introduce the considered C-RAN system model in terms of the functional split, the network topology of C-RAN and the considered HARQ timing constraint of LTE system. These three factors will be used throughout in this chapter.

2.2.1 Functional split

Based on the aforementioned flexible centralization concept, several *functional splits* are defined to distribute the baseband processing between RRU and BBU, as the Split A to Split I shown in Figure 2.2 for both uplink (UL) and downlink (DL) directions. These identified splits can be mapped to the interfaces introduced by NGFI [172, 174], SCF [30], NGMN alliance [175], CPRI forum [170], and 3GPP [32]. In the following, we briefly introduce these functional splits:

- *Split A*: This corresponds to the initial C-RAN vision, in which the RRU only includes antennas with radio frequency (RF) frontend, while the BBU includes all other functionalities. Therefore, the time-domain I/Q samples are transported through the FH interface.

- *Split B*: Additional (inverse) discrete Fourier transform (DFT/IDFT) and cyclic prefix removal/addition operations are done by the RRU, and thus frequency-domain I/Q samples of all physical resource blocks (PRBs) are transported.

- *Split C*: The PRB (de-)mapping is further pushed to the edge RRU and only the allocated PRBs are transported in the FH, i.e., non-allocated PRBs will not be transported.

- *Split D*: Almost all layer 1 processing is distributed at RRUs, while keeping the bit-rate processing (i.e., channel decoder/encoder) at BBU. Hence, the coded bits in the DL direction and the log-likelihood ratio (LLR) of each bit in the UL direction are carried in the FH interface.

- *Split E*: All L1 processing is performed by RRUs and the MAC protocol data units (PDUs) are transported in a per transmission time interval (TTI) basis. Note that above five functional splits must respect the stringent HARQ timing constraint.

- *Split F*: Some low-level MAC processing (i.e., functions with strict timing requirement like the HARQ process, or latency-sensitive function like random access process) is further distributed at RRUs, while the centralized high-level MAC performs the interference coordination and joint scheduling. Such split can remedy the FH latency requirement but complicates the scheduling operation.

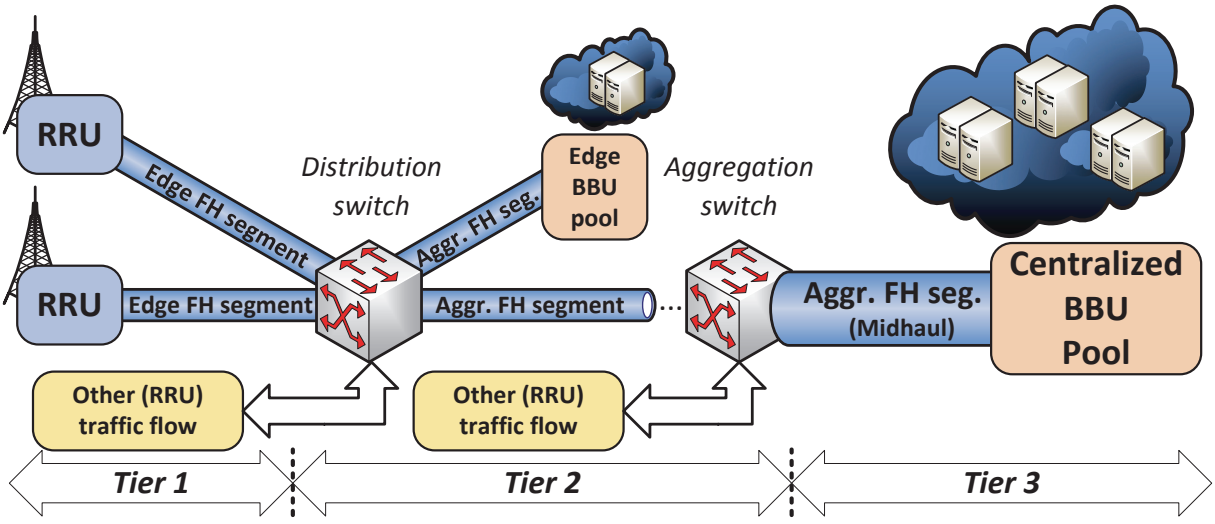
- *Split G*: All MAC functionalities are performed distributively; however, the same latency requirement as split F remains, due to the close relation between the radio link control (RLC) and MAC sub-layer in particular in the DL direction. What is worse, some centralization benefits like joint scheduling are disabled[1].

- *Split H*: Only packet data convergence protocol (PDCP) and upper layers processing are centralized. This split allows traffic inter-working between different RATs and can further lower the FH latency requirement.

- *Split I*: Such split has the same latency requirement as split H but with distributed PDCPs, i.e., no multi-RAT inter-working. However, the MEC entity and its associated MEC services can be placed more closer to end user in order to exploit the latency reduction benefits. Note that both Splits H and I have been surveyed in the 4G era during the introduction of dual connectivity technique in Release 12.

Specifically, we concentrate on those layer 1 split (i.e., from Split A to Split E) in this thesis, since their latency constraint (i.e., HARQ timing constraint) is more stringent and challenging than the ones of splits above (i.e., from Split F to Split I). Without loss of generality, the UL direction is more focused, due to the DL processing is less compute-intensive and it can be prepared even before the UL processing is finished.

[1] An inter-RLC split can circumvent this condition, i.e., 3GPP split option 3.



Functionalities		Tier 1	Tier 2	Tier 3
3GPP	TS 38.401	gNB-DU		gNB-CU
	TR 36.756	eNB-DU		eNB-CU
	TR 38.816	lls-DU	lls-CU	
NGMN alliance		RU	DU	CU
CPRI forum	CPRI	Radio equipment (RE)	Readio equipment control (REC)	
	eCPRI	eCPRI RE (eRE)	eCPRI REC (eREC)	
NGFI		RRU	Radio aggretaion unit (RAU)	Radio cloud centre (RCC)
IEEE 1914		RU	DU	CU
xRAN forum		RU	lls-CU	
TIP		RRU	vBBU	

Figure 2.3: Considered network topology of C-RAN

2.2.2 Network topology of C-RAN

Another key question is how the information between RRU and BBU is transported over the FH link. Based on the aforementioned benefits claimed in Section 2.1, we hereby leverage the RoE approach and expect the FH network will evolve to a more complex multi-hop mesh network topology. This evolution can provide substantial improvements over scalability, capital expenditure, and multiplexing compared with the point-to-point FH link, and thus the FH network will become similar to the backhaul network [176]. An example of a multi-segment FH network topology is shown in Figure 2.3. Here the distribution and aggregation switches can be utilized to transport not only the C-RAN traffic, but also other traffic flows, which is inline with the concept that the C-RAN could reuse already deployed Ethernet networks. Further, the considered network topology of C-RAN in Figure 2.3 can be mapped to the three-tier architecture proposed by several organizations, such as 3GPP [177–179], NGMN alliance, CPRI forum, NGFI, IEEE 1914, xRAN forum [173], telecom infra project (TIA) [180], and SCF, in which there are two functional splits in between: low level split (LLS) and high level split (HLS). To conclude, we interchangeably use the term “RRU” and “RU” for distributed radio elements, and “BBU”, “DU” and “CU” for pools of baseband units through the thesis.

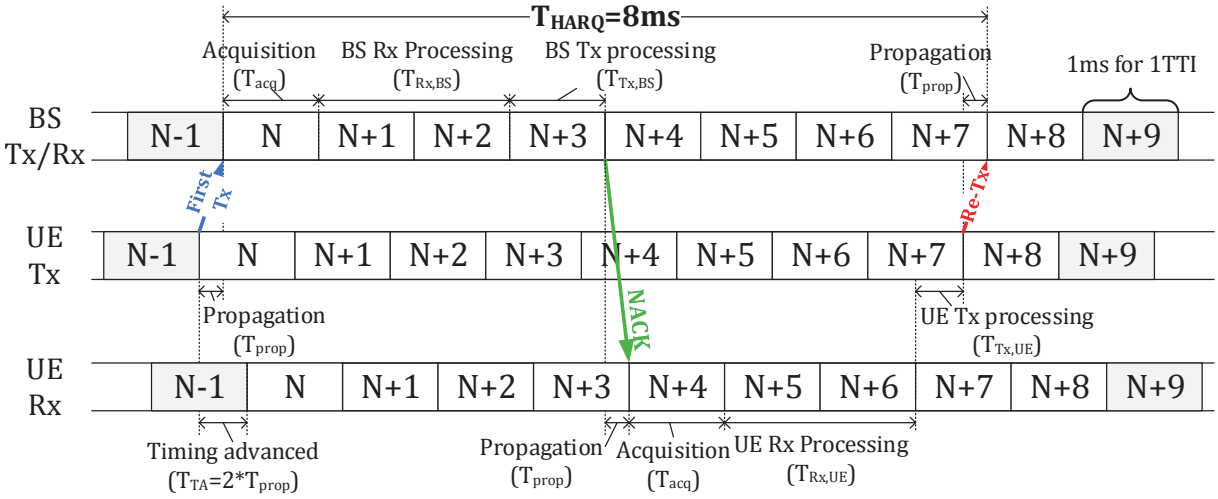


Figure 2.4: Uplink HARQ timing of LTT FDD mode

2.2.3 HARQ timing constraint

The HARQ timing constraint is the one that sets a stringent limitations on the C-RAN deployment and FH transportation; practically, it is to require every received MAC PDUs should be acknowledged or non-acknowledged within a specific time duration. Take the frequency division duplexing (FDD) mode in LTE as an example, the HARQ round trip time (T_{HARQ}) is 8 ms and the uplink HARQ timing is displayed in Figure 2.4. Each MAC PDU sent by the user equipment (UE) in he UL direction at the N^{th} TTI is propagated (T_{prop}), acquired (T_{acq}) and processed through the BS reception processing ($T_{Rx,BS}$). Then, according to the outcome from the reception processing, the BS transmission processing ($T_{Tx,BS}$) will generate the ACK or NACK response to the UE. This ACK or NACK response will be received at the $(N+4)^{th}$ TTI by the UE and the re-transmission will start at the $(N+8)^{th}$ TTI if needed.

Based on Figure 2.4, the maximum processing time for BS reception and BS transmission is written in Eq. (2.1a). Not that the acquisition time (T_{Acq}) aims to acquire all samples within one TTI and is 1 ms[2]. Even though the BS transmission processing can not start until the BS reception processing task for the ACK or NACK response is done; however, most of its processing can be prepared beforehand, e.g., physical layer processing except the physical HARQ indicator channel. In that sense, the BS transmission processing time ($T_{Tx,BS}$) can be treated as independent from the BS reception processing time and it can be bounded irrelevant to the ACK or NACK response. Specifically, We can apply the assumption used in [166] that sets the maximum BS transmission processing time as 1 ms and write the constraint in Eq. (2.1b) for reception processing only. Last but not least, this 2 ms budget will be spent not only on the BS reception processing (i.e., T_{proc}) but also on the FH transportation (i.e., T_{FH}).

$$T_{Rx,BS} + T_{Tx,BS} \leq \frac{T_{HARQ}}{2} - T_{Acq} = 3\text{ms} \tag{2.1a}$$

$$T_{Rx,BS} = T_{proc} + T_{FH} \leq 3\text{ms} - \max\left(T_{Tx,BS}\right) = 2\text{ms} \tag{2.1b}$$

[2]This acquisition time depends on the RF frontend capability and its worst case is 1 ms.

Table 2.1: Configuration parameters per scenario

Scenario	**1**	**2**	**3**
Radio bandwidth	20 MHz		
Oversampling Ratio	1		
Rx Antennas	4		
Cyclic prefix length	Normal		
MIMO layer	4 layer		
Control channel overhead	4.3%		
Random access overhead	0.3%		
Modulation	64 QAM	16 QAM	QPSK
TBS index	26	16	9
Time domain sample bit width	16		
Frequency domain sample bit width	16		
LLR bit width	8		

2.3 Peak-rate analysis

Based on the system model, we first perform the peak-rate analysis via assuming the maximum UL/DL data rate are transported by all BSs simultaneously. This analysis shows the impact of different functional splits on the FH data rate and its relation to some important parameters, such as the number of MIMO layers, modulation order, and transport block size (TBS). Specifically, we focus on three LTE-advanced (LTE-A) cell configurations as listed in Table 2.1 without considering any packetization impacts. For simplicity, the number of sectors in a cell is not considered in this analysis, but it could be introduced simply as a constant factor.

Moving on from the general multi-segment example in Figure 2.3, a specific two-segment Ethernet-based FH network is considered in this analysis, assuming 4 Gbps and 20 Gbps FH capacity for the edge and aggregated FH segments, respectively. Afterwards, the FH data rate is computed among the three considered scenarios and we derive the number of supported RRUs within this network as the performance metrics, as shown in Table 2.2. It can be observed that the FH data rate, when moving from Split A to B, is almost halved, due to the guard band and cyclic prefix removal. Continuing, Split C offers smaller additional gains, when compared with Split B. This is expected, as the rate for Split C depends on the number of allocated PRBs, and almost all PRBs are used since the all cells are transported in their peak rates. For Split D, the data rate depends on the applied modulation order. Specifically, when the modulation order is high, e.g., 64QAM, Split D will have a negative impact compared to Split C due to more bits being required to represent each data sample. As for Split E, the data rate is determined by the overall TBSs, and it exhibits more than 90% reduction in the FH data rate when compared with Split A. Nevertheless, this improvement comes with the cost of performing all layer 1 processing at the RRHs separately without any possibility of cooperative processing. Note that such peak-rate analysis represents a conservative estimate of potential multiplexing gains and is served as the baseline for later comparisons.

Table 2.2: Required FH data rate and the number of supported RRUs

Scenario	1	2	3
Split A	3.93 Gbps, 5 RRUs		
Split B	2.15 Gbps, 9 RRUs		
Split C	2.14 Gbps, 9 RRUs		
Split D	2.63 Gbps, 7 RRUs	1.76 Gbps, 11 RRUs	878.3 Mbps, 22 RRUs
Split E	300.8 Mbps, 66 RRUs	123.9, Mbps, 161 RRUs	63.7 Mbps, 313 RRUs

2.4 Impact of functional split and packetization

2.4.1 Packetization process overview

First of all, the packetization is defined as a process of bundling data into packets according to a predefined protocol. In a typical UL transmission, RRU needs to packetize the received data samples before being transported over the FH link to the BBU, while the BBU needs to de-packetize so as to fetch all required samples before processing. Practically, three characteristics are important when designing the packetization process: latency, overhead and complexity. Note that the first two characteristics are more critical, due to the aforementioned capacity and latency requirements on the FH link. Ideally, the packetization process should minimize latency and overhead simultaneously; however, reducing the overhead per frame will require extra waiting time to fill up the payload. In this sense, a trade-off can be found between packetization latency and overhead.

Moreover, based on the HARQ timing constraint pointed out in Eq. (2.1b), we can see that the overall BS reception processing shall be less than 2 ms. Such delay budget shall be shared between the processing time (T_{proc}) and the overall delay introduced by FH transportation (T_{FH}). Note that such T_{proc} is the sum of the processing time at both RRU and BBU, and it is a function of the PRB number, modulation coding scheme (MCS) index and the virtualized BS execution platform. For simplicity, we can set an upper bound to this value via (1) assuming both RRU and BBU use the same virtualization platform that takes the longest processing time (i.e., DOCKER as measured in [166]) and (2) applying the maximum number of PRBs (100 PRBs for 20 MHz bandwidth) and the highest MCS index. In that sense, we can write the actual constraint for the FH transportation as in Eq.(2.2) with following decomposed elements:

1. Packetization delay T_{pkt} is the time required to packetize samples into a packet,
2. RRU queuing delay T_{Q_r} is the interval from the packet is packetized until it is transported over the edge FH segment,
3. FH propagation delay T_{prop} is the time to transport a packet over all FH segments, i.e., edge FH and aggregated FH segments,
4. Switch queuing delay T_{Q_s} is the interval from the packet arrival time at the switch until it is transported over the aggregated FH segment. Note that the switch can apply different packet scheduling policies and/or discard packets that are past the deadline.

$$T_{FH} = T_{pkt} + T_{Q_r} + T_{prop} + T_{Q_s} \leq 2\text{ms} - \max\left(T_{proc}\left(\text{PRB}, \text{MCS}, \text{Platform}\right)\right) \tag{2.2}$$

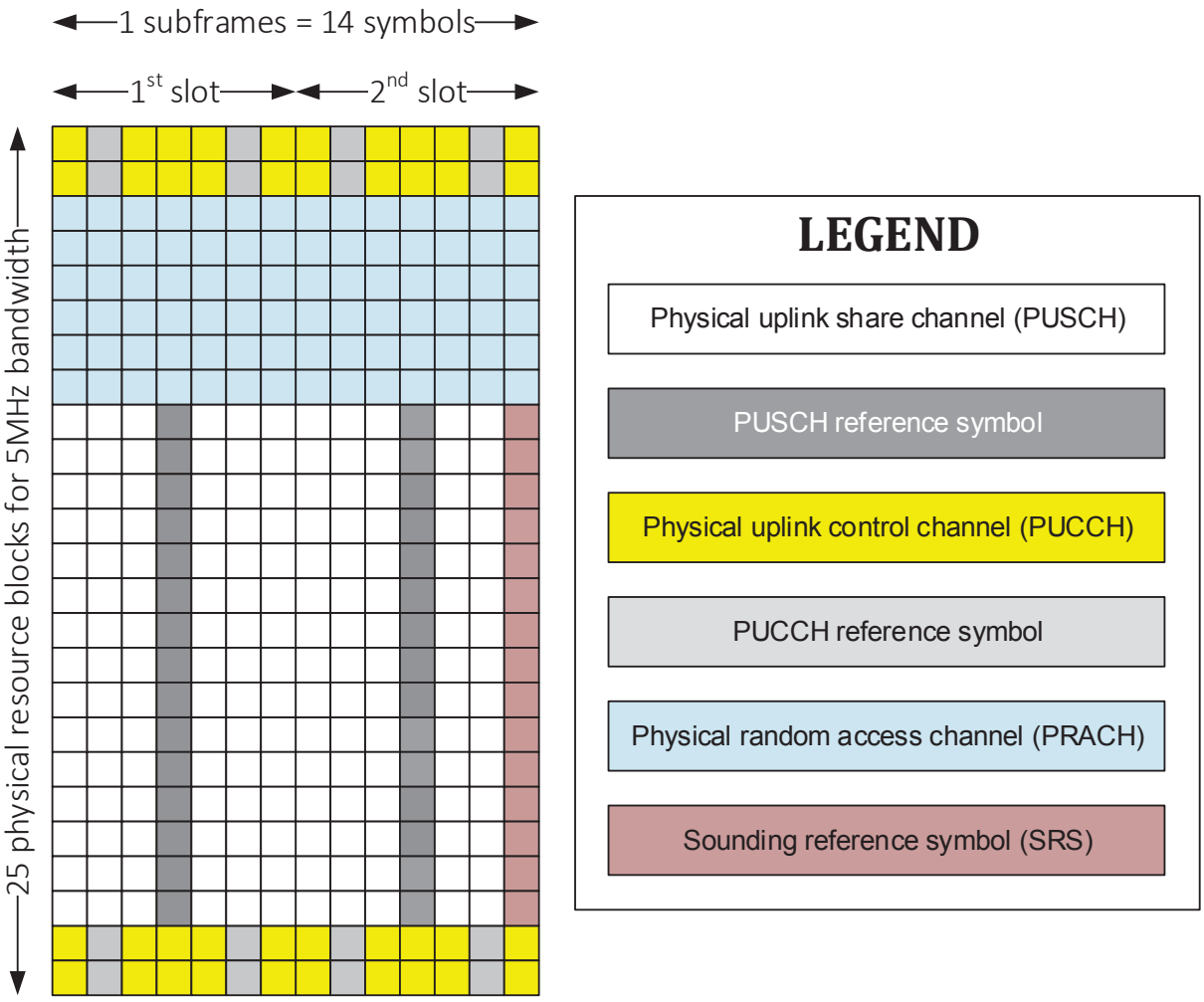


Figure 2.5: LTE uplink frame resource grid example

Last but not least, this constraints is applicable only to the *time-critical symbols* of each functional splits. In specific, the time-critical symbols are the ones that will used by the BBU to initialize the reception processing, e.g., every orthogonal frequency-division multiplexing (OFDM) symbol for Splits A and B, the last reference signal (RS) and data symbols used for the demodulation processing for split C, the last demodulated and decoded symbol in each TTI for Splits D and E, respectively. Take the LTE uplink resource grid of 5 MHz bandwidth in Figure 2.5 as an example, there are 14 OFDM symbols within a subframe and 25 physical resource block in the frequency domain. The aforementioned last RS and data symbols for the physical uplink shared channel (PUSCH) is at the 11^{th} symbol and the 14^{th} symbol, respectively. Note that here our focus is not on the control and random access channels, i.e., physical uplink control channel (PUCCH) and physical random access channel (PRACH), since the bottleneck for packetization process is the PUSCH. Further, the aforementioned last demodulated and decoded symbols is at the 14^{th} symbol.

2.4.2 Packetization of several functional splits

Based on the constraint formulated in Eq. (2.2), we further discuss the impacts of packetization in following three directions: (1) latency-overhead trade-off, (2) burst traffic offloading, and (3) maximum payload size.

2.4.2.1 Latency-overhead trade-off

In the first direction, we can see that the decision to packetize samples immediately or with more samples from upcoming symbols shall be made for those aforementioned time-critical symbols. On one hand, it takes $T_{sym} \approx 71.354\mu s$ to wait for the next symbol; on the other hand, $O = 78$ bytes overhead will be generated for each extra Ethernet packet. Considering this trade-off, we provide the *zero cross-symbol* criteria in Eq. (2.3) to make this decision, i.e., the incoming samples will be packetized immediately if this criteria is fulfilled. Note that $r = 4$ Gbps and $R = 20$ Gbps are the FH capacity for the respective edge FH and aggregated FH segments, N is the number of RRUs to be supported, $E[x_{i,j}]$ is the expected number of samples in bytes within the j-th OFDM symbol of the i-th RRU, and P is the packet payload size in bytes.

$$T_{sym} \cdot \min\left(r, \frac{R}{N}\right) - 8 \cdot \left(E[x_{i,j}] + O\left\lceil \frac{E[x_{i,j}]}{P} \right\rceil\right) > 0 \tag{2.3}$$

Based on the parameters utilized in the Section 2.3 and a decent number of RRUs (i.e., less than the maximum number derived in Section 2.6), this condition can be fulfilled for all functional splits. Thus, all samples of time-critical symbols are packetized immediately, i.e., $T_{pkt} = 0$. In contrast, the samples of not-time-critical symbols do not need to be packetized immediately since the constraints of Eq (2.2) is not applicable to them.

2.4.2.2 Burst traffic offloading

For Splits A, B, and C, the sample output time for packetization process is periodic in per OFDM symbol basis. However, for Splits D and E, the baseband processing done by the RRU will across several OFDM symbols, i.e., a large number of samples will be outputted in a short period. Take the channel estimation and demodulation processing of Split D as an example, we can observe in Figure 2.6 that this processing is done across 14 OFDM symbols. In the beginning, the RS symbols (i.e., the 4^{th} and the 11^{th} OFDM symbol) are utilized for channel estimation. Then, the interpolation is done to get the channel estimation results for the remaining data symbols. Within this step, a non-causal operation is observed, i.e., using future RS symbols (i.e., the 11^{th} OFDM symbol) to interpolate for the previous channel estimation results (i.e., from the 7^{th} to the 10^{th} OFDM symbol). In this sense, these data symbols can only be outputted until the future RS symbol is received. To be more specific, this non-causal operation can be characterized as the *look-ahead depth* (LAD) parameter, which equals to the longest waiting time, i.e., it is 4 in the example of Figure 2.6. After the interpolation, the data symbols will be equalized, demodulated and outputted for packetization according to the pre-defined operating period. A list of possible LADs of both normal and extended cyclic prefix lengths is presented for two operating periods (i.e., slot or subframe) in Table 2.3.

According to the aforementioned introduction on the channel estimation and demodulation processing of Split D, we can see one significant characteristic is its bursty traffic at the end of the operating period, either subframe or slot, and it imposes a heavy burden on the FH link

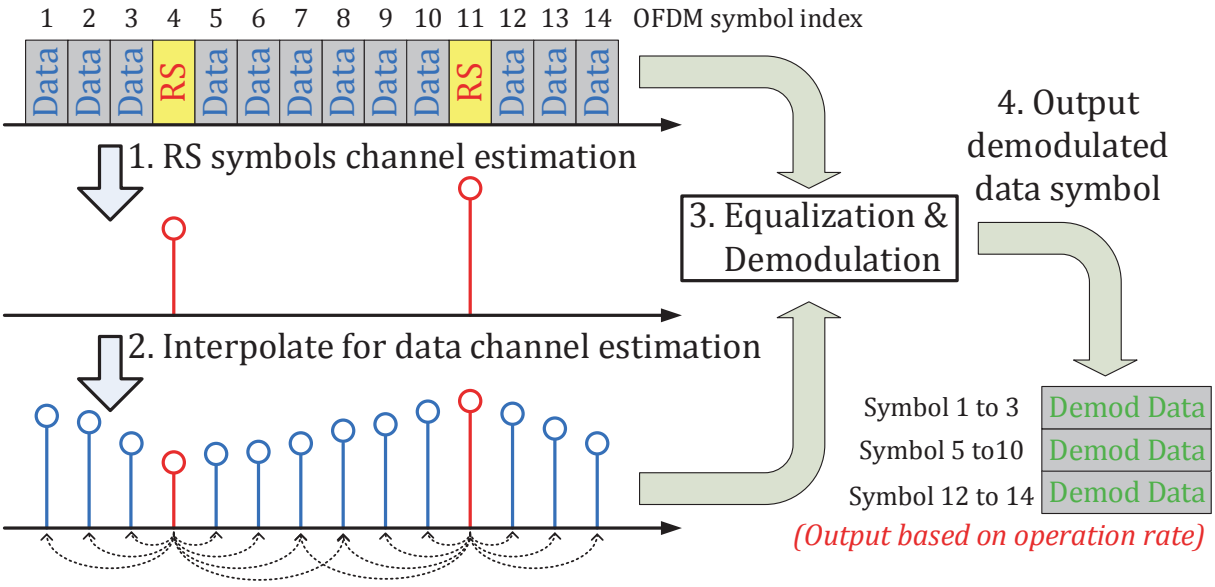


Figure 2.6: Channel estimation and demodulation processing

Table 2.3: Possible look-ahead depth values

Operating period	Subframe	Slot
Normal cyclic prefix length	4, 5, 6, 8, 9, 10	3
Extended cyclic prefix length	3, 4, 5, 7, 8	2

since a number of packets will be flooded and the traffic is congested only in a short period. Hence, we devise a *pre-fetch* scheme aiming to equalize, demodulate and packetize samples once all required RS symbols within the LAD are available for interpolation (cf. Step 3a in Figure 2.7) rather than following the pre-defined operating period (Step 3b in Figure 2.7). In other words, this pre-fetch scheme aims to packetize as soon as possible to avoid the bursty traffic and FH congestion in a very short period, at the cost of the increasing overhead. Practically, the pre-fetch scheme is critical to mitigate the bursty traffic at the end of the operating period, i.e., OFDM symbol 1 to 3, 5 to 10, and 12 to 14 are all outputted at the end of the operating period as shown in the Step 3b of Figure 2.7. Furthermore, the pre-fetch scheme can also be applied to the control channel (i.e., PUCCH in Figure 2.5); however, little impact is observed since the control channel has much fewer samples than the data channel.

Nevertheless, such pre-fetch scheme is not applied to Split E due to the following two reasons. First, even Split E has the same bursty traffic characteristic as Split D; the FH capacity requirements is less stringent for Split E (cf. Table 2.2) and thus the pre-fetch scheme shows few improvements. Second, the early stop mechanism at the channel decoder of Split E may drastically impact the user performance and cause unnecessary retransmissions when the coding rate is high, i.e., close to 1.

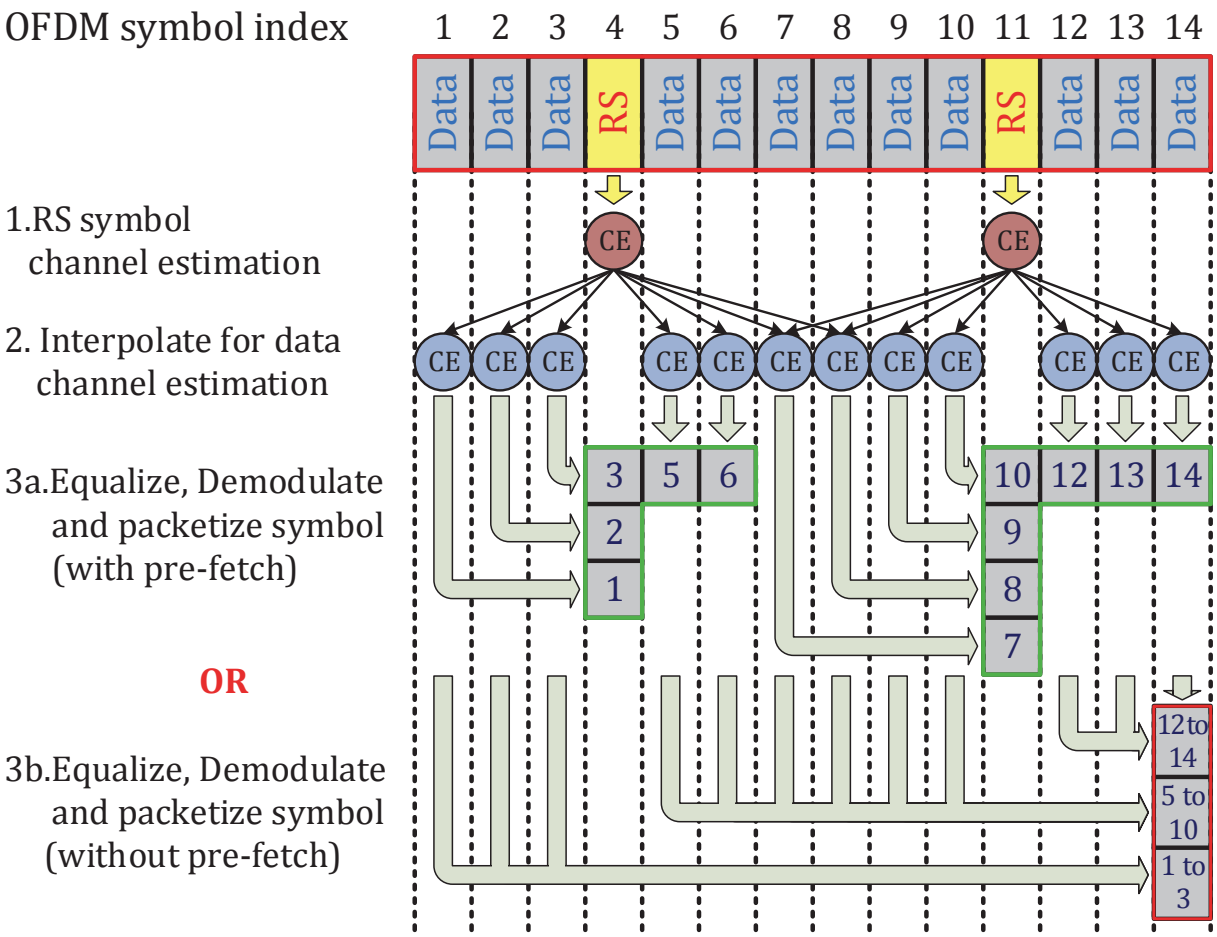


Figure 2.7: Pre-fetch scheme on the channel estimation and demodulation processing

2.4.2.3 Optimal payload size

As the third direction, we here consider the characteristic of using different maximum payload size, i.e., P. In practice, the value of P shall be less than the maximum payload size in an Ethernet jumbo frame, i.e., $J = 8960$ bytes. It has to be noted that using a smaller payload size can replace a large packet with few small packets to decrease RRU queuing delay but with extra overhead. Considering this trade-off, the optimal payload size that minimizes the FH delay (cf. T_{FH} in Eq. (2.2)) for each split can be known after exhaustive simulations; however, a simpler method to approximate this value is proposed when the FH link is *highly-loaded*. The FH delay (T_{FH}) is this "highly-loaded FH duration" will be enlarged, and thus the aforementioned HARQ timing constraint in Eq. (2.1b) may be violated. More specific, in this duration, the number of samples of the first symbol is much larger than the maximum payload size of an Ethernet jumbo frame, i.e., $E[x_{i,1}] \gg J$.

Our aim is to find the value of P that can minimize the FH delay T_{FH} via the derivative method shown in Eq. (2.4). First of all, only T_{Q_r} and T_{Q_s} are relevant to P, and thus we can ignore both T_{pkt} and T_{propo} terms. Since the FH link is highly-loaded, the summation of T_{Q_r}

and T_{Q_s} can be modeled as the sum of (1) the duration of transporting the first packet on edge FH segment, and (2) the serialization time of all packets on the aggregated FH segment. Note that $E[x_i] = \sum E[x_{i,j}]$ is the summation of the expected number of samples among all OFDM symbols in our concerned highly-loaded FH duration from the i-th RRU, and $M_i = \left\lceil \frac{E[x_i]}{P} \right\rceil$ is the expected number of packets within the highly-loaded FH duration from the i-th RRU.

To go one step forward, we can notice that the size of the first packet will equal to P, since the number of samples of the first OFDM symbol will be much larger than the maximum payload size of an Ethernet jumbo frame, when in a highly-loaded FH duration, i.e., $E[x_{i,1}] \gg J \geq P$. Thus, we can get the following two approximations: $P \approx \min(P, E[x_{i,1}])$ and $M_i \approx \frac{E[x_i]}{P}$. Then, we can get the optimal payload size that makes this first-order derivative equals to 0, i.e., P_{opt}. Before showing the result, we review two exceptional conditions.

1. For the splits that do not have this "highly-loaded FH duration" (i.e.., $E[x_{i,1}] \not\gg J$) like the Split D with either small LAD or pre-fetch scheme and the Split E, T_{Q_r} is irrelevant to the payload size P and the derivative in Eq. (2.4) will be always negative. Hence, the optimal payload size will equal to the maximum payload size of an Ethernet jumbo frame, i.e, $P_{opt} = J$.

2. For the splits with a higher data rate, e.g., Splits A ans B, the payload size shall be larger than a lower bound, i.e., P_{lb}, to avoid the FH overflow caused by the excessive packet overhead as shown in Eq. (2.5). Note that the $x = x_{i,j}, \forall i, j$, since these two splits have a constant number of samples in all OFDM symbols. Take the Split A as an example, if we apply a payload size lower than this bound $P_{lb} = 5011$, as will be shown in Section 2.6.2.1, then more packets will be generated and a significant portion of the FH capacity (i.e., r and R) will be wasted on the overhead O transportation.

Finally, we can write the optimal payload size as P_{opt} in Eq. (2.6).

$$\begin{aligned} \frac{\partial T_{FH}}{\partial P} &= \frac{\partial (T_{Q_R} + T_{Q_G})}{\partial P} = \frac{\partial \left(\frac{(min(P, E[x_{i,1}]) + O)}{r/8} + \frac{\sum_{i=1}^{N} (E[x_i] + O \cdot M_i)}{R/8} \right)}{\partial P} \\ &\approx \frac{\partial \left(\frac{(P+O)}{r/8} + \frac{\left(1 + \frac{O}{P}\right) \cdot \sum_{i=1}^{N} E[x_i]}{R/8} \right)}{\partial P} = \frac{1}{r/8} - \frac{O \cdot \sum_{i=1}^{N} E[x_i]}{R/8 \cdot P^2} = 0 \end{aligned} \tag{2.4}$$

$$P_{lb} = \frac{8 \cdot O \cdot x}{T_{sym} \cdot \min\left(r, \frac{R}{N}\right) - 8 \cdot x} \tag{2.5}$$

$$P_{opt} = \begin{cases} \max\left(\sqrt{\frac{O \cdot r}{R} \cdot \sum_{i=1}^{N} E[x_i]}, P_{lb} \right), & \text{If } E[x_{i,1}] \gg J \\ J, & \text{Otherwise} \end{cases} \tag{2.6}$$

Finally, Algorithm 2.1 summarizes the proposed packetization algorithm, in which I_{pack} represents current OFDM symbols shall be packetized immediately or not according to the condition in Eq. (2.3) and the pre-fetch scheme applicability, X_b refers the remaining bits in the packetization buffer, and P_{opt} is the optimal maximum payload size derived in Eq. (2.6).

Algorithm 2.1: Proposed packetization algorithm

Input : $x_{i,j}$ input samples in bytes for the i-th RRU at the j-th OFDM symbol
X_b bytes in the packetization buffer
P_{opt} is the optimal maximum payload size
Output: Transmitted packets
begin
  $x_{input} = x_{i,j}$ /* Get the current input samples */
  while $x_{input} > 0$ **do**
    $X_{ori,b} = X_b$; /* Get the original packetization buffer size */
    $X_b = X_b + \min(x_{input}, P_{opt} - X_b)$;
    $x_{input} = x_{input} - (X_b - X_{ori,b})$;
    if $X_b == P_{opt}$ **then**
      Send packet with payload equals to P_{opt} ;
      $X_b = 0$;
  if *Eq.* (2.3) *is satisfied* ***or*** *Pre-fetch is applied* **then**
    I_{pack} = true ;
  else
    I_{pack} = false ;
  if $I_{pack} ==$ *true* ***and*** $X_b > 0$ **then**
    Send packet with payload equals to X_b ;
    $X_b = 0$;

2.5 Impact of packet scheduling

In the considered C-RAN topology (cf. Figure 2.3), all packets from different RRUs in the uplink direction will be scheduled by the shared switch to be transported to the BBU pool. Moreover, to increase the efficiency of FH transportation, packets that are beyond the deadline set by the aforementioned HARQ timing constraint will be discarded. In following, we discuss these two schemes in more details.

2.5.1 Packet scheduling

Generally, the single-machine, multi-queue model can be applied at a switch, and the packets are stored in their respective queue and are processed following the first-in-first-out (FIFO) queue discipline, i.e., there is a one-to-one mapping between the i-th queue to the i-th RRU. Then, based on the applied scheduling algorithm, each packet will be scheduled and transported over the aggregated FH segment. A simple packet scheduling policy is to allocate packets based on their arrival time, i.e., first-come, first-served (FCFS). In this sense, all queues can be viewed as one FIFO queue that is shared by all RRUs. Before introducing other scheduling policies, we first define two useful metrics as following:

- *Unscheduled bits*: Specifically, this metric represents the number of unscheduled bits in the i-th FIFO queue that belonging to the same OFDM symbol as the head-of-the-queue packet, i.e., B_i.

- *Remaining time*: It represent the remaining time till the deadline for the head-of-the-queue packet, i.e., D_i.

Via utilizing these two metrics, several packet scheduling policies can be applied:

1. First-come, first-served (FCFS): Select the i-th queue with the first arrival time.
2. Shortest processing time (SPT): Select the i-th queue with the the minimum packet size at the head of the queue.
3. Least remaining bit (LRB): Select the i-th queue with the minimum B_i. This policy prioritizes the RRU with the minimum remaining unscheduled bits.
4. Earliest due date (EDD): Select the i-th queue with the minimum D_i.
5. Least slack time (LST): Select the i-th queue with the minimum slack, i.e., $S_i = D_i - \frac{B_i}{R}$. This policy further considers the remaining processing time than the EDD policy.

Beyond aforementioned qualitative descriptions on these scheduling policies, a quantitative analysis will be given in Section 2.6.2.2

2.5.2 Packet discard

When the slack of a queue is negative, i.e., $S_i < 0$, then at least one packets of the same OFDM symbol cannot be delivered to the BBU on time. Therefore, this packet and all other packets of the same TTI from the same RRU shall be discarded. This is because no further processing can be applied when the deadline expires. Thus, the NACK will be sent from the corresponding RRUs to all served UEs on the downlink control channel in order to trigger the uplink re-transmission. Besides, the same downlink data will be re-transmitted on the downlink data channel since no ACK will be received in the uplink direction.

2.6 Simulations

2.6.1 Parameter setup

2.6.1.1 Multi-cell resource provisioning bound

Before presenting the simulation results, we first highlight that the FH data rates of Splits C, D and E are related to the user channel quality statistics. The reason behind is that these splits transport the user data rather than the cell time- and frequency-domain I/Q samples by Split A and Split B respectively. In this sense, it is rare for all RRUs to always have the peak load at the same moment, and thus more RRUs can be supported after considering such effect. Several works try to model this effect, for example, the daily load measurement on six cells is provided in [172], and the authors suggest to use three levels (e.g., idle, medium and busy) to categorize the BS activities. Moreover, NGMN alliance provides two coarse levels of cell load (i.e., busy and quiet hours) and forms two bounds for multi-cell resource provisioning in [181], i.e., lower provisioning bound (LPB) and conservative lower bound (CLB).

To extend these two simple bounds, we first investigate the cumulative distribution function (CDF) of aggregated cell data rate under different user densities. In Figure 2.8, the average and

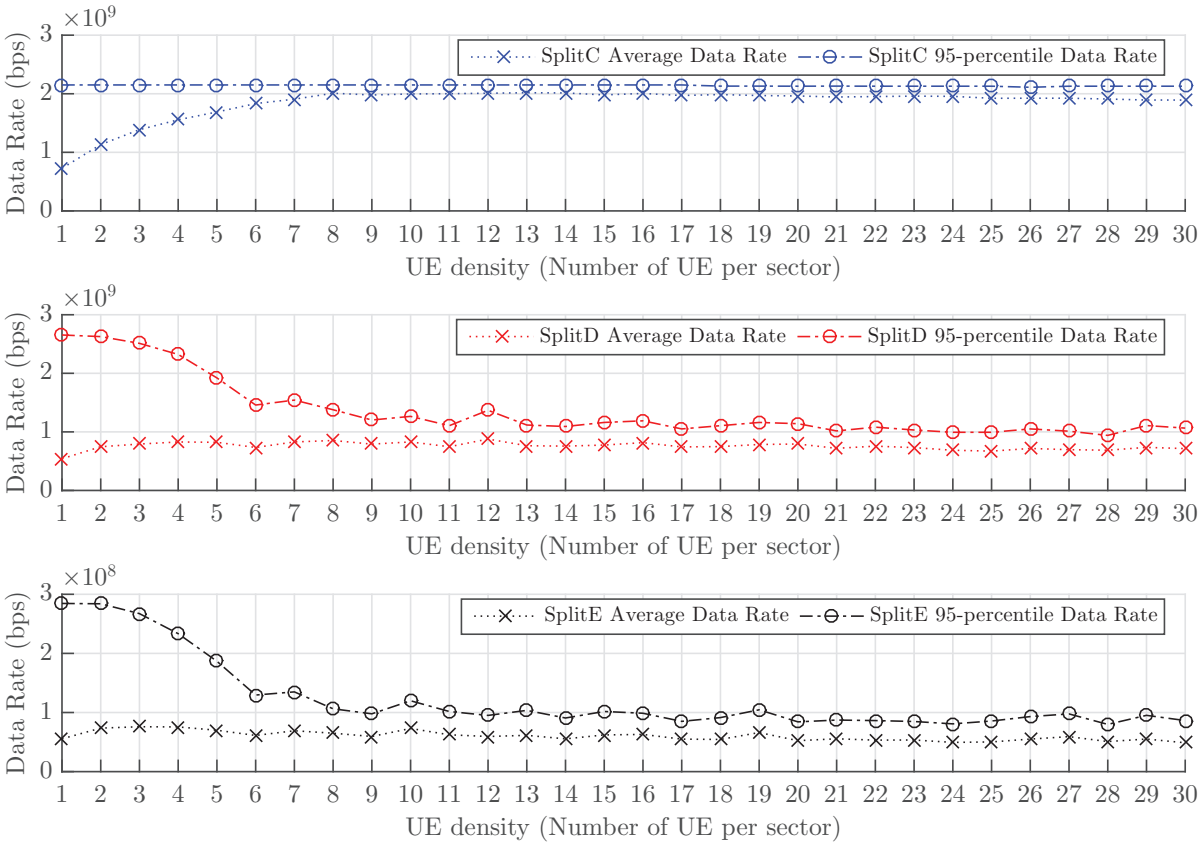


Figure 2.8: Data rate for different user density

95^{th} percentile cell data rate (i.e., data rate is below this value for 95% of the time) for different user density values are provided for the aforementioned three functional splits. We can observe that the 95^{th} percentile results belonging to different user density values could be divided into three distinct regions:

- Low density (LoD) (i.e., User density $\in [1, 2]$): $R_{LoD}^{95^{th}}$
- Medium density (MeD) (i.e., User density $\in (2, 10]$): $R_{MeD}^{95^{th}}$
- High density (HiD) (i.e., User density $\in (10, 30]$): $R_{HiD}^{95^{th}}$

Then, two extra bounds for overall N cells are proposed by utilizing these three regions, i.e., aggregated common bound (ACB) in Eq. (2.7a) and aggregated strict bound (ASB) in Eq. (2.7b), compared to the legacy LPB (LPB $= N \cdot R_{HiD}^{95^{th}}$) and CLB (LPB $= R_{LoD}^{95^{th}} + (N-1) \cdot R_{HiD}^{95^{th}}$).

$$\text{ACB} = R_{LoD}^{95^{th}} + 6 \cdot R_{MeD}^{95^{th}} + (N-7) \cdot R_{HiD}^{95^{th}} \tag{2.7a}$$

$$\text{ASB} = \frac{N}{19} \cdot \left(R_{LoD}^{95^{th}} + 6 \cdot R_{MeD}^{95^{th}} + 12 \cdot R_{HiD}^{95^{th}} \right) \tag{2.7b}$$

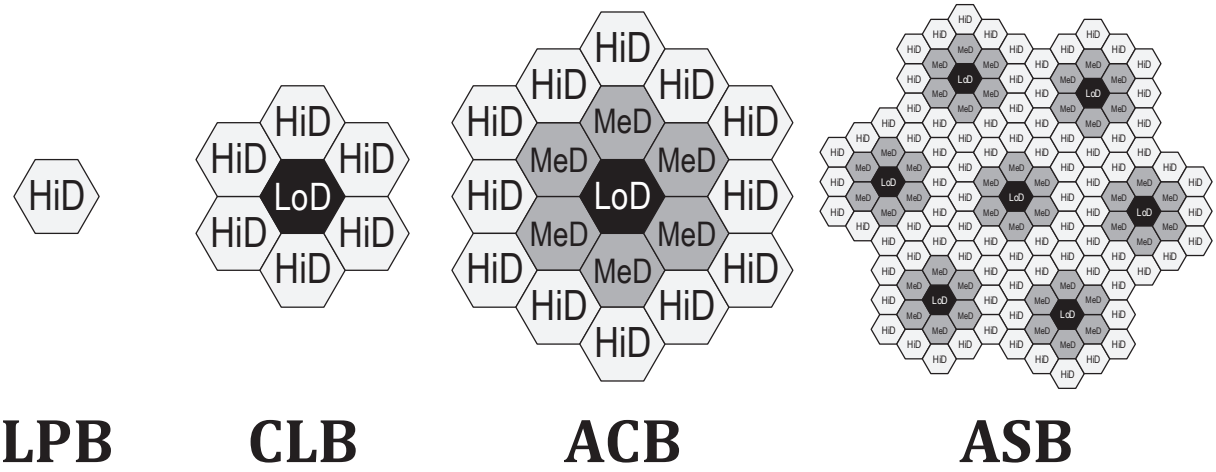


Figure 2.9: Different multi-cell resource provisioning bounds on hexagonal cell planning

To visually represent these bounds, we refer to the hexagonal cell deployment geometry in which the central BS is with a low user density, BSs of the second ring are with medium density, and other BSs are with the high user density, as illustrated in Figure 2.9. The LPB is formed only from high user density BSs and CLB further extends the LPB via including one central BS with low user density. As for the ACB, it further considers the second ring of BSs with medium user density. Last but not least, the BSs with low, medium and high levels of user density for the ASB case will follow the explicit discrete distribution: $\left[\frac{1}{19}, \frac{6}{19}, \frac{12}{19}\right]$.

2.6.1.2 Simulation parameters

Besides aforementioned multi-cell resource provisioning bounds, most of the simulation parameters applied to the UE, RRU, and BBU are taken from the 3GPP standards in [182–184] and the NGMN documents in [185]. To provide a fair comparison with the aforementioned peak-rate analysis, we apply the same RRU setting as the Scenario 1 listed in Table 2.1, in which both 64QAM and 4-layer MIMO are used. Moreover, we apply the full-buffer traffic model for uplink data transportation and the random walk model for user mobility. Further, details on the simulation parameters are listed in Table 2.4.

2.6.2 Simulation results

Numerical results and discussions of underlying insights on the number of supported RRUs over a capacity-limited FH are presented based on the MATLAB simulator. Notice that the number of additional supported RRUs shows the multiplexing gain on how much additional data rate we could push to the FH network. Quantitatively, we use the 95^{th}-percentile of the FH delay to decide the number of supported RRUs, i.e., 95% of the time over all RRUs can satisfy the aforementioned constraint for the FH transportation, and compare this value with the peak-rate analysis results in Table 2.2 for the multiplexing gain G_{rru} formulated in Eq. (2.8).

$$G_{rru} = \frac{\text{Number of supported RRUs after considering the packetization impact}}{\text{Number of supported RRUs in peak-rate analysis}} \tag{2.8}$$

Table 2.4: Simulation parameters

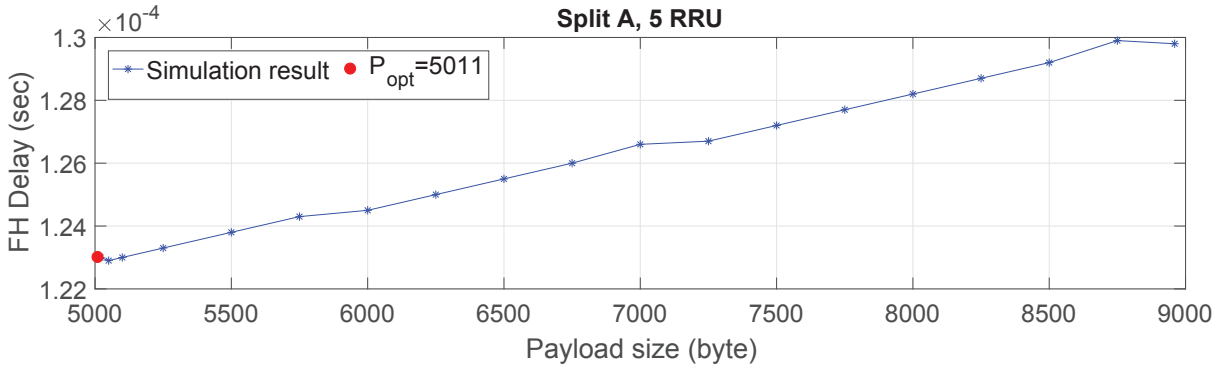
Parameter	Value
Carrier frequency	2.0 GHz
Radio Bandwidth	20 MHz
Cyclic Prefix length	Normal
Uplink Tx/Rx Antenna number	4 Tx antenna / 4 Rx antenna
Uplink transmission mode	2
Inter-site distance	500 meter
Pathloss model	Urban model
Thermal noise density	-174dBm/Hz
Minimum coupling loss	70dB
Shadowing mean	0
Shadowing standard deviation	8
Inter-cell shadow correlation	0.5
Inter-sector shadow correlation	1.0
Propagation channel model	EPA
Initial UE distribution	Uniform distribution
UE speed	Randomly selected from [3, 30, 120] km/hr
UE direction	Uniform distributed in [0, 360] degree
UE antenna gain	0dB
Cell antenna pattern	3-sector cell pattern
Cell antenna gain	15dB

Table 2.5: Number of supported RRUs for Splits A and B

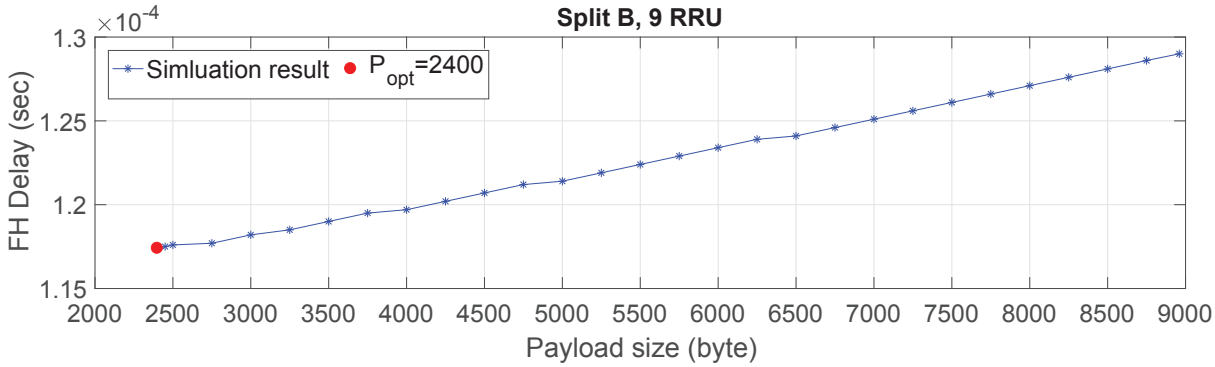
Functional split	Peak-rate analysis	Packetization impact
Split A	5	5
Split B	9	9

2.6.2.1 Impact of packetization

Split A and split B Due to the high data rate characteristic of these two splits, the optimal payload size P_{opt} will equal to its lower bound, derived as $X_{lb} = 5011$ and $X_{lb} = 2400$ for Splits A and B, respectively. Via substituting this optimal payload size from the maximum payload size of an Ethernet jumbo frame, the FH delay (i.e., T_{FH}) will be reduced by 5% and 9% respectively as shown in Figure 2.10a and Figure 2.10b, respectively. As mentioned beforehand, using a smaller payload size than this lower bound is not feasible since the FH link will be overflowed by the abundant packetization overhead. For instance, using the standard Ethernet frame with 1460 bytes payload size is not feasible. However, the number of supported RRUs in Table 2.5 is identical to the results from the peak-rate analysis due to the constant data rate characteristic of these two splits, i.e., these two splits transport cell I/Q samples. To sum up, no multiplexing gains can be observed for these two splits, i.e., $G_{rru} = 1$.



(a) Impacts of optimal payload size on Split A



(b) Impacts of optimal payload size on Split B

Figure 2.10: Impacts of optimal payload size on Splits A and B

Table 2.6: Number of supported RRUs for Split C

Functional split	Bound	Peak-rate analysis	Packetization impact
Split C	LPB	9	10
	CLB	9	10
	ACB	9	10
	ASB	9	11

Split C The impact of optimal payload size on the FH delay is shown in Figure 2.11, and we can see that the optimal payload size $P_{opt} = 4363$, which is very close to the local minimum point at around $[4250, 4500]$ through the extensive simulations. Moreover, the FH delay can be reduce by 12% after replacing the Ethernet jumbo frame payload size J with the optimal payload size.

Table 2.6 shows the number of supported RRUs compared with the peak-rate analysis results under different multi-cell resource provisioning bounds. We can see that after considering these bounds and the impact of packetization, the number of supported RRU is slightly increased. This result reveals the multiplexing gain of the proposed packetization process when compared with the peak-rate analysis results, i.e., $G_{rru} = 1.11$ (LPB, CLB, ACB) and $G_{rru} = 1.22$ (ASB).

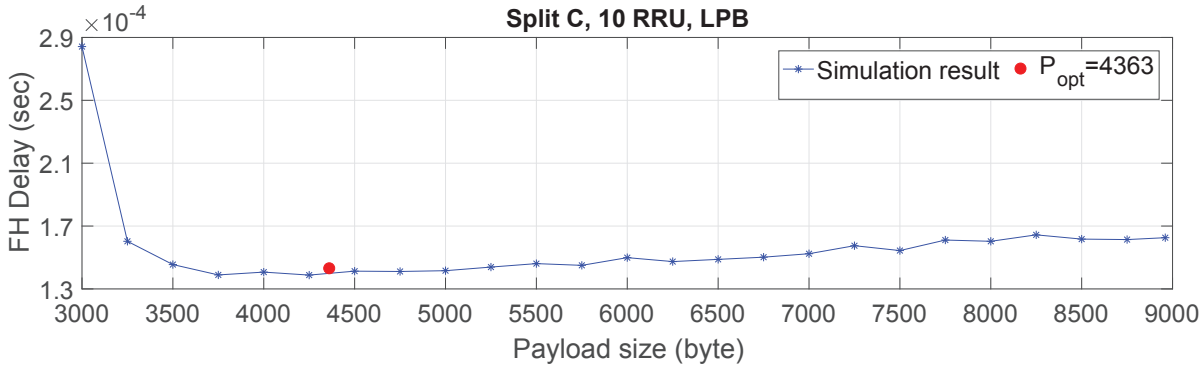


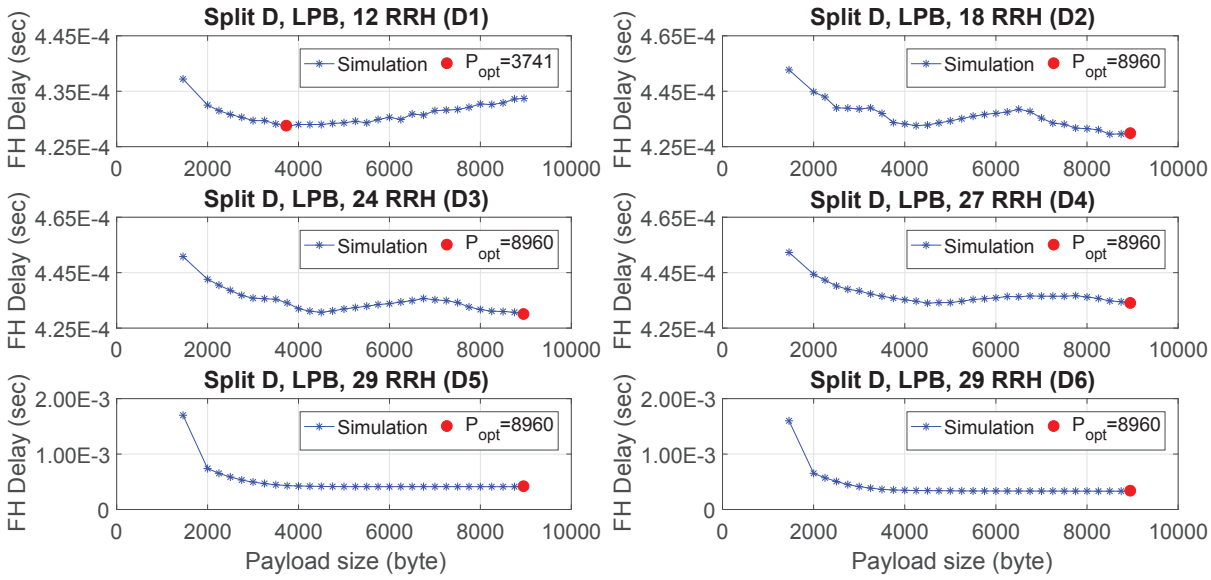
Figure 2.11: Impact of optimum payload size on Split C

Table 2.7: Number of supported RRUs for Split D

Functional split	**Bound**	**Peak-rate analysis**	**Packetization impact**					
			D1	D2	D3	D4	D5	D6
Split D	LPB	7	12	18	24	27	29	29
	CLB	7	11	17	24	27	29	29
	ACB	7	12	18	22	24	26	26
	ASB	7	13	19	25	27	29	29

Split D The impact of optimal payload size on the FH delay is presented in Figure 2.12 with six different modes (from D1 to D6), each with different parameters in terms of the operating period, the LAD and the pre-fetch scheme. Since the highly-loaded FH condition can only be fulfilled in the D1 mode (with the largest LAD and no pre-fetch scheme), we can see that the optimal payload size $P_{opt} = 3741$ closely matches the local minimum point through the simulation approach. However, the optimal payload size of other modes (from D2 to D6) will equal to the maximum payload size of an Ethernet jumbo frame $P_{opt} = J = 8960$ according to Eq. (2.6), beling aligned with the simulation results.

The number of supported RRUs is shown in Table 2.7. We can see that after considering the multi-cell resource provisioning bounds and the impact of packetization, the number of supported RRU is greatly increased. More specific, the pre-fetch scheme can bring 50% gain, when comparing the results between D1 mode and D2 mode. It justifies that the pre-fetch scheme can significantly reduce the instantaneous congestion on the FH interface. Furthermore, as the LAD is decreased from 10 (D2 mode) to 3 (D6 mode), the number of supported RRUs can almost be doubled due to the further reduction on the bursty traffic transportation. Nevertheless, using a smaller LAD may degrade the channel estimation interpolation accuracy which is out of our scope in this thesis. To sum up, a much higher multiplexing gain can be observed in Split D and G_{rru} can reach up to 3.71 (ACB) and 4.14 (LPB, CLB, ASB) when applying either D5 mode or D6 mode. That is to say, we can support up to 3-times more RRUs in the considered network topology of C-RAN.



Mode	Operating period	Look-ahead depth	Pre-fetch
D1	subframe	10	No
D2	subframe	10	Yes
D3	subframe	6	Yes
D4	subframe	5	Yes
D5	subframe	4	Yes
D6	slot	3	Yes

Figure 2.12: Impact of optimal payload size on different modes of Split D

Table 2.8: Number of supported RRUs for Split E

Functional split	Bound	Peak-rate analysis	Packetization impact
Split E	LPB	66	153
	CLB	66	154
	ACB	66	141
	ASB	66	142

Split E As for this split, the highly-loaded FH condition is neither fulfilled, and thus the optimal payload size will be $P_{opt} = J = 8960$. Then, Table 2.8 shows the number of supported RRUs, and we can observe up to 2.3-fold improvement, i.e., $G_{rru} = 2.3$. The reason behind is that the peak-rate scenario rarely happens, and thus the number of supported RRUs can be increased via applying the packetization and guaranteeing 95% of the FH delay can satisfy the HARQ timing constraint.

Table 2.9: Number of supported RRUs among several packet schedule policies

Functional split	Bound	Scheduling policy				
		FCFS	**SPT**	**LRB**	**EDD**	**LST**
Split E	LPB	153	156	162	153	140

2.6.2.2 Impact of scheduling

To go one step further, we further investigate the impact of packet scheduling in this paragraph. Not that the above investigation results only consider the simplest FCFS scheduling at the switch; that is to say, every packet is scheduled to be transported over the FH based on their arrival time. This FCFS scheduling policy can neither be fair to RRUs of different loads nor exploit extra multiplexing gains to support more RRUs. To this end, several other scheduling policies aforementioned in Section 2.5.1 are examined.

For simplicity, we focus only on the Split E with LPB bound, but the same trends can be viewed when applying different configurations. Table 2.9 then shows the number of supported RRUs when applying different packet scheduling policies and packet discard scheme. It can be seen that the LRB policy can support the most RRUs (162 RRUs), since it prioritizes the RRUs with the fewest remaining bits, i.e., the lightly-loaded RRUs. The same trend can be observed in the empirical fronthaul delay CDF of Figure 2.13: The LRB policy shows the lowest 95^{th} percentile of FH delay. For the SPT policy, there are fewer improvements in terms of the number of supported RRUs (cf. Table 2.9) and the 95^{th} percentile of FH delay (cf. Figure 2.13) when compared with the LRB policy, since the priority is given to the smaller packets rather than the lighter-loaded RRUs. On the other hands, the LST policy shows completely different trend. Since the LST policy will prioritize the RRUs with less slack, i.e., the minimum allowable processing time till the deadline, and thus the FH delay from different RRUs can be more balanced and the variance of the FH delay is largely reduced as shown in Figure 2.13. Nevertheless, this advantage is at the cost of supporting only 140 RRUs as shown in Table 2.9, which is even less than the number of supported RRUs of FCFS policy.

Moreover, we show the Jain's fairness index and the packet discard statistics in Figure 2.14 for all policies. According to the fairness comparison in Figure 2.14a, we can see that the LST policy shows the best fairness in terms of the average FH delay among RRUs, corresponding to the smallest FH delay variance in Figure 2.13. Moreover, the LST policy also has the best fairness in terms of the queue size due to the slack metric is in negative proportional to the number of unscheduled packets. In comparison, the LRB policy shows the worst fairness in terms of the queue size. Further, the packet discard statistics are shown in Figure 2.14b in terms of the number of discarded packet and the total discarded bits. We can observe that the LST policy discards more packets but with fewer total discarded bits; the reason behind is that most of smaller packets with negative slack values are discarded. In contrast, the total discarded bits are large for LPB policy since it is more unfair to the heavily-loaded RRUs.

In summary, these policies can be categorized into two groups depending on their trade-offs between the fairness and the multiplexing gain. On one hand, both LRB and SPT policies aims to increase the multiplexing gain and thus the fairness among different RRUs is decreased. On the other hand, the LST and EDD policies try to be fairness to different RRUs but at the cost of supporting fewer number of RRUs.

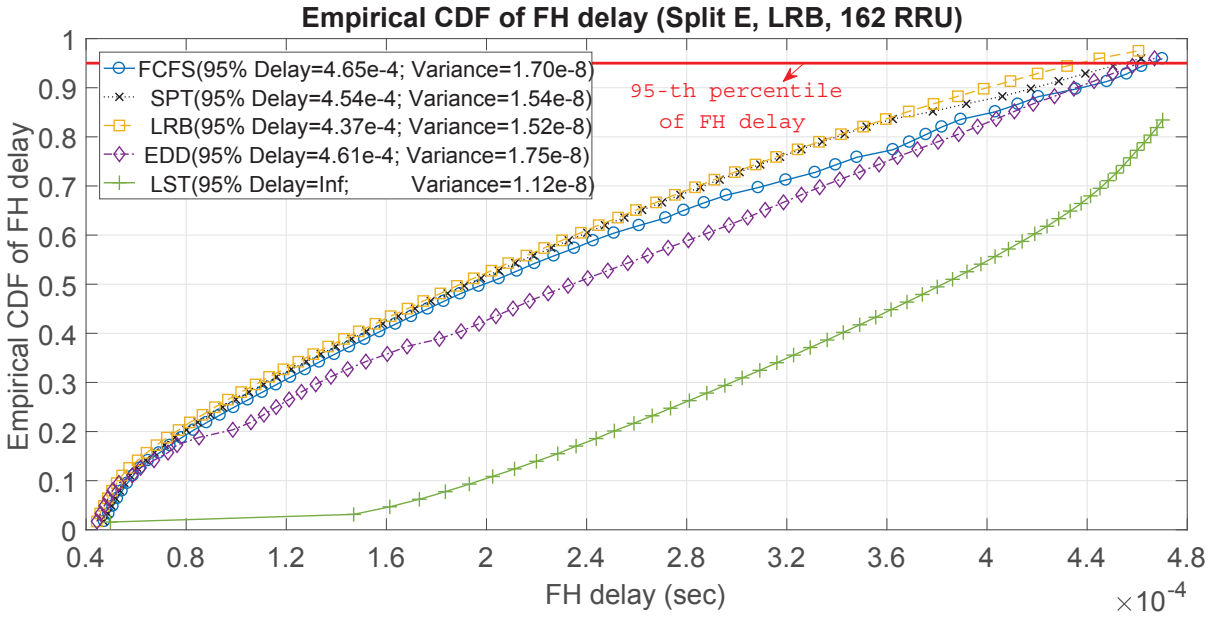
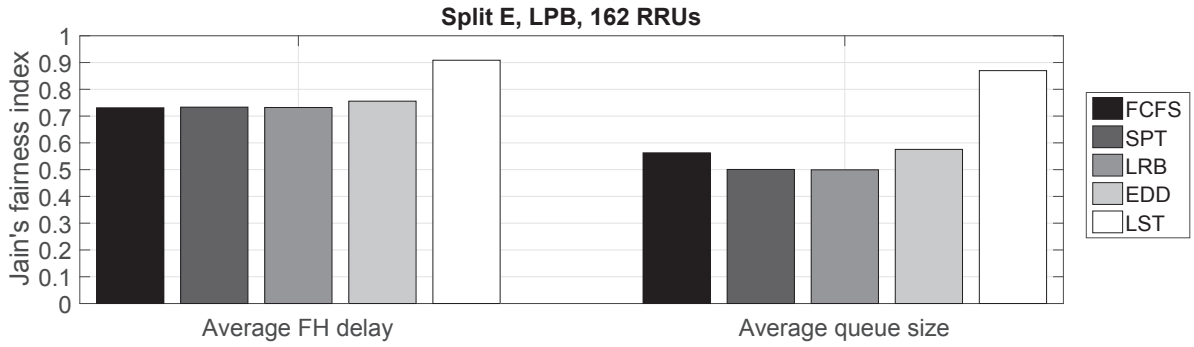
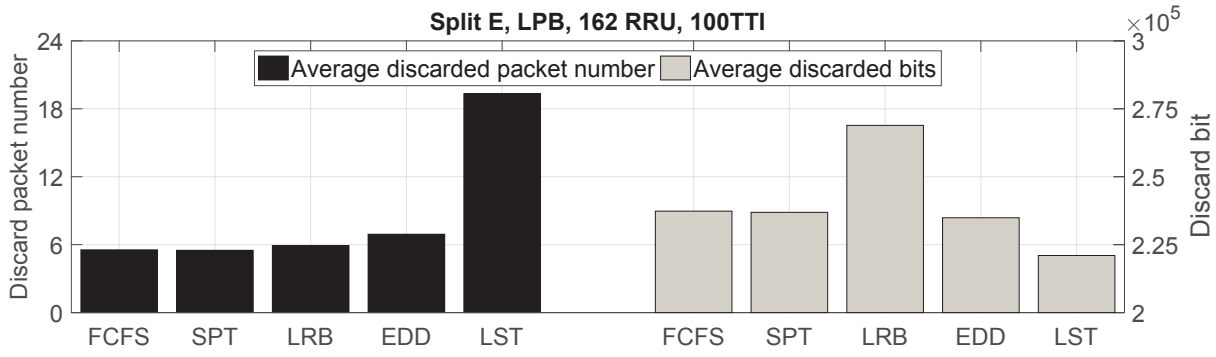


Figure 2.13: Empirical CDF of FH delay among several packet scheduling policies



(a) Fairness comparison



(b) Packet discard statistics comparison

Figure 2.14: Fairness and packet discard statistics among several packet schedule policies

2.7 Discussions

In the aforementioned study, the impacts of functional split, packetization and packet scheduling on the multiplexing gain and fairness are conducted. To further extends these results, we provide two possible directions that are elaborated in the following paragraphs.

The first direction is to extend the multiplexing gain for the capacity-limited FH network defined in this chapter to further consider the limitations of a computing-limited C-RAN deployment. Such vision is aligned with the RAN disaggregation notion introduced in Chapter 1, in which the disaggregated computing resources are taken into account. After further considering such limitation, we can expect that the multiplexing gain can be further enlarged, since the computing resource centralized at the BBU can be flexibly utilized for the processing of more RRUs, depending on their loads. Note that this enlarged multiplexing gain is also dependent on the functional splits (e.g., no multiplexing gain for Splits A and B), and thus a joint consideration on the applied functional splits is needed. Last but not least, we can also explore the trade-off between multiplexing gain and fairness via considering the task scheduler in the computing-limited C-RAN deployment.

The second direction is to investigate the adoption of Ethernet using some state-of-the-arts experimental platforms, such as OpenAirInterface [164]. The reason behind is to examine the real-world deployment issues, such as the UP performance impacts, the influences of different FH transportation protocol, the synchronization issues for a number of RRUs, the coexistence with non C-RAN traffic flows, and so forth. In order to deal with these issues, a small- to medium-scale C-RAN deployment shall be built and investigated together with the commercial-of-the-shelf user equipments to clearly examine the end-to-end performance impacts resulted from the C-RAN deployment.

2.8 Conclusions

In this chapter, we first provide a thoroughly review on the C-RAN system model and the inherent challenges. Practically, we propose the use of a packet-based FH network and study the joint impact of different packetization methods and functional splits on the achievable multiplexing benefits in terms of the number of RRHs one could support. Moreover, we provide the packetization algorithm over the FH links and analyze various packet scheduling policies applied at the aggregated switch. The numerical results justify our proposed packetization algorithm and show that there is a potential interplay between fairness and multiplexing gain when using different packet scheduling policies.

Chapter 3

Flexible Functional Split Framework over Ethernet Fronthaul in C-RAN

3.1 Introduction

As mentioned in the previous chapter, the trade-off between the level of centralization and the FH requirements is noticed among different functional splits. For instance, the Split A can have the full centralization benefits at the cost of a extremely high FH capacity requirement, while the higher level splits can largely reduce the FH capacity and latency requirements but with fewer or no possibilities on the centralized processing. This trade-off between functional splits inhibit us from deriving only one ultimate functional split to deal with multiple services in a heterogeneous environment. Hence, one intuitive idea is to devise a flexible functional split to support an on-the-fly split (re-)configuration of baseband functionality (e.g., adapting to current traffic conditions and/or service requirements) to increase the flexibility for different types of deployment and service requirement. To this end, the steps of RAN evolution is depicted in Figure 3.1, in which the legacy D-RAN can be (1) fully centralized as *prototype C-RAN*, (2) partial centralized based on the applied functional split as *Split-enabled C-RAN*, and (3) flexible centralized between RAN entities as *flexible C-RAN*. This evolution exploits both softwarization and virtualization techniques as tailored in Chapter 1.

Take the ultra-dense networks (UDNs) deployment [186] as an example, in which densely-deployed BSs can provide seamless coverage and increased spectral efficiency at the cost of abundant capital expenditures. What is even worse is that the UDN deployment requires a low-latency interface to perform the coordinated processing and interference management among these small cells. Via leveraging the flexible functional split concept, we can apply either full centralization to boost the user/network performance when the FH network requirements are satisfied or the partial centralization to exploit the user proximity to the RRUs when the FH network has limited capability. For instance, the ideal FH network can provide low-latency and high-capacity links to support different functional splits all the way to full centralization/coordination; however, the non-ideal FH network can only provide medium-latency and medium-capacity links to allow for partially centralized deployments. A specific comparison between using D-RAN or flexible C-RAN for the UDN deployment is shown in Table 3.1. To conclude this example, We can see that the flexible C-RAN is more agile and versatile than the legacy D-RAN in several aspects to enable the UDN deployment.

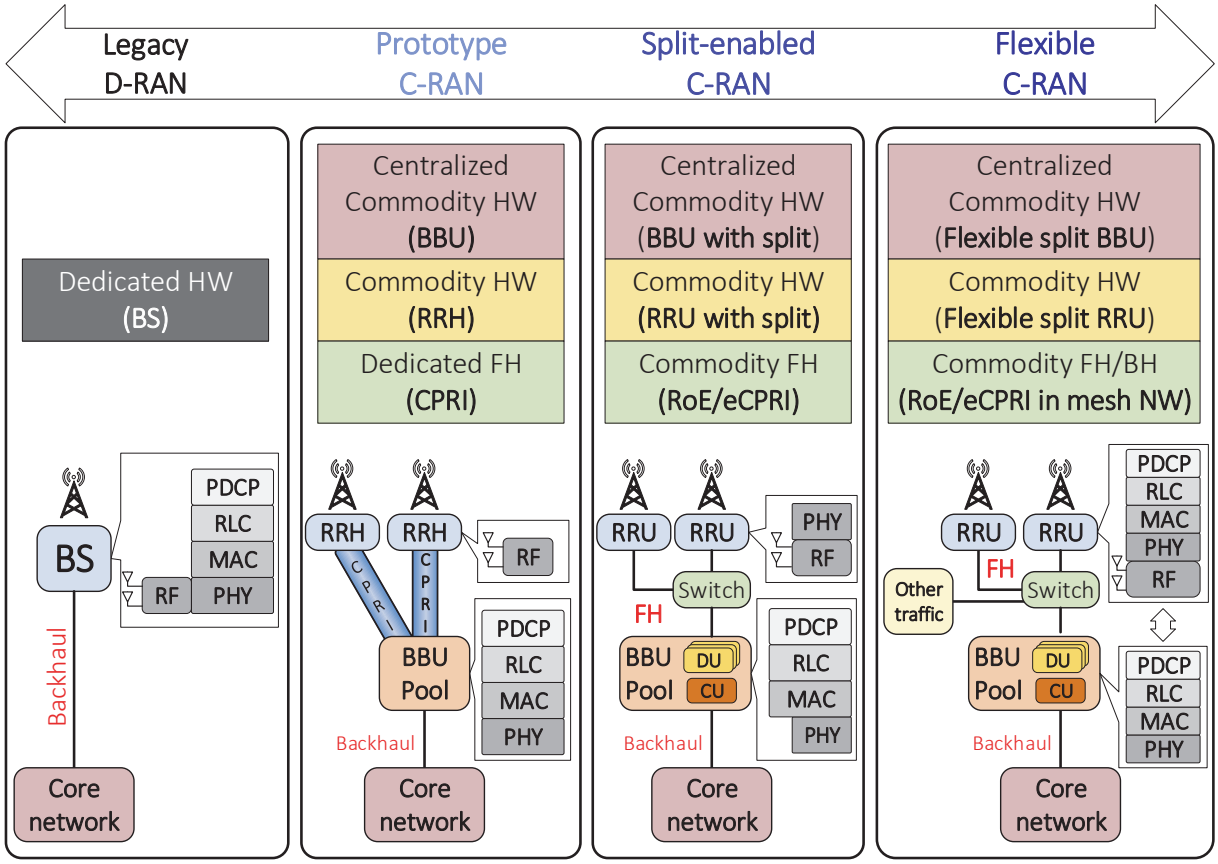


Figure 3.1: RAN evolution from Legacy D-RAN to flexible C-RAN

Table 3.1: Compare C-RAN and D-RAN within the UDN deployment

Impacts on UDN	Flexible C-RAN	D-RAN
RAN edge infrastructure	Dedicated and/or general purpose infrastructure	Dedicated full protocol stack radio access node
RAN node densification	Flexible densification among both RAN edge infrastructure (RRU) and central entity (BBU)	Only densifies RAN edge infrastructure (BS)
Coordination capability	Centralization level based on functional split	Depend on the connected interface quality [187]
Deployment flexibility	Support flexible split (re-)configuration	Not applicable
Split of CP and UP	Naturally achievable due to its centrality manner in heterogeneous UDN	Need low-latency collaboration between macro and small cells
Multi-operator and multi-vendor sharing	Dynamic and flexible sharing between C-RAN entities (RRU, BBU)	Pre-defined sharing policies such as MOCN and MORAN
Edge computing compatibility	Co-deployable with MEC entity to form a F-RAN	Require dedicated MEC entity with specific protocols

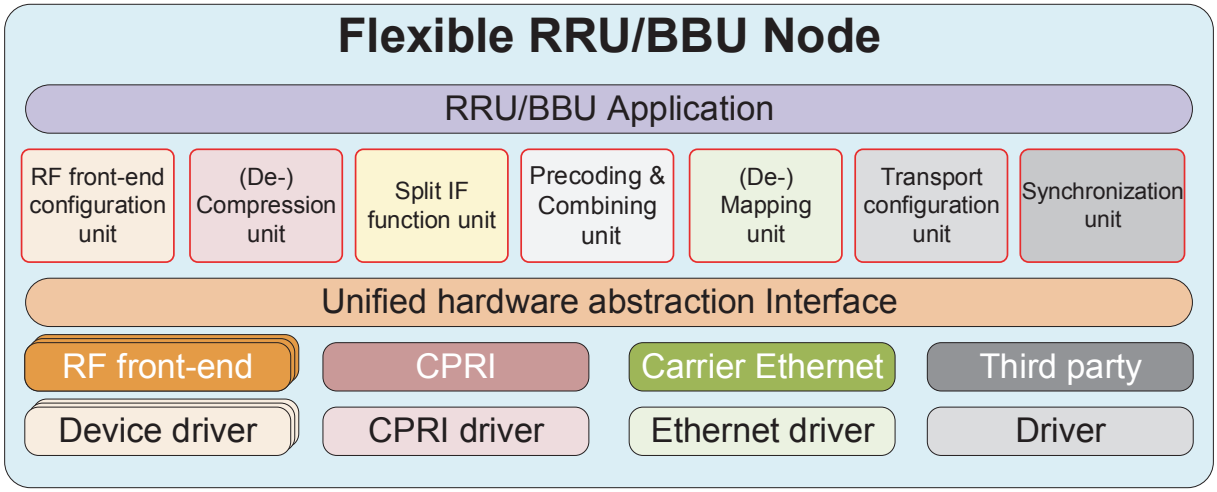


Figure 3.2: Proposed flexible RRU/BBU architecture

Corresponding to the aforementioned benefits via utilizing the flexible RRU/BBU deployment, we propose a flexible RRU/BBU architectural framework in this chapter that can support a flexible functional split together with an Ethernet-based transport protocol over the FH link. It is noted that there are some works [188–190] and standardization bodies [170, 171] discussing on using a packet-based Ethernet for the FH transportation. However, there is no prior arts investigating the real implementation for the packet-based transportation and supporting the flexible functional splits. To that end, we implement the envisioned C-RAN network architecture using the OAI platform, and evaluate several important KPIs within a subset of C-RAN network deployment scenarios.

3.2 Proposed C-RAN Framework

While RRUs require dedicated RF front-end devices and synchronization mechanisms with the centralized BBU, both BBU and RRU share the same set of baseband functions in order to provide the flexible functional split between these two entities and to support a number of heterogeneous deployments, e.g., UDN, distributed antenna system (DAS), massive MIMO. In addition, the associated Ethernet-based FH interface needs to be designed to support the RRU-BBU dynamic associations. Towards these two directions, we provide a **flexible RRU/BBU architecture** as well as an **Ethernet-based FH transport** protocol that are implemented in the OAI platform.

3.2.1 Designed flexible RRU/BBU architecture

In general, the RRU is an entity that hosts multiple RF front-end devices, processes the incoming samples based on the applied functional split, and transmits/receives data samples through the connected FH interface. These functions are also (mostly) mirrored in the BBU as well. Specifically, our proposed flexible RRU/BBU architecture is shown in Figure 3.2 and it comprises the following main components:

- *RF front-end configuration and monitoring unit*: It is responsible to apply the configuration (e.g., Rx/Tx gains, Rx/Tx operating frequencies, and so forth), indicated by the BBU, to the RF front-end equipment and to provide the status report to the BBU. Thus, it serves as an agent on behalf of the BBU for the (re-)configuration and monitoring of the underlying RF front-end devices.

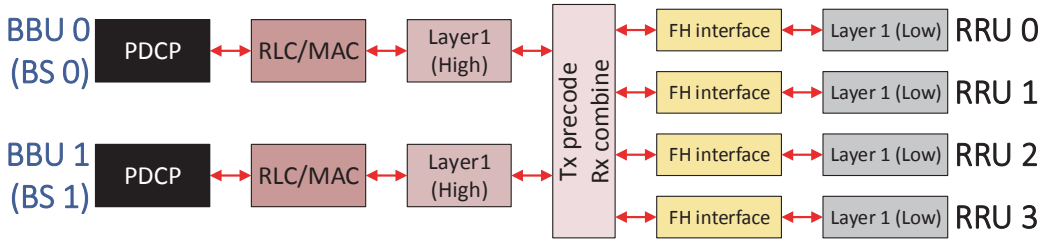
- *Split interface function unit*: It performs the split-specific processing on the incoming data samples in both UL and DL directions based on the applied functional splits. Practically, this applied functional split is (re-)configured through a in-band mechanism by the BBU. Moreover, there are two ways to deploy the network functions at RRU: (i) *Split-specific deployment* that only deploys the necessary functions at RRU to save the expenditures but with the loss in terms of fewer flexibility (cf. Split-enabled C-RAN in Figure 3.1), or (ii) *Flexible-split deployment* that deploys all baseband functions at the RRU (i.e., similar to the legacy BS) to allow on-the-fly update of functional splits (cf. Flexible C-RAN in Figure 3.1).

- *(De-)Compression unit*: It provides a (de-)compression service for the data samples in order to lower the FH capacity requirements. This compression service is configured by the BBU. More specifically, two compression approaches can be supported: (i) *lossless compression* that can reconstruct original data perfectly, and (ii) *lossy compression* that permits reconstructing an approximate version of the original data. In this chapter, we use a simple lossy compression on the data samples to significantly decrease the FH throughput at the cost of negligible user plane performance degradation.

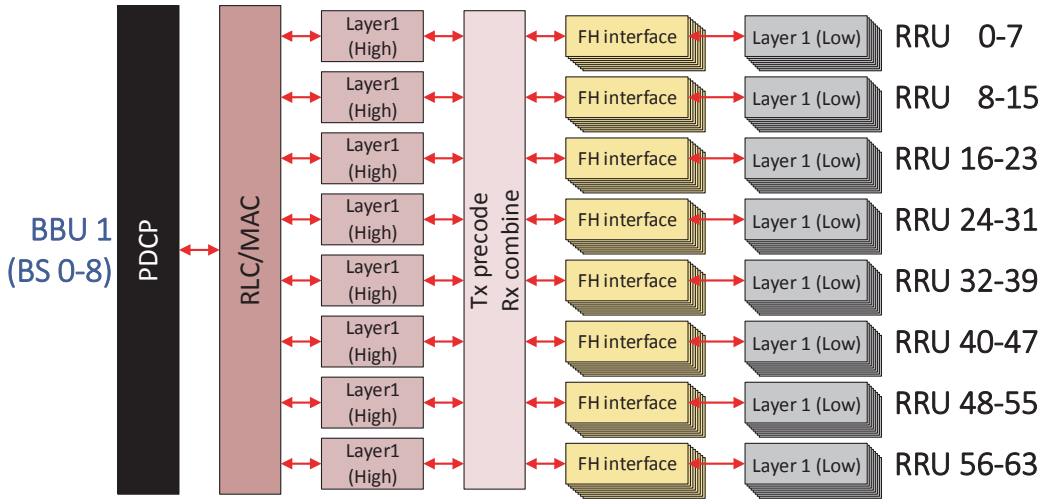
- *Precoding/Combining unit*: It can provide the transmitter precoding capability that supports jointly multiple BS and/or multiple user transmission on available RRUs within the same RRU cluster. Note that such precoding capability can be done at either RRU and BBU side according to the applied transmission modes (TMs). Moreover, it can combine among all receiving antennas from RRUs within the same cluster in the UL direction, allowing for the spatial multiplexing enhancement and handover minimization.

- *(De-)Mapping unit*: It maps the corresponding BBU to each connected antenna port of the RRU relying on the extra control information within the packet header. In this sense, the extra antenna identity is included in the packet header to successfully deliver the packet through the corresponding interface to/from the targeted remote BBUs. This antenna identity can be further combined with the component carrier identity as the antenna-carrier (AxC) combinations per data flow.

- *Transport configuration unit*: This unit serves two purposes. First of all, it applies the packetization scheme chosen by the BBU, i.e., the packet payload size and the network maximum transmission unit (MTU) for the FH interface. As tailored in Chapter 2, the payload size shall be selected properly to guarantee the FH transportation efficiency. Secondly, it adjusts the timestamp between the RRU and BBU, with respect to the round trip time (RTT) statistics continuously measured between these two endpoints. The timestamp of each packet is generated to have a reference clock of the RF front-end device. When a packet reaches the BBU pool, it bears the time value of the RF front-end device when the payload was generated. On the other hand, when a packet reaches RRU, it is stamped with an adjusted version of this timestamp.



(a) DAS system



(b) Massive MIMO

Figure 3.3: Two C-RAN deployment examples

- *Synchronization unit*: It enables the synchronization mechanisms to provide a reliable frequency distributed from the BBU across several RRUs. The PTP of IEEE 1588 protocol [191] can be used to provide a precise synchronization through a grandmaster (i.e., BBU) acting as a time server.

Corresponding to aforementioned seven units, there are seven categories of RRU parameters configured/reconfigured by the BBU: (a) RF front-end parameters, (b) DL/UL baseband and split parameters, (c) (De-) Compression parameters, (d) (De-) Mapping parameters, (e) Precoding parameters, (f) FH interface transport parameters, and (g) Synchronization information. Moreover, a unified hardware abstraction interface is provided to read/write data samples from/to different types of commercial-of-the-shelf (COTS) RF front-end devices, Ethernet device, and CPRI device. This common interface enables the RRU/BBU to stream incoming/outgoing data samples through different interfaces (i.e., RF frontend, CPRI, Ethernet). Further, the RRU/BBU application on the top of the framework can locally configure/reconfigure the underlying architecture and retrieve information for the control and management purpose.

The aforementioned RRU/BBU architecture is applicable to several deployments. Generally, it can even be applied in the legacy D-RAN deployment as monolithic BS via only maintaining necessary units (i.e., RF front-end configuration and monitoring unit, split interface function unit, precoding and combining unit) and interface (i.e,, RF front-end for air-interface and Carrier Ethernet backhaul). Moreover, we provide two examples shown in Figure 3.3 as:

1. *DAS system*: In Figure 3.3a, there are 2 BS instances precoded into 4 RRUs that are distributed using individual FH interface. These 2 BSs will use the corresponding MAC/RLC instance as separate virtual cells.
2. *Massive MIMO*: In Figure 3.3b, 8 BSs can be seen via using individual component carriers with a single MAC/RLC instance for scheduling. These BSs are further precoded over 64 antenna elements, interpreted as collocated RRUs in a massive MIMO deployment.

3.2.2 Ethernet-based fronthaul transportation

The design aims to provides a flexible and packet-based networking solution in the FH network. Our approach is aligned with the standardization directions [170,171,173] and it can be extended for more complex C-RAN topologies. A top-down description of each layer is as followings:

- **Topology Layer**: The relation between BBU and RRU from a high-level perspective follows the client-server model, where the BBU is client and the RRU is server. Moreover, a BBU can be associated to a set of RRUs (1:N relation), possibly asymmetric in receiver and transmitter, depending on the desired coordinated transmission and reception modes.
- **Interface Layer**: The FH interface consists of two logical streams:
  1. *Control*: It carries packets for in-band or out-of-band control used for RRU configuration and management. Out-of-band control is mainly used for parameter setup at the RRU side, during the configuration setup period.
  2. *Data*: It carries the compressed data samples that are packed in the payload part within the Ethernet frames.
- **Session Layer**: The RRU and BBU are mapped statically for simplicity. It means that the association between them is pre-defined, so there is no need for network discovery and link set-up. However, this pre-defined association can be extended easily with the addition of discovery and handshake messages, to be exchanged in the beginning of the session. What is currently in place, regarding the RRU-BBU session, is the parameter configuration on the RRU side done by the BBU. In particular, the BBU sends a set of parameters to configure the RRU and, as soon as the RRU acknowledges its successful reception, the data samples are able to be transported between these two endpoints.

  In Figure 3.4, the four periods and the associated steps are identified within the session between RRU and BBU. We can see that the steps 1a, 1b, 2, and 4 are performed in the control and management interface, while the step 3a and 3b utilizes the data interface to enable a high speed transfer of data samples in both UL and DL directions. Note that the in-band control takes place within step 3c to enable agile RRU reconfiguration. Further, step 4 happens when the association between RRU and BBU changes, it will require to release the RRU and close the connection with the BBU.
- **RoE Layer**: To utilize the off-the-shelf Ethernet approach in terms of underlying protocol and packet format, the design approach here is to make the Ethernet frame to be unaware of the fact that it carries the radio data samples and to use the pre-defined Ethernet frame structure. However, extra sub-header is multiplexed with the radio data samples as the preamble for identification. The proposed encapsulation method and sub-header format are detailed in the following:

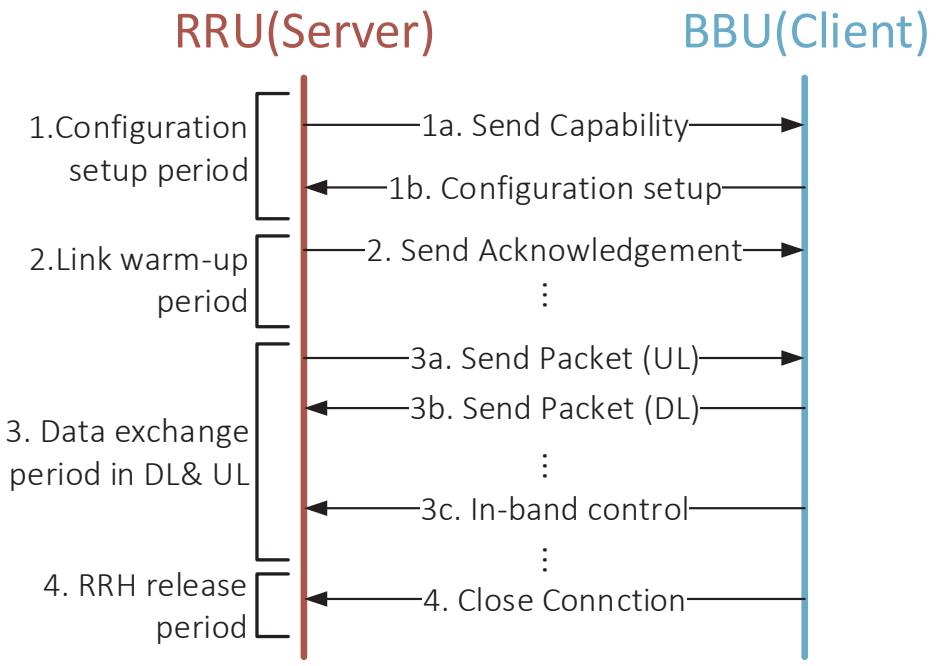


Figure 3.4: Communication session between RRU and BBU

1. **Encapsulation method**: The packet payload is constructed using a split-aware encapsulation method, i.e., the end-points (i.e., BBU/RRU) of a FH interface are aware of the data type to be encapsulated. To this end, both BBU and RRU can properly initiate the sample decapsulation, based on the applied functional split. Further, each data sample is originally with 16-bit in both I/Q domains; however, it can be compressed into 8 bits in both domains using the A-law compression [192].
2. **Sub-header format**: The split-dependent sub-header provides a minimum set of control information, in order to decapsulate the payload. For Splits A and B, the sub-header format is listed in Table 3.2 and Table 3.3, respectively. The size of the payload is configured during the configuration setup period; as a result, there is no need to include this information in the sub-header. Moreover, the subtype in Table 3.3 is used to differentiate packets containing different physical channels within the same data stream, for example, the physical random access channel and other uplink physical channels in the UL direction from RRU to BBU.

- **Transport Layer**: Two approaches are considered and examined: the user datagram protocol (UDP) and RAW Ethernet transportations. The RAW mode utilizes the raw Ethernet that transports packets with minimal processing delay. However, it requires a virtual LAN and spanning tree to support the one-to-many relations between BBU and RRU, as well as the multiplexing in wide area network. On the other hand, UDP-based FH transport protocol offers the ability to accommodate multiple data flows under the same Ethernet interface at the slightly higher processing cost in the protocol stack, which can be mitigated using the zero-copy methods (e.g., Intel dataplane development kit (DPDK) [193] or NetMap framework [194]).

Table 3.2: Sub-header format of Split A

Field	Size (bits)	Descriptions
Timestamp	64	The time when the payload was generated by the RF front-end equipment
Antenna identity	16	Map a packet to the antenna of the RF front-end equipment
Sequence number	16	Indicate the packet sequence number used for reception serialization

Table 3.3: Sub-header format of Split B

Field	Size (bits)	Descriptions
Frame status	32	Include the LTE time indexes (frame, subframe, symbol) and antenna ID
Subtype	16	Identify the packet type: downlink data, uplink data, uplink random access
PRACH configuration	32	Contain the PRACH packet configuration (antenna index, frame number, subframe number)

3.2.3 Design of synchronization

Precise carrier frequency synchronization between all RRUs is required to achieve LTE/5G performance requirements. This is typically achieved by distributing an atomic reference throughout the network. Moreover, the time synchronization among RRUs is also required within one RRU cluster. Such time synchronization can not only allow the joint processing at the user side like co-channel interference suppression, but also facilitate the centralized processing for both UL and DL directions like CoMP at the network side. The distribution of a reference is usually achieved by a timing protocol, such as that used by the CPRI forum with master-slave clock recovery mechanism [167], or more generic protocols such as IEEE 1588 PTP protocol [191] and synchronous Ethernet (SyncE) [195]. An example is depicted in Figure 3.5. The network will assume the so-called grandmaster functionality through a hierarchy of synchronization relays on the paths toward each BBU and synchronization client functionality with frequency distribution unit, through which the reference frequency are distributed toward RRUs. To this end, both timing and frequency references can be transported through the FH interface toward the RRUs in order to regenerate the clock references.

Moreover, the frame timing synchronization can be achieved through the protocol design. Specifically, the timestamp included in the packet sub-header can allow the different transmitted signals from different RRUs are received with a constant time-difference, which is smaller in comparison to the cyclic prefix duration. In this sense, the network MIMO [196] vision can be achieved. Note that the forms of timestamps depend on the applied functional splits, for instance, we use the sampling time of the first sample for split A (cf. Table 3.2), and the indexes of radio frame, subframe and OFDM symbol for Split B (cf. Table 3.3),

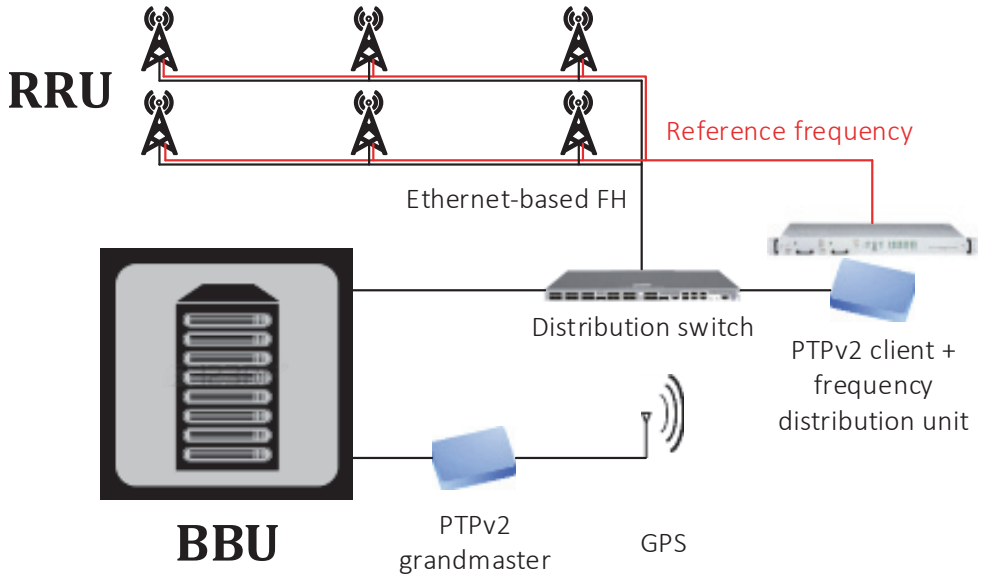


Figure 3.5: Example of FH and synchronization network with addition reference frequency

3.3 C-RAN system key performance index

In this section, we present some important KPIs that will be used to evaluate our proposed framework implementation over the OAI platform. These KPIs will also be used to characterize the performance of different applied techniques in the C-RAN, e.g., data sample compression. Next, we elaborate on these KPIs in three different categories:

3.3.1 Fronthaul KPIs

- *FH interface throughput*: It measures the network throughput in terms of bps over the FH interface between RRU and BBU. This KPI can be used for the FH link capacity provisioning during the C-RAN deployment. Moreover, the required capacity can be reduced under the data sample compression scheme. For instance, the *A-law* compression algorithm can be applied to compress each incoming sample with 16-bit I/Q parts into 8-bit form on both parts. That is to say, the compression ratio is 50%.

- *RTTs of FH and RF front-end*: These two metrics aim to measure the round-trip latency between the FH interface and the RF devices, respectively. Practically, the RTT of FH is defined as the time elapsed at the RRU side from the start of sending the Rx data samples over the FH interface till the end of reading the corresponding Tx data samples from BBU over the FH interface. In detail, this RTT of FH is made up of 5 components: (i) data sample compression time, (ii) FH interface write time, (iii) FH link RTT, (iv) FH interface read time, and (v) data sample decompression time. On the other hand, the RTT of RF front-end is defined as the time elapsed at the RRU side from the start of reading Rx data samples from the RF device till the writing of corresponding Tx data samples to the RF device. One can easily see that the RTT of RF front-end includes the RTT of FH. Last but not least, these two KPIs are important to evaluate the applicable functional splits

and are highlighted in several works. For example, NGMN alliance adopt 250 μs as the maximum one-way fronthaul latency [175], SCF categorizes the one-way FH latency from ideal (250 μs), near ideal (2 ms), sub ideal (6 ms), and non ideal (30 ms) in order to evaluate the applicable functional split [30], and 3GPP identifies different maximum allowed one way latency for each functional split in [32].

3.3.2 Endpoint KPIs

- *RRU/BBU hardware load*: The hardware load at the RRU/BBU comprises the central processing unit (CPU) utilization (the percentage of CPU processing time used by a process out of the total processing time) and the memory utilization (in bytes). Such endpoint KPIs can be utilized for two purposes: (i) Estimate the number of RRUs that can be supported under a limited number of BBU resource in the pool given the FH link condition and the applied functional split, and (ii) Dynamic place the Tx/Rx processing based on the current hardware load to balance loading of all available cloud an edge resources.

3.3.3 User plane KPIs

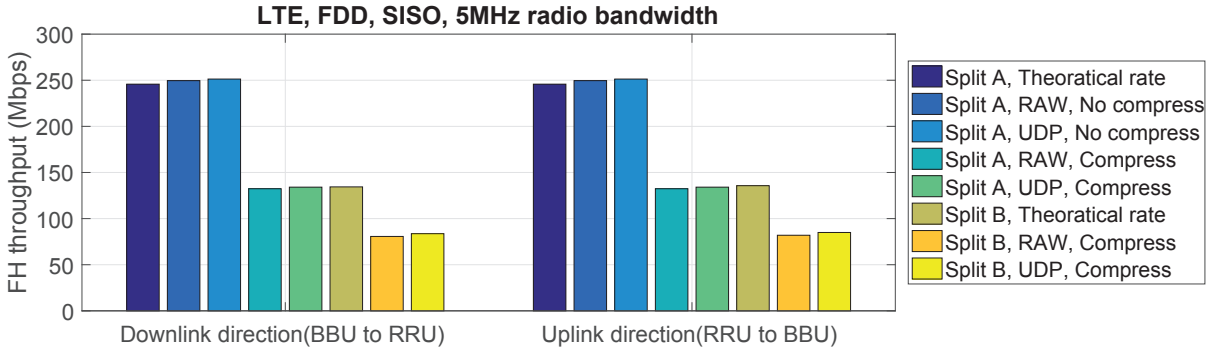
Before introducing the user plane KPIs, we first clarify the relations between user plane and fronthaul KPIs. Take the delay of FH as an example, it can be absorbed and compensated via scheduling the transmission ahead of time, which in turn reduces the total Tx/Rx processing time in order to provide some extra time for the FH transportation. However, this shortened Tx/Rx processing time might not be enough for some compute-intensive processing (e.g., channel decoder) in some cases [158], and thus it will cause the extra user plane latency due to the re-transmission scheme. In following, we elaborate on two considered user plane KPIs:

- *User plane QoS*: To characterize the user plane QoS, we use the *iperf* at both user side and gateway side to measure the good-put, packet drop rate, and delay jitter, both in UL and DL directions. Here, the good-put is the application level successful throughput that is significant for user experience.
- *User plane delay*: For this KPI, we measure the application level RTT over the default radio bearer via exploiting the ping utility at the gateway side. This KPI not only considers the impacts of FH link but also includes the Tx/Rx processing time at both RRU and BBU for both DL and UL directions.

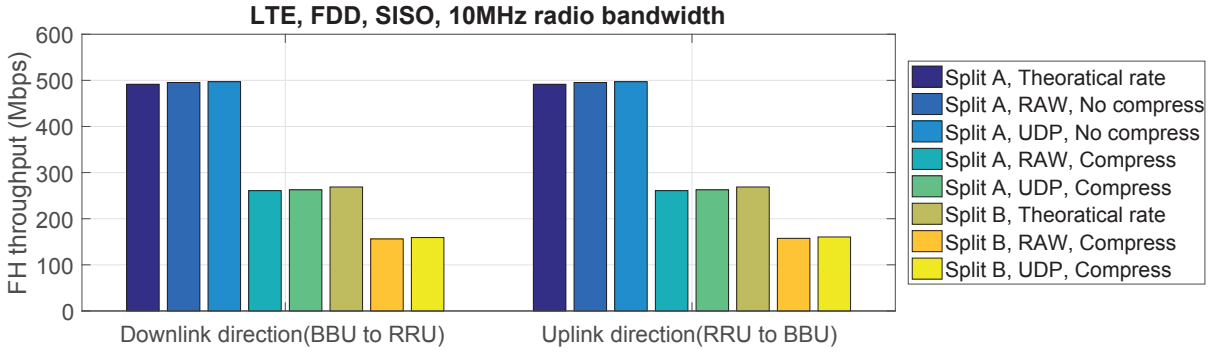
3.4 Implementation results

3.4.1 System setup

In this section, we evaluate the C-RAN implementation over the OAI platform [164], a software-based LTE/LTE-A system implementation spanning the full protocol stack, with the third party evolved packet core, COTS UE (i.e., Samsung Galaxy S6), and USRP B210 software defined radio. Specifically, we consider two network topologies of C-RAN: (a) 1 hop between RRU and BBU, and (b) 2 hops between RRU and BBU. Also, each FH segment is made up of a 3-meter cable wire. For simplicity, only 1 RRU and 1 BBU are in the considered C-RAN network, using either Split A or Split B shown in Figure 2.2, with 5 MHz or 10 MHz radio bandwidth of the LTE FDD mode.



(a) FH throughput of 5 MHz radio bandwidth



(b) FH throughput of 10 MHz radio bandwidth

Figure 3.6: FH throughput of 5 MHz and 10 MHz radio bandwidth

3.4.2 Fronthaul KPIs

The FH interface throughput as well as the theoretical data rate of both 5 MHz and 10 MHz cases are shown in Figure 3.6a and Figure 3.6b, respectively. Since these two considered functional splits will transport a constant traffic through the FH interface; therefore, these results are irrelevant to the user plane traffic. First of all, the RAW transportation mode only has little overhead (between 3 to 4 Mbps) compared with the theoretical data rate analysis. The UDP transportation will include some extra overhead (less than 3Mbps) compared with the RAW mode. Moreover, the applied *A-law* compression scheme can reach almost 50% reduction in the FH throughput as expected. Furthermore, using Split B shows the gain of 43.8% in terms of FH throughput reduction, when compared with Split A, via moving DFT/IDFT operation to the RRU. This reduction ratio is close to the analysis outcome that shows 45.3% of FH throughput reduction.

Further, we elaborate on how the RTT of FH and RTT of RF front-end are measured in Figure 3.7. First of all, the RF front-end collects all I/Q data samples within one TTI at the time instance $T_{trx-read}$. After finishing the UL processing at the RRU side (according to the applied functional split), the data samples are ready to be transported through the FH interface at the time instance $T_{if-send}$. Then, we can see that the RTT of FH is from $T_{if-send}$ to $T_{if-recv}$, in which the aforementioned five components in Section 3.3.1 are contained, as shown in Eq. (3.1a). Finally, after the DL processing at the RRU side, the RF device will transmit all samples at

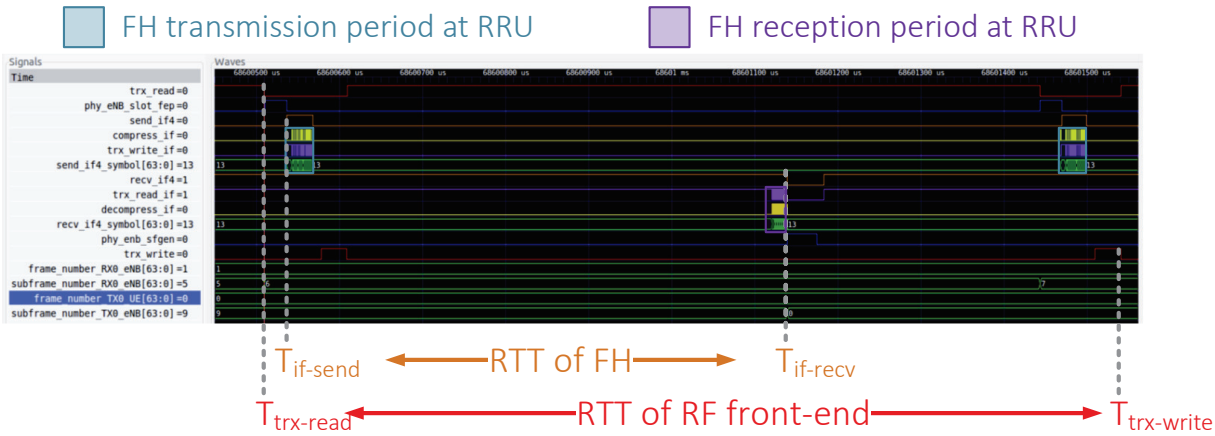
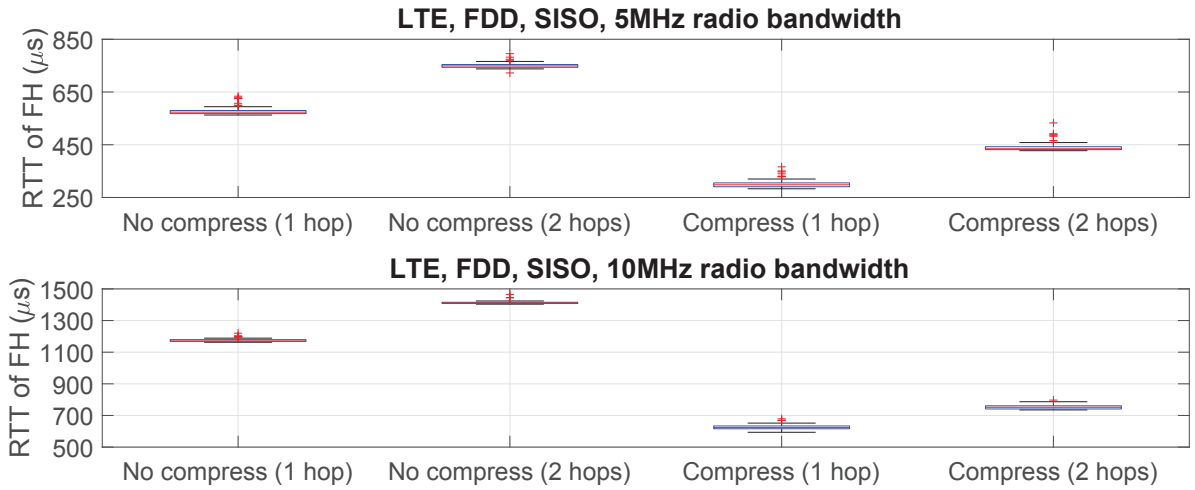


Figure 3.7: Example of RTT of FH and RTT of RF front-end

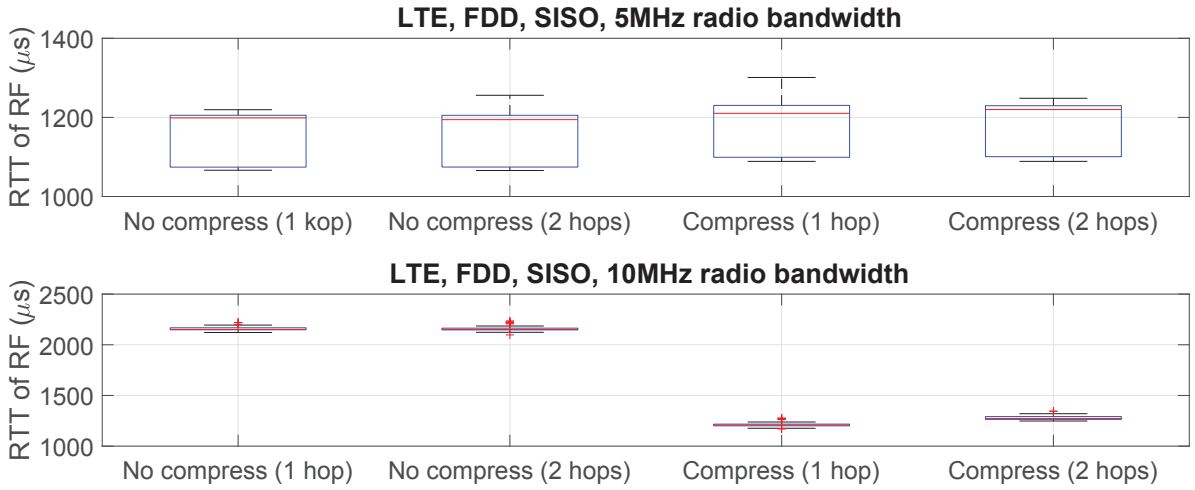
the time instance $T_{trx-write}$. As shown in Eq. (3.1b), the RTT of RF front-end comprises (1) the RTT of FH, (2) the UL and DL baseband and protocol processing time at RRU side, and (3) the data sample read/write time from/to the RF device, and it can complete the Rx/Tx transmission of a single TTI.

$$\begin{aligned}\text{RTT of FH} &= T_{if-recv} - T_{if-send} \\ &= \text{Data sample compression time} + \text{FH interface write time} + \text{FH link RTT} \\ &\quad + \text{FH interface read time} + \text{Data sample decompression time} \qquad (3.1a) \\ \text{RTT of RF} &= T_{trx-write} - T_{trx-read} \\ &= \text{RF device read time} + \text{UL processing time at RRU} + \text{RFF of FH} \\ &\quad + \text{DL processing time at RRU} + \text{RF device write time} \qquad (3.1b)\end{aligned}$$

Further, the measured RTTs of FH and RF front-end are shown in Figure 3.8a and 3.8b, respectively in boxplot. We can observe that the compression scheme can reduce the RTT of FH via interplaying the extra time taken for the data sample compression/decompression and the reduced FH interface reading/writing time, as summarized in Table 3.4. In practice, the reduction of FH interface reading/writing time is comparatively larger than the extra time taken for the compression and decomporession operations as shown in Table 3.4, which confirms the benefit of the compression scheme in the FH network. Moreover, these results also reveal that all above considered deployment scenarios and Ethernet transportation can fulfill the most stringent one-way delay in terms of *250*μs made by the NGMN alliance [175], SCF [30] and 3GPP [32] and can support all physical layer functional splits, via subtracting the FH interface reading and writing time, the data sample compression and decompression time from one-half of the RTT of FH. Additionally, it can be seen from Figure 3.8b that the average RTT of RF front-end for 5 MHz case is close among different scenarios to finish the complete Rx/Tx transmission of a single TTI. However, we can see an extra 1 ms for the 10 MHz case when the compression scheme is not applied The reason behind is due to the fact that the RTT of FH for this scenarios is already larger than the duration of 1 TTI (1ms) as shown in Figure 3.8a; hence, another 1 ms is needed to complete the Rx/Tx transmission of a single TTI.



(a) RTT of FH for 5 MHz and 10 MHz radio bandwidth



(b) RTT of RF front-end for 5 MHz and 10 MHz radio bandwidth

Figure 3.8: RTT of FH and RTT of RF front-end

Table 3.4: Processing time for FH interface reading/writing and data sample compression/decompression

Radio bandwidth	Compression	Data sample compression time (μs)	FH interface write time (μs)	FH interface read time (μs)	Data sample decompression time (μs)
5 MHz	No	-	69.19	224.37	-
	Yes	16.53	68.80	53.13	23.43
10 MHz	No	-	72.35	469.53	-
	Yes	31.79	71.95	170.13	35.93

Table 3.5: Endpoint KPIs in terms of CPU ratio and memory usage

Radio bandwidth	Functional split	Endpoint	CPU Ratio (%)	Memory usage (KByte)
5MHz	eNB		40.15%	1002019
	A	RRU	16.76%	917486
		BBU	26.57%	918794
	B	RRU	24.19%	917478
		BBU	22.71%	917174
10MHz	eNB		65.02%	1195059
	A	RRU	29.23%	1107382
		BBU	45.70%	1180126
	B	RRU	41.12%	1107374
		BBU	32.40%	1124989

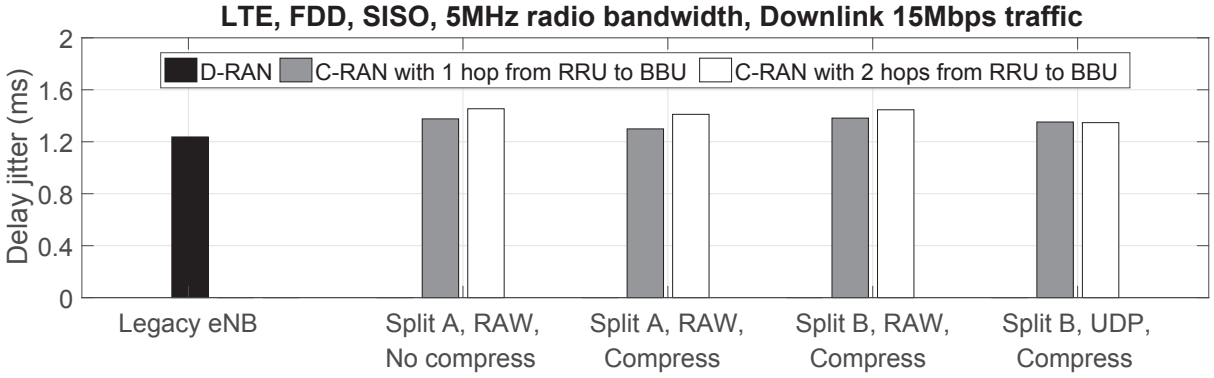
3.4.3 Endpoint KPIs

As for the endpoint KPI, we compare the C-RAN deployment (RRU, BBU) with the legacy D-RAN deployment (evolved Node B [eNB] as BS in LTE) based on the OAI platform. The results are listed in Table 3.5 where the CPU utilization ratio is the percentage of CPU processing time of the process and the memory usage is measured based on the proportional set size in KByte. Since the endpoint KPI depends on the air-interface traffic, we here use the traffic generated by the iperf tool: 15 Mbps/30 Mbps in the DL direction and 5 Mbps/10 Mbps in the UL direction for 5 MHz/10 MHz radio bandwidth respectively.

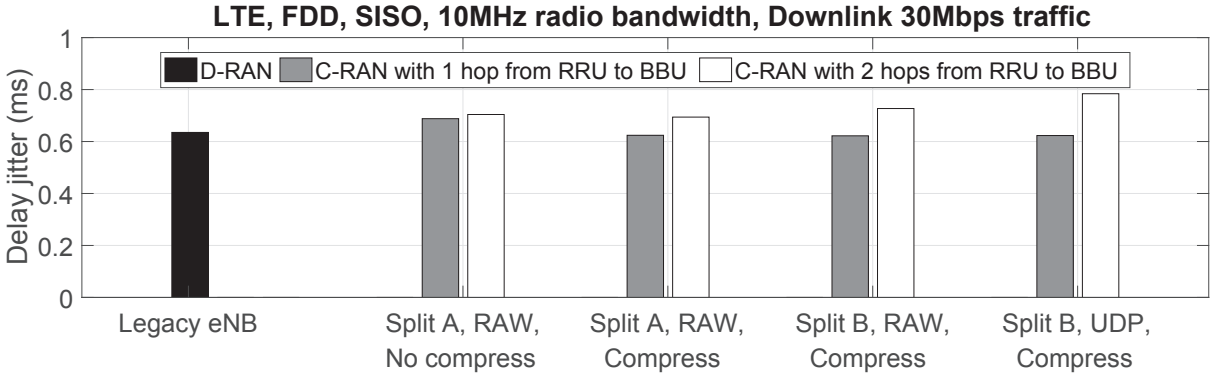
The RRU, BBU and eNB are deployed in a 6-core machine each with Intel i7 Sandy Bridge architecture in 3.2 GHz processor frequency. As a result, we can see that 2 CPU cores are required to deploy the proposed RRU architecture using 10 MHz radio bandwidth with Split A, and 3 CPU cores are required for the case with Split B[1]. Moreover, we can observe that the overall required CPU cores by RRU and BBU is slightly higher than the required ones by the legacy monolithic eNB. Such extra number of CPU cores for C-RAN deployment is utilized for FH transportation, data sample (de-)compression and the processing overhead at both RRU and BBU. Furthermore, when comparing the results of 5 MHz and 10 MHz, a close amount of the increased CPU cores is observed for both D-RAN and C-RAN deployments.

In addition, the memory usages at RRU and BBU do not have large differences since here we apply the aforementioned *flexible-split deployment* in Section 3.2.1, i.e., all baseband functionalities are still required at both RRU and BBU sides to support a full flexible function split change between RRU and BBU. In contrast, if we only deploy necessary functions according to the applied functional split, i.e. split-specific deployment, the memory usage can be largely reduced, for instance, to 16 KBytes for RRU when using Split A. In this case, only Split A can be applied and thus no flexibility can be exploited.

[1]Note that the number of required CPU cores can be reduced when using new series of Intel processor, e.g., Skylake, Kaby Lake, Coffee Lake.



(a) Packet delay jitter for 5 MHz radio bandwidth



(b) Packet delay jitter for 10 MHz radio bandwidth

Figure 3.9: Packet delay jitter for 5 MHz and 10 MHz radio bandwidth

3.4.4 User plane KPIs

In following, we compare the C-RAN deployment with the legacy D-RAN in terms of the user plane KPIs by transferring 15 Mbps and 30 Mbps traffic for 5 MHz and 10 MHz radio bandwidth in the DL direction, separately. First, we compare the packet delay jitter in Figure 3.9a and Figure 3.9b. Due to the extra hop from the user to the gateway that leads to less available Tx/Rx processing time as explained before, the delay jitter of C-RAN deployment is larger than the one of legacy D-RAN. Similarly, the delay jitter will be further enlarged if there are more hops from RRU to BBU, i.e., 2 hops. In comparison, few differences are observed when applying the different functional splits, different transport protocols, or compression scheme.

Then, we present the packet drop rate in Table 3.6, in which a close performance is seen between the C-RAN and D-RAN deployments except in the case when applying the compression scheme for Split A. This degradation may due to the large peak-to-average power ratio (PAPR) characteristic of time-domain data samples, i.e., some data samples can have very high amplitudes. Nevertheless, these high-amplitude data samples will be largely distorted by the applied compression scheme (detailed in Section 3.5) and this large distortion will affect all other subcarriers due to the spread operation after applying the DFT. In that sense, few more packets will be dropped in the network protocol stack (e.g., MAC and RLC layers) and the re-transmission scheme will be triggered.

Table 3.6: Packet drop rate for 5 MHz and 10 MHz radio bandwidth

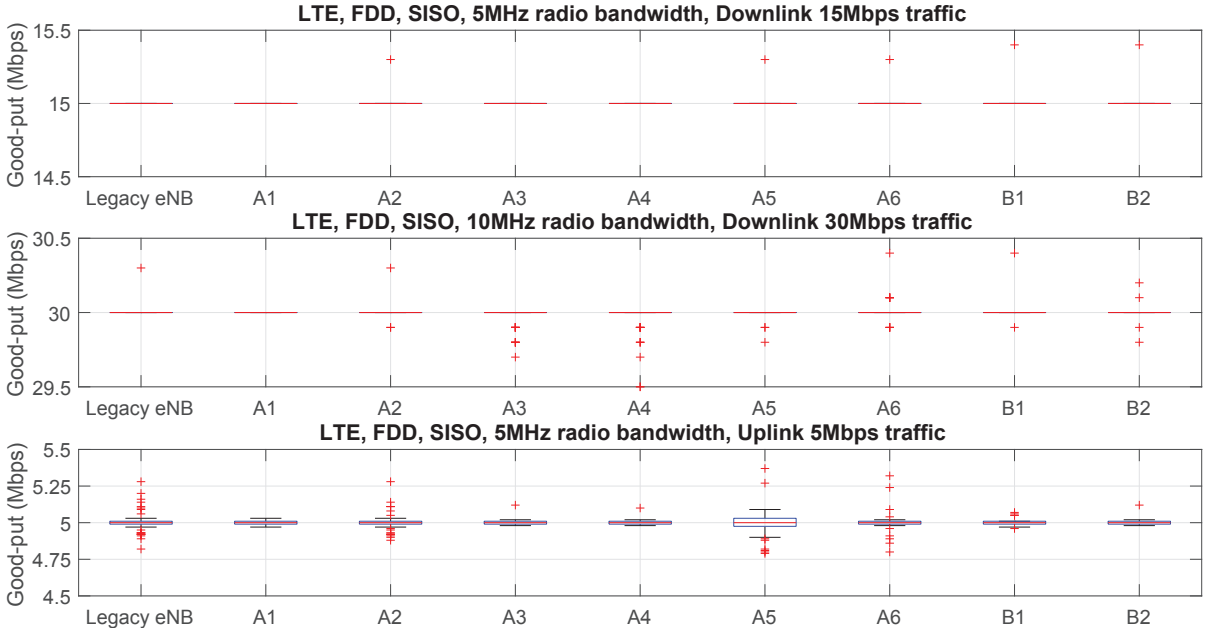
Deployment	Functional split	Protocol	Compression	Packet drop rate (%) of different radio bandwidth	
				5 MHz	10 MHz
D-RAN				0.002%	0.013%
C-RAN	A	RAW	No	0.001%	0.008%
		RAW	Yes	0.009%	0.040%
		UDP	No	0.001%	0.018%
		UDP	Yes	0.010%	0.023%
	B	RAW	Yes	0.001%	0.017%
		UDP	Yes	0.003%	0.014%

Further, the measured user good-put at the application level is shown in Figure 3.10 comparing eight specific C-RAN deployment scenarios (A1 to A6, B1, B2 in the table below Figure 3.10) and legacy D-RAN. In the UL direction, we transport a 5 Mbps user throughput when using 5 MHz radio bandwidth. We can observe that these different C-RAN deployment scenarios show almost the same good-put variation as the ones of the D-RAN deployment; that is to say, there is no observable difference in the experienced good-put among C-RAN or D-RAN deployments. Moreover, when comparing the results of A1 and A2 modes, we can see that the compression scheme will not decrease the experienced user good-put even with a slight higher packet drop rate in Table 3.6. Additionally, the packet delay jitter introduced by the extra hop (cf. Figure 3.9) will neither impact the good-put, when viewing the results from A3 to B2 modes. In summary, this similar good-put performance can justify the claim that either C-RAN or D-RAN deployments can be transparent in terms of providing the similar user QoE.

Finally, we measure the RTT between the user and the gateway in Figure 3.11 via leveraging the *ping* utility with 8192 bytes packet size and 0.2 second inter-packet departure time. We can first observe that the average RTT of all C-RAN deployments is close to the one of the legacy D-RAN deployment. Moreover, the user plane RTT distribution is similar between D-RAN and C-RAN when there is only 1 hop between RRU and BBU (i.e., A1 and A2 modes). Such results not only confirm again that the C-RAN deployment can be transparent to the user QoE, but also implies that the compression will not impact the user plane RTT. Further, among the cases with 2 hops between RRU and BBU (i.e., A3 to A6, B1, B2), a long-tail distribution is exhibited. This phenomenon is due to the extra hops from the user to the gateway, which will reduce the time for Tx/Rx processing as explained beforehand.

3.4.5 Summary

To sum up, the packet-based C-RAN architecture is realized through the OAI platform and several different C-RAN deployment scenarios are exploited. As we can observe, a number of advantages are shown via using the compression scheme, i.e., less FH throughput, shorter RTT of FH and RF front-end; however, it potentially increase the packet drop rate. Moreover, the user plane KPIs reveal two potential performance degradation causes: (i) Reduced Tx/Rx processing time due to FH transportation, and (ii) Packet loss in the FH and the packet drop due to the compression scheme.



Mode	Functional Split	Protocol	Compression	Hop count
A1	A	RAW	No	1
A2			Yes	
A3			No	2
A4		UDP		
A5		RAW	Yes	
A6		UDP		
B1	B	RAW		
B2		UDP		

Figure 3.10: Measured good-put of several C-RAN and D-RAN deployment scenarios

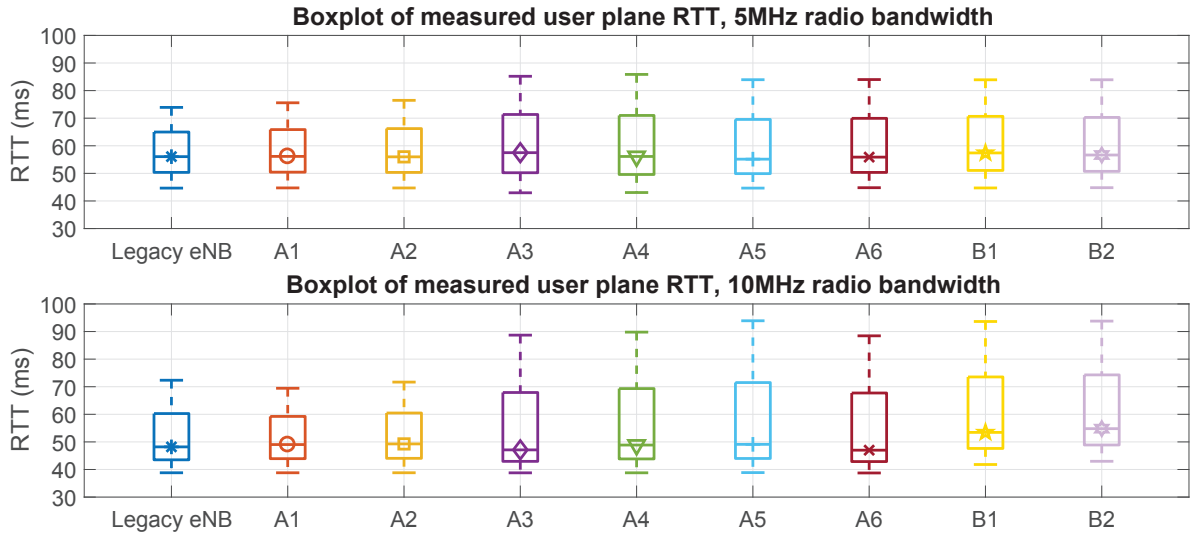


Figure 3.11: Downlink user plane packet RTT for 5 MHz and 10 MHz radio bandwidth

Table 3.7: Original A-law compression scheme

Input (16 bits)		Output (8 bits)			Expansion error	
From	To	Sign (1 bit)	Exponent (3 bits)	Mantissa (4 bits)	Minimum	Maximum
-2^{15}	$-2^{14}-1$	1	7	From 0 to 15	-511	512
-2^{14}	$-2^{13}-1$	1	6	From 0 to 15	-255	256
-2^{13}	$-2^{12}-1$	1	5	From 0 to 15	-127	128
-2^{12}	$-2^{11}-1$	1	4	From 0 to 15	-63	64
-2^{11}	$-2^{10}-1$	1	3	From 0 to 15	-31	32
-2^{10}	$-2^{9}-1$	1	2	From 0 to 15	-15	16
-2^{9}	$-2^{8}-1$	1	1	From 0 to 15	-7	8
-2^{8}	-1	1	0	From 0 to 15	-7	8
0	$2^{8}-1$	0	0	From 0 to 15	-7	8
2^{8}	$2^{9}-1$	0	1	From 0 to 15	-7	8
2^{9}	$2^{10}-1$	0	2	From 0 to 15	-15	16
2^{10}	$2^{11}-1$	0	3	From 0 to 15	-31	32
2^{11}	$2^{12}-1$	0	4	From 0 to 15	-63	64
2^{12}	$2^{13}-1$	0	5	From 0 to 15	-127	128
2^{13}	$2^{14}-1$	0	6	From 0 to 15	-255	256
2^{14}	$2^{15}-1$	0	7	From 0 to 15	-511	512

3.5 Further explorations on data sample compression

Within the above work, we apply the well-known A-law compression scheme that was originally provided by ITU-T in [192]. Such A-law compression approach will compress the incoming 16-bit sample into an 8-bit output (1 sign bit, 3 exponent bits and 3 mantissa bits), and then expand it back to the original 16-bit level. However, it will introduce severe expansion error when the input signal amplitude is large as summarized in Table 3.7. For instance, when the input signal is either from -2^{15} to $-2^{14}-1$ or from 2^{14} to $2^{15}-1$, the largest expansion error will generate $\frac{512}{2^{15}} = 1.56\%$ signal distortion. Even this large input signal rarely happens in the average case; however, the distorted value will impact all other sub-tones significantly due to the spreading effect introduced by the DFT/IDFT operations. Hence, all symbols of the same time index will be influenced by this large expansion error. What is worst is that such high-amplitude inputs will show up more frequently due to the aforementioned high PAPR characteristic for OFDM To this end, we investigate some possible approaches that can be applied for the improvement, while still maintaining the aforementioned benefits brought by the compression scheme, e.g., FH throughput reduction and few compression/decompression time.

The first applicable approach is to dynamically quantize the incoming data samples when receiving a chunk of input samples, e.g., each OFDM symbol or each time slot, based on the provisioned FH link capacity. That is to say, we can replace the (de-)compression unit within the proposed RRU/BBU architecture with the FH (de-)quantization unit for this purpose. This approach will further include the quantization control information to be packed together with the sub-header to facilitate the data sample recovery. Nevertheless, the optimal quantization approach may take too much time to find the best solution, and it might take even longer time than the data sample (de-)compression time shown in the Table 3.4. To this end, we only consider the use of a uniform linear quantizer that can be finished in a much shorter time period.

Table 3.8: Scheme of the first candidate to improve original A-law compression

Input (16 bits)		Output (8 bits)			Expansion error	
From	To	Sign (1 bit)	Exponent (2 bits)	Mantissa (5 bits)	Minimum	Maximum
-2^{15}	$-2^{14}-1$	1	3	From 0 to 31	-255	256
-2^{14}	$-2^{13}-1$	1	2	From 0 to 31	-127	128
-2^{13}	$-2^{12}-1$	1	1	From 0 to 31	-63	64
-2^{12}	-1	1	0	From 0 to 31	-63	64
0	$2^{12}-1$	0	0	From 0 to 31	-63	64
2^{12}	$2^{13}-1$	0	1	From 0 to 31	-63	64
2^{13}	$2^{14}-1$	0	2	From 0 to 31	-127	128
2^{14}	$2^{15}-1$	0	3	From 0 to 31	-255	256

As the second approach, we can improve the original A-law compression via adding the extra bits for the mantissa part, while reducing the number of bits used for the exponent part. The reason behind is that the 4-bit mantissa part in Table 3.7 is not enough for the high-amplitude inputs, which will generate significant distortion to all sub-tones. Note that reducing the exponent part may also decrease the resolution especially when the input signal is small. Hence, our aim here is to do the trade-off between the signal distortion between high- and low-amplitude inputs and we propose two possible candidates as summarized in Table 3.8 and Table 3.9 respectively. The first candidate directly uses one more bit for the mantissa part from the original exponent part in order to decrease the expansion error for a wide range of high-amplitude inputs (i.e., from -2^{15} to $-2^{12}-1$ or from 2^{12} to $2^{15}-1$); however, the low-amplitude inputs will suffer significantly. On the other hand, the second candidate strikes a balance in this trade-off via only reducing the expansion error for a smaller but critical region (i.e., from -2^{15} to $-2^{13}-1$ or from 2^{13} to $2^{15}-1$) and giving fewer side-effects to the low-amplitude inputs. A visual comparison of these two candidates with the original scheme in terms of the output expansion error is shown in Figure 3.12. We can observe that both candidates can reduce the largest error as $\frac{256}{2^{15}} = 0.78\%$, while the second candidate has a two-times smaller error when the input signal is from -2^{11} to $2^{11}-1$. Finally, one may be curious to know whether we can continually increase the mantissa part by 1 bit to further reduce the largest error for the peak input signal. Nevertheless, adding one more bits for the mantissa part will lead to a constant expansion level between $[-127, 128]$ for all inputs, and thus it will become a linear uniform quantizer, which will be examined in the first approach.

In the following, we show the performance of these two approaches over two extreme high-order modulation cases: (1) the DL direction using Split A with 1024QAM, and (2) the UL direction using Split B with 256QAM. These high-order modulations make the expansion error become a performance dominating factor. In Figure 3.13, we can see that the original 16-bit A-law compression scheme has a large gap toward the all floating point receiver, i.e., using 16 bits without any compression. Specifically, the gap is approximately 2.9 dB and 1.2 dB when we see the level with 10^{-4} uncoded bit error rate. If we apply the first approach, i.e., using an 8-bit linear uniform quantizer as the replacement, it can reduce the gap by 1.1 dB and 0.4 dB, respectively for two cases. Further, if we examine the second approach via modifying the original A-law compression scheme, both two candidates can show significant performance enhancement. Specifically, when compared with the original A-law compression scheme, the first candidate can

Table 3.9: Scheme of the second candidate to improve original A-law compression

Input (16 bits)		Output (8 bits)			Expansion error	
From	To	Sign (1 bit)	Exponent (2 bits)	Mantissa (5 bits)	Minimum	Maximum
-2^{15}	$-2^{14}-1$	1	3	From 0 to 31	-255	256
-2^{14}	$-2^{13}-1$	1	2	From 0 to 31	-127	128
-2^{13}	$-2^{12}-1$	1	1	From 16 to 31	-127	128
-2^{12}	$-2^{11}-1$	1	1	From 0 to 15	-63	64
-2^{11}	$-2^{10}-1$	1	0	From 16 to 31	-63	64
-2^{10}	-1	1	0	From 0 to 15	-31	32
0	$2^{10}-1$	0	0	From 0 to 15	-31	32
2^{10}	$2^{11}-1$	0	0	From 16 to 31	-63	64
2^{11}	$2^{12}-1$	0	1	From 0 to 15	-63	64
2^{12}	$2^{13}-1$	0	1	From 16 to 31	-127	128
2^{13}	$2^{14}-1$	0	2	From 0 to 31	-127	128
2^{14}	$2^{15}-1$	0	3	From 0 to 31	-255	256

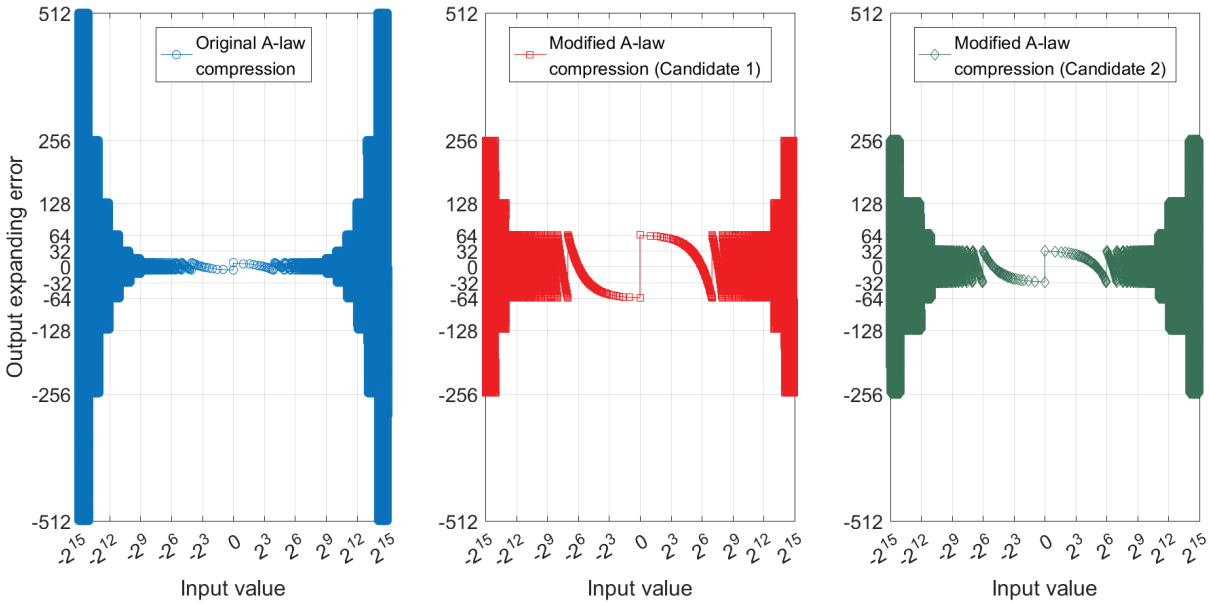
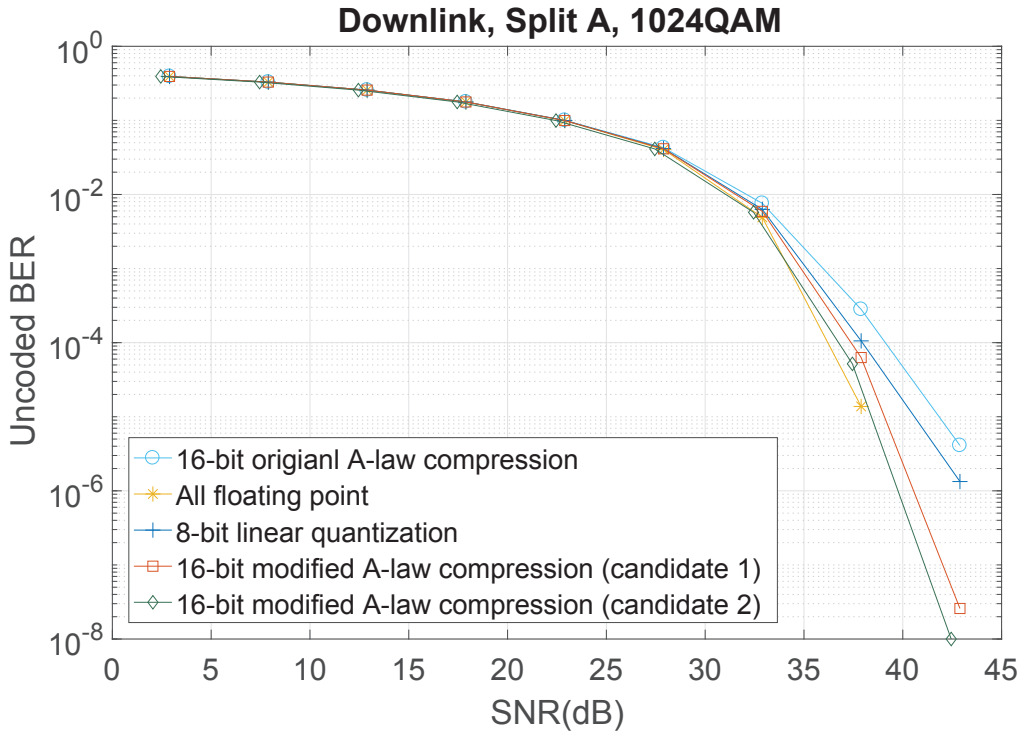
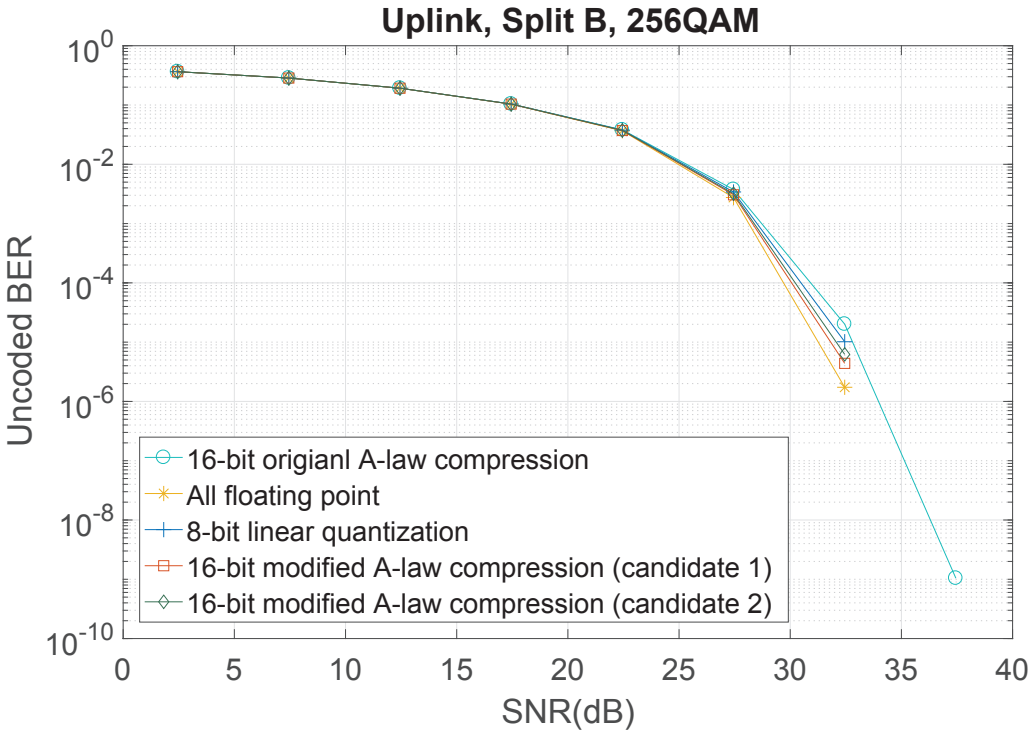


Figure 3.12: Output distortion comparisons between original and two modified A-law scheme

bring 1.7 dB and 0.8 dB gain, and the second candidate will generate 2.3 dB and 0.7 dB gain, respectively. Such gains not only justify the claim that the performance bottleneck of the original A-law compression scheme is at the high-amplitude inputs, but also indicate a better performance can be achieved via properly trade-off between high- and low-amplitude expansion errors. To conclude, these possible approaches can replace the original A-law compression scheme and still maintain the aforementioned compression benefits mentioned in Section 3.4.



(a) Donwlink direction using 1024QAM with Split A



(b) Uplink direction using 256QAM with Split B

Figure 3.13: Performance comparison of different A-law compression scheme enhancements

3.6 Discussions

In the aforementioned study, the architecture of flexible RRU/BBU architecture for the flexible functional split over the Ethernet-based FH is implemented and investigated in several different aspects. As the next step, we list three potentials directions as the following.

First of all, one direction is to investigate the impacts of functional split reconfiguration on the user performance, i.e., UP KPI investigated in this chapter. Also, such reconfiguration scheme shall be designed to generate as few impacts as possible toward the overall C-RAN deployment, i.e., non-destructive reconfiguration. Finally, some other radio resource management operations can be applied, such as handover and reserved random access, to mitigate the influences from split reconfiguration.

Second, the synchronization issue of C-RAN shall be dealt in a large-scale deployment to fulfill the stringent synchronization requirements. For instance, 3GPP defines the synchronization requirements as 65 ns, 130 ns and 260 ns in [197] for MIMO transmissions or transmission diversity, intra-band contiguous carrier aggregation and inter-band carrier aggregation, respectively[2]. Such stringent requirements need a thorough investigation on the delay contributions from the FH transportation.

Last nut not least, the compression scheme is a critical topic that needs further examinations due to the following trade-off. On one hand, the real-time compression scheme is desired to reduce its impacts on other C-RAN deployment issues like synchronization. On the other hand, the (sub-)optimal compression scheme will require more time complexity for a better compression ratio. To this end, different compression schemes can be deployed according to the FH condition, delay budget, and service requirements.

3.7 Conclusions

To conclude, in this chapter we propose a unified RRU/BBU framework that supports the envisioned flexible C-RAN with the capability of flexible functional split and Ethernet-based FH transportation. Such flexible C-RAN acts as as the natural evolution from the split-enabled C-RAN studied in Chapter 2. Moreover, three categories of KPI are identified to evaluate different C-RAN deployment scenarios. Based on the implementation over the OAI platform, the C-RAN concept is proved to be applicable and it shows the compatible user experience with the legacy D-RAN. Finally, we examine some possible enhancements on the considered compression scheme to further extend the C-RAN benefits.

[2]There are some other less stringent requirements, such as 500ns to 1.5 μs for CoMP and 1.5 μs for LTE time-domain division mode.

Chapter 4

Flexible C-RAN Centralization of End-to-end RAN Service

4.1 Introduction

To further exploit the benefits of the flexible C-RAN vision, we aim to investigate the impacts of different levels of C-RAN centralization on providing an E2E RAN service. Notice that this E2E RAN service spans from the centralized BBU pool[1], a multi-segment FH network between RRUs and BBUs, several distributed RRUs toward a number of served end-users. Nevertheless, several important design issues shall be addressed properly to deliver the E2E RAN service, elaborated in the following paragraphs.

The first encountered issue is to allocate the suitable *functional split* between RRU and BBU, as explained in the previous two chapters, to strike a balance between the centralization benefit and the requirements of FH network via re-distributing the network functions. Another issue that can also remedy the FH network requirements is to pool the processing for different groups of RRUs at several geographically distributed locations. Hence, all RRUs can be partitioned into different RRU clusters [49], and all data samples of RRUs within the same RRU cluster are transported to/from the same BBU, termed as the anchor BBU, where the centralized and coordinated processing takes place. One important point is that RRUs within the same cluster can be jointly processed to enable the cooperation mechanism, while RRUs belonging to different clusters can only cooperate opportunistically in a larger time-scale. Nevertheless, a trade-off can be observed when applying such *RRU clustering* technique between the number of RRUs in a cluster and the FH network requirements toward the anchor BBU.

Based on the established RRU clusters and the applied functional splits, different levels of centralization can be observed in the C-RAN deployment. Moreover, to exploit the benefits among different levels of centralization, the *user association* design issue is of high importance to improve the overall network spectral efficiency via associating each user to the serving RRU cluster. One can notice that each user is now associated with a number of RRUs, in which the coordination processing of that user can naturally happens at the anchor BBU according to the applied functional split. Finally, to guarantee the proper functionalities of the formed E2E RAN service, extra considerations on the *FH network routing* and *BBU anchoring* shall be made. More specifically, the FH requirements in terms of capacity and latency shall be satisfied

[1]Further decomposition into CU and DU is out of the scope of this chapter.

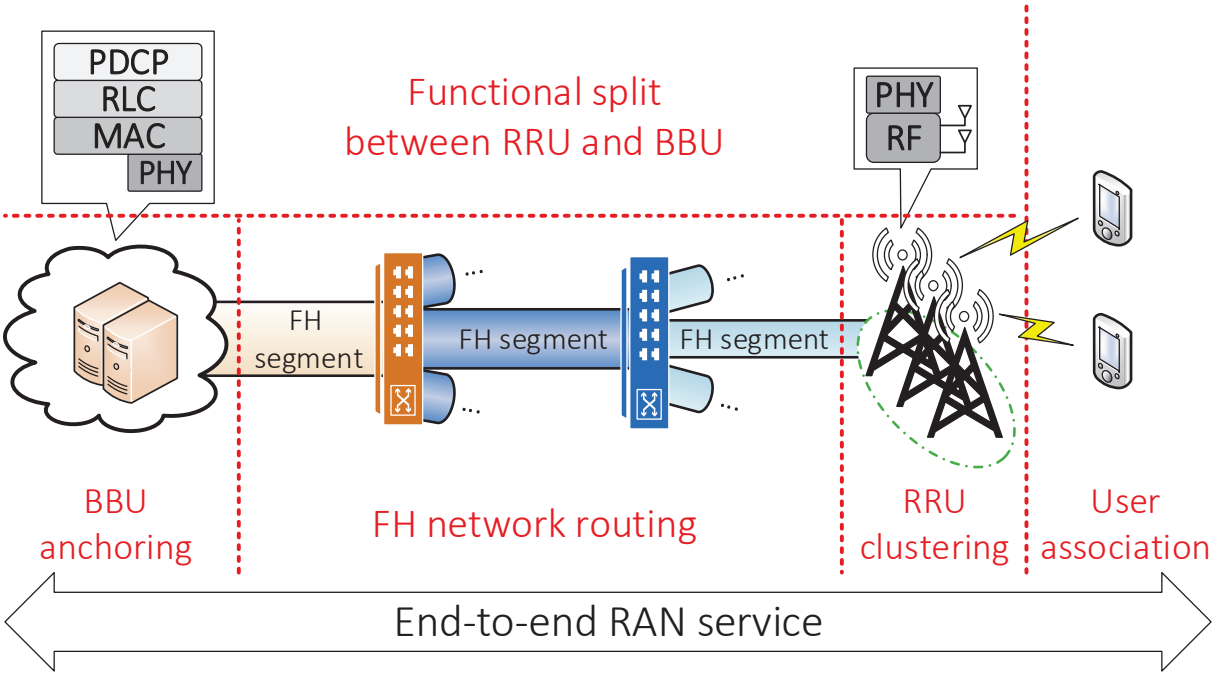


Figure 4.1: Five design issues of C-RAN to deliver the end-to-end RAN service

to successfully route the FH traffic between each RRU toward the anchor BBU. As explained in Chapter 2, the FH throughput will be dependent on the applied functional split as well as the user traffic characteristic.

According to the aforementioned outline, there are five crucial design issues to deliver the E2E RAN service: (a) Functional split, (b) RRU clustering, (c) user association, (d) BBU anchoring, and (e) FH network routing. A visual explanation of how these five issues can compose the E2E RAN service is shown in Figure 4.1, and we can notice that these issues are relevant and will impact on each other. For instance, forming a larger RRU cluster can exploit more centralization benefits to the associated users, but it will also introduce a higher FH throughput requirement toward the centralized anchor BBU. To remedy this higher FH throughput requirements, we can apply different functional splits to shift some network functions from the centralized anchor BBU to the distributed RRUs to reduce the FH requirements, at a cost of reducing the centralization benefit. In that sense, all these design issues shall be considered at the mean time. Take the Figure 4.2 as an network topology example, each RRU cluster can be formed with the respective functional split, anchor BBU, and associated users. Moreover, RRUs within the same cluster can deliver the FH traffic through different routing paths toward the same anchor BBU.

In this chapter, we first introduce our considered system model and formulate the optimization problem to maximize the overall network spectral efficiency. However, since the formulated problem is proved to be NP-hard, we then revisit the problem and propose a practical and systematic solution. Finally, we will provide the extensive simulation results to justify our proposed solution.

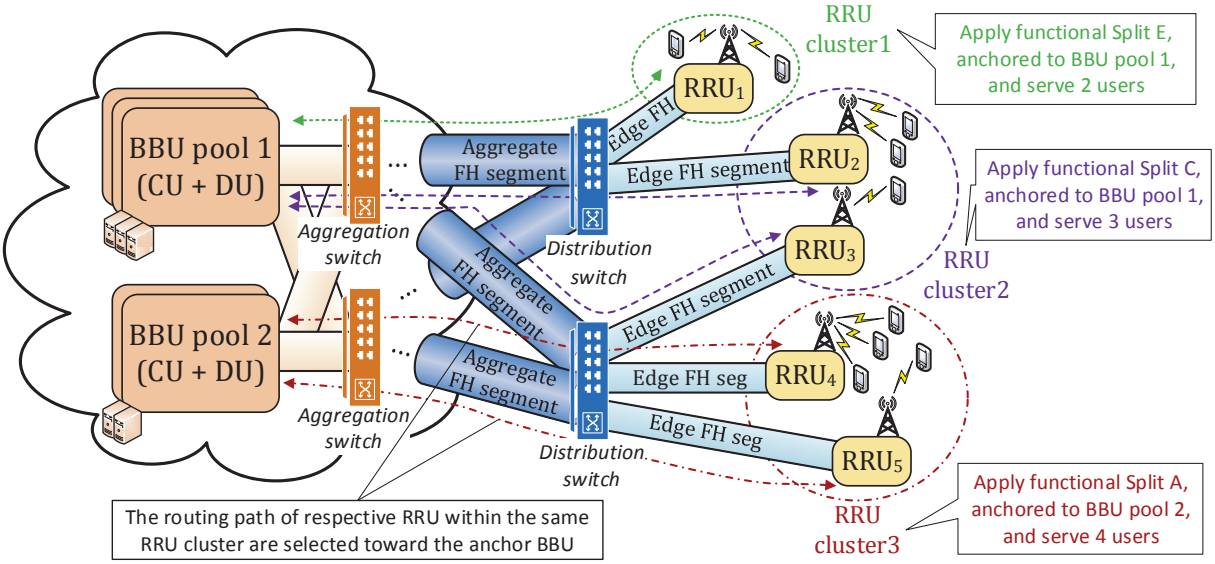


Figure 4.2: Network topology of C-RAN to support the E2E RAN service

4.2 System model

First of all, we give a whole view on the considered C-RAN system model. In Table 4.1, we summarize some important parameter notations that are used through this chapter[2].

4.2.1 Network topology

In the considered network topology, there are $|\mathcal{R}|$ RRUs $\mathcal{R} \coloneqq \{r_1, \cdots, r_{|\mathcal{R}|}\}$, each with $M > 1$ antennas, that are responsible to serve all $|\mathcal{U}|$ single-antenna users $\mathcal{U} \coloneqq \{u_1, \cdots, u_{|\mathcal{U}|}\}$. Further, the $M \times 1$ frequency domain fading channel vector from the k-th user to the i-th RRU at time t is denoted as $\mathbf{h}_{r_i,u_j}(t) \sim CN\left(\mathbb{0}_{M\times 1}, \sigma^2_{r_i,u_j} \cdot \mathbf{I}_M\right)$, where $CN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate complex Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. For simplicity, we drop the time index t in the rest parts of this chapter. Then, we define the anchor RRU for the j-th user as r_{u_j} in Eq. (4.1a) and the associable RRU set of the j-th user as R_{u_j} in Eq. (4.1b), in which γ_{th} is a pre-defined minimum user associable threshold in terms of the signal power gain. The corresponding $|\mathcal{R}| \times |\mathcal{U}|$ indicator matrix is formed as $\mathbf{Q}$, in which its (i,j)-th element $q_{i,j}$ is 1 if $r_i \in R_{u_j}$ and is 0 otherwise.

$$r_{u_j} = \underset{r_i \in \mathcal{R}}{\operatorname{argmax}}\ \sigma^2_{r_i,u_j} \tag{4.1a}$$

$$R_{u_j} = \left\{r_{u_j}\right\} \cup \left\{r_i \in \mathcal{R} : \sigma^2_{r_i,u_j} \geqslant \gamma_{th}\right\} \tag{4.1b}$$

Afterwards, all $|R|$ RRUs are connected to a number of centralized BBUs $\mathcal{B} \coloneqq \{b_1, \cdots, b_{|\mathcal{B}|}\}$ through the packet-based Ethernet FH network that consists of packet-switching forwarding

[2]Additionally, bold uppercase letters denote matrices; bold lowercase letters denote column vectors; $(\cdot)^T$ and $(\cdot)^H$ are transposition and Hermitian transposition operators respectively; $\|\cdot\|_p$ is the p-norm of the column vector; $\mathbb{1}_{M\times N}$ and $\mathbb{0}_{M\times N}$ are the $M \times N$ all-ones and all-zeros matrix respectively; $\mathbf{I}_N$ is the $N \times N$ identity matrix.

Table 4.1: Parameter Notation

Parameter	Description
$\mathcal{R}$	The set of RRUs
$\mathcal{B}$	The set of BBUs
$\mathcal{U}$	The set of users
$\mathcal{C}$	The set of RRU clusters
$\mathcal{V}$	The set of RRUs, BBUs and forwarding nodes
$\mathcal{F}$	The set of functional splits
$\mathcal{E}$	The set of FH links between RRUs, BBUs, and forwarding nodes
$\mathcal{P}_{i,n}^{m}$	The set of FH routing paths from RRU r_i to BBU b_n with split f_m
Π	The set of FH routing path selection variables
$\mathbf{Q}$	Associable RRU indicator matrix
$\overline{\mathbf{Q}}$	Associable cluster indicator matrix
$\mathbf{C}$	RRU clustering variable matrix
$\overline{\mathbf{C}}$	Inter-RRU clustering relation indicator matrix
$\mathbf{N}$	RRU number normalization diagonal matrix
$\mathbf{X}$	User association variable matrix
$\overline{\mathbf{X}}$	Inter-user interference indicator matrix
$\mathbf{F}$	Functional split variable matrix
$\mathbf{A}$	BBU anchoring variable matrix
$\mathbf{E}$	FH routing variable matrix
$\overline{\mathbf{E}}$	Routable FH link toward BBU indicator matrix
$\mathbf{H}$	Overall channel matrix
$\mathbf{S}$	Signal to interference plus noise ratio matrix

nodes $\mathcal{D} := \{d_1, d_2, \cdots\}$. Finally, the set of all nodes comprising RRUs, forwarding nodes, BBUs is formed as $\mathcal{V} := \mathcal{R} \cup \mathcal{D} \cup \mathcal{B}$, and the set of all directional FH link within the network topology is denoted as $\mathcal{E}$.

4.2.2 RRU clustering and user association

To cooperate among RRUs and to reduce the inter-cell interference, all RRUs are dynamically grouped into $|\mathcal{C}|$ disjoint RRU clusters as $\mathcal{C} := \left\{c_1, \cdots, c_{|\mathcal{C}|}\right\}$, in which there are $|c_l|$ RRUs in the l-th cluster. Hence, the $|\mathcal{R}| \times |\mathcal{R}|$ variable matrix is denoted as $\mathbf{C}$, where its (i,j)-th element $c_{i,j}$ will be 1 if the i-th RRU belongs to the j-th cluster (0 otherwise). Nevertheless, in reality, at most $C_{\max}$ RRU cooperation is affordable due to the control signaling overhead and processing complexity, i.e., $|c_l| \leqslant C_{\max}, \forall c_l \in \mathcal{C}$. We also denote $\overline{\mathbf{C}} = \mathbf{C} \cdot \mathbf{C}^T$, where its (i,j)-th element $\overline{c}_{i,j}$ is 1 if both the i-th and the j-th RRUs belong to the same cluster and is 0 otherwise. Further, the $|\mathcal{R}| \times |\mathcal{R}|$ normalized diagonal matrix is formed as $\mathbf{N}$ in Eq. (4.2), in which the diag $(\cdot)$ operator can build a diagonal matrix whose diagonal elements are from the input column vector.

$$\mathbf{N} = \text{diag}\left(\left[\frac{1}{\sum_{j=1}^{|\mathcal{R}|}\overline{c}_{1,j}}, \frac{1}{\sum_{j=1}^{|\mathcal{R}|}\overline{c}_{2,j}}, \cdots, \frac{1}{\sum_{j=1}^{|\mathcal{R}|}\overline{c}_{|\mathcal{R}|,j}}\right]^T\right) \tag{4.2}$$

Table 4.2: Parameters of direct FH link

Characteristic	Parameter	Description
Delay	**t**	FH link delay in second
Capacity	**l**	FH link capacity in bits per second

Moreover, we denote $\overline{\mathbf{Q}} \coloneqq \overline{\mathbf{C}} \cdot \mathbf{Q} \succ \mathbb{0}_{|R|\times|U|}$ as the associable cluster indicator matrix, in which $\succ$ is the element-wise "greater than" relational operator and its (i,j)-th element $\overline{q}_{i,j}$ is 1 if the j-th user can be associated with the i-th RRU considering the RRU clustering effect and is 0 otherwise. Afterwards, the $|\mathcal{R}| \times |\mathcal{U}|$ variable matrix $\mathbf{X}$ is used to represent the user association control decisions, in which its j-th column ($1 \leqslant j \leqslant |\mathcal{U}|$) is denoted as $\mathbf{x}_j$ and its (i,j)-th element $x_{i,j}$ is 1 if the j-th user is associated to the i-th RRU and is 0 otherwise. Finally, the inter-user interference indicator matrix is defined as $\overline{\mathbf{X}} \coloneqq \mathbf{X}^T \cdot \left(\mathbb{1}_{|R|\times|U|} - \mathbf{X}\right) \succ \mathbb{0}_{|U|\times|U|}$, where its (i,j)-th element $\overline{x}_{i,j}$ is 1 if the inter-user interference exists between the i-th and the j-th users and is 0 otherwise.

4.2.3 Functional split and BBU anchoring

The processing of a single BS is decomposed into a number of network functions according to the applied functional split between RRU and BBU. A set of considered functional split options is formed as $\mathcal{F} \coloneqq \left\{f_1, \cdots, f_{|F|}\right\}$, i.e., $\mathcal{F} = \{\text{Split A}, \text{Split B}, \text{Split C}, \text{Split D}, \text{Split E}\}$ for our considered five physical layer functional split options. To enable the joint processing among RRUs within a cluster, the same functional split is applied at RRUs within the same cluster and the centralized processing of these RRUs are collocated at the same anchor BBU. Hence, we define the $|\mathcal{R}| \times |\mathcal{F}|$ variable matrix $\mathbf{F}$ denoting the applied functional split, in which its (i,j)-th element $f_{i,j}$ is 1 if the i-th RRU applies the j-th functional split option in $\mathcal{F}$ and is 0 otherwise. Further, the $|\mathcal{R}| \times |\mathcal{B}|$ BBU anchoring variable matrix $\mathbf{A}$ is defined in which its (i,j)-th element $a_{i,j}$ is 1 if the j-th BBU is the anchor BBU of the i-th RRU and is 0 otherwise. Note that different functional splits imply different characteristics in terms of the coordinated processing, the processing time for RRU/BBU, and the FH traffic throughput. We will further elaborate on these characteristics in the following sections.

4.2.4 FH network routing

To route the data samples in the FH network between RRU and the corresponding anchor BBU, the sets of forwarding nodes (i.e., $\mathcal{D}$) and directional FH links (i.e., $\mathcal{E}$) are utilized. Two critical per-link characteristics are highlighted: delay and capacity as $\mathbf{t}$ and $\mathbf{l}$ denoted in Table 4.2 respectively. Both parameters can be represented as the $|\mathcal{E}| \times 1$ column vectors and the i-th entry within $\mathbf{t}$ and $\mathbf{l}$, i.e., t_i and l_i, show the respective values for the i-th FH link in set $\mathcal{E}$. Further, we define the $|\mathcal{R}| \times |\mathcal{E}|$ variable matrix $\mathbf{E}$, in which its (i,j)-th entry $e_{i,j}$ is 1 if the j-th FH link in $\mathcal{E}$ is used by the i-th RRU to route the FH traffic toward its anchor BBU and is 0 otherwise. Another $|\mathcal{E}| \times |\mathcal{B}|$ indicator matrix $\overline{\mathbf{E}}$ is defined in which its (i,j)-th entry $\overline{p}_{i,j}$ is 1 if the j-th FH link in $\mathcal{E}$ can be utilized to route FH traffic toward the j-th BBU and is 0 otherwise.

4.3 Problem formulation

Based on the aforementioned system model, we then formulate the overall optimization problem. As highlighted beforehand, the targeting benefit of centralization in this chapter is to have a coordinated processing capability and to boost the overall network spectral efficiency; however, every considered constraint shall be satisfied.

4.3.1 Problem overview

The overall problem is formulated in Eq. (4.4), in which our objective function of Eq. (4.4a) aims to maximize the overall network spectral efficiency summed from all $|\mathcal{U}|$ users, i.e., R_j represents the spectral efficiency of the j-th user in bps per Hz (bps/Hz). This spectral efficiency can be derived according to the Shannon capacity that takes the value of SINR as the input, in which $E[s_{j,m}]$ is the expected SINR value of the j-th user when using the m-th functional split and n_i is the i-th diagonal element within $\mathbf{N}$ defined in Eq. (4.2) for the normalization purpose. To represent it in a matrix form, we define the $|U| \times |F|$ matrix $\mathbf{S}$ that collects all SINR values as specified in Eq. (4.3) and use the Hadamard product operator (i.e., $\circ$ in Eq. (4.4a)) that takes two equally-sized matrices as the inputs to do the element-wise production.

$$\mathbf{S} = \begin{bmatrix} s_{1,A} & s_{1,B} & s_{1,C} & s_{1,D} & s_{1,E} \\ s_{2,A} & s_{2,B} & s_{2,C} & s_{2,D} & s_{2,E} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{|U|,A} & s_{|U|,B} & s_{|U|,C} & s_{|U|,D} & s_{|U|,E} \end{bmatrix} \tag{4.3}$$

Moreover, several constraints are formulated considering different design issues:

1. *RRU clustering* and *user association*: From Eq. (4.4b) to Eq. (4.4f),
2. *Functional split*: From Eq. (4.4g) to Eq. (4.4h),
3. *BBU anchoring* and *FH network routing*: From Eq. (4.4i) to Eq. (4.4m),
4. *FH traffic transportation*: From Eq. (4.4n) to Eq. (4.4o).

In the following, we detailedly elaborate on all these constraints.

4.3.2 Formulated constraints

4.3.2.1 RRU clustering and user association

The constraint of Eq. (4.4b) implies that each RRU shall belong to only one cluster. Further, the number of RRUs in a cluster can not exceed $C_{\max}$ as mentioned beforehand, as restricted by Eq. (4.4c). Moreover, Eq. (4.4d) states that each user shall be served by a number of RRUs within exactly one cluster. Within Eq. (4.4e), it ensures that each user is only associated to the associable RRUs based on the $\overline{\mathbf{Q}}$ defined in Section 4.2.2. Finally, Eq. (4.4f) provides several insights as summarized in Table 4.3, in which all possible combinations are explained exploiting the symmetric matrix property of $\overline{\mathbf{C}}$, i.e., $\overline{c}_{i,k} = \overline{c}_{k,i}, \forall i, k$.

Table 4.3: Feasible combinations of Eq. (4.4f)

Variables			Descriptions
$x_{i,j}$	$\overline{c}_{i,k}$	$x_{k,j}$	
1	1	1	u_j is associated to r_i & r_i and r_k are in same cluster $\rightarrow$ u_j is associated to r_k
0	1	0	u_j is not associated to r_i & r_i and r_k are in same cluster $\rightarrow$ u_j is not associated to r_k
1	0	0	u_j is associated to r_i & r_i and r_k are in different clusters $\rightarrow$ u_j is not associated to r_k
0	0	0 or 1	u_j is not associated to r_i & r_i and r_k are in different clusters $\rightarrow$ u_j may or may not be associated to r_k

$$\underset{\mathbf{C},\mathbf{X},\mathbf{F},\mathbf{A},\mathbf{E}}{max} \sum_{u_j \in \mathcal{U}} R_j = \sum_{u_j \in \mathcal{U}} \sum_{r_i \in \mathcal{R}} \sum_{f_m \in \mathcal{F}} x_{i,j} \cdot n_i \cdot f_{i,m} \cdot \log\left(1 + E\left[s_{j,m}\right]\right)$$

$$= \mathbb{1}_{|\mathcal{U}|\times 1}^T \cdot \left(\mathbf{X}^T \cdot \mathbf{N} \cdot \mathbf{F} \circ \log\left(\mathbb{1}_{|\mathcal{U}|\times|\mathcal{F}|} + E\left[\mathbf{S}\right]\right)\right) \cdot \mathbb{1}_{|\mathcal{F}|\times 1} \tag{4.4a}$$

$$s.t.\ \mathbf{C} \cdot \mathbb{1}_{|\mathcal{R}|\times 1} = \mathbb{1}_{|\mathcal{R}|\times 1} \tag{4.4b}$$

$$\left\|\mathbf{C}^T \cdot \mathbb{1}_{|\mathcal{R}|\times 1}\right\|_\infty \leqslant C_{\max} \tag{4.4c}$$

$$\mathbf{X}^T \cdot \mathbf{N} \cdot \mathbb{1}_{|\mathcal{R}|\times 1} = \mathbb{1}_{|\mathcal{U}|\times 1} \tag{4.4d}$$

$$x_{i,j} \leqslant \overline{q}_{i,j}, \quad \forall r_i \in \mathcal{R},\ u_j \in \mathcal{U} \tag{4.4e}$$

$$x_{i,j} \cdot \overline{c}_{i,k} = x_{k,j} \cdot \overline{c}_{k,i} = x_{i,j} \cdot x_{k,j}, \quad \forall r_i \neq r_k \in \mathcal{R},\ u_j \in \mathcal{U} \tag{4.4f}$$

$$\mathbf{F} \cdot \mathbb{1}_{|\mathcal{F}|\times 1} = \mathbb{1}_{|\mathcal{R}|\times 1} \tag{4.4g}$$

$$f_{i,m} \cdot \overline{c}_{i,k} = f_{k,m} \cdot \overline{c}_{k,i}, \quad \forall r_i \neq r_k \in \mathcal{R},\ f_m \in \mathcal{F} \tag{4.4h}$$

$$\mathbf{A} \cdot \mathbb{1}_{|\mathcal{B}|\times 1} = \mathbb{1}_{|\mathcal{R}|\times 1} \tag{4.4i}$$

$$a_{i,n} \cdot \overline{c}_{i,k} = a_{k,n} \cdot \overline{c}_{k,i}, \quad \forall r_i \neq r_k \in \mathcal{R},\ b_n \in \mathcal{B} \tag{4.4j}$$

$$\sum_{\epsilon \in \delta^+(v)} e_{i,\epsilon} - \sum_{\epsilon \in \delta^-(v)} e_{i,\epsilon} = \begin{cases} -1, & \text{if } v = r_i \\ a_{i,n}, & \text{if } v = b_n \in \mathcal{B}\ , \forall r_i \in \mathcal{R},\ v \in \mathcal{V} \\ 0, & \text{else} \end{cases} \tag{4.4k}$$

$$\sum_{\epsilon \in \delta^+(v)} e_{i,\epsilon} \leqslant 1, \quad \forall r_i \in \mathcal{R},\ v \in \mathcal{V} \tag{4.4l}$$

$$a_{i,n} \cdot e_{i,\epsilon} \leqslant \overline{e}_{\epsilon,n}, \quad \forall r_i \in \mathcal{R},\ b_n \in \mathcal{B},\ \epsilon \in \mathcal{E} \tag{4.4m}$$

$$T_R\left(f_{i,1}, \cdots, f_{i,|\mathcal{F}|}\right) + \sum_{e_{i,\epsilon}==1, \forall \epsilon \in \mathcal{E}} t_\epsilon + T_B\left(f_{i,1}, \cdots, f_{i,|\mathcal{F}|}\right) \leqslant T_{rx}^{\max}, \forall r_i \in \mathcal{R} \tag{4.4n}$$

$$\sum_{r_i \in \mathcal{R}} e_{i,\epsilon} \cdot W_R\left(f_{i,1}, \cdots, f_{i,|\mathcal{F}|}, x_{i,1}, \cdots, x_{i,|\mathcal{U}|}\right) \leqslant l_\epsilon, \quad \forall \epsilon \in \mathcal{E} \tag{4.4o}$$

$$x_{i,j}, c_{i,k}, f_{i,m}, a_{i,n}, e_{i,\epsilon} \in \{0,1\}, \forall r_i, r_k \in \mathcal{R}, x_j \in \mathcal{U}, f_m \in \mathcal{F}, b_n \in \mathcal{B}, \epsilon \in \mathcal{E} \tag{4.4p}$$

Table 4.4: Feasible combinations of Eq. (4.4h)

Variables			Description
$f_{i,m}$	$\overline{c}_{i,k}$	$f_{k,m}$	
1	1	1	r_i uses the m-th functional split & r_i and r_k are in same cluster $\rightarrow$ r_k uses the m-th functional split
0	1	0	r_i does not use m-th functional split & r_i and r_k are in same cluster $\rightarrow$ x_i does not use the m-th functional split
0 or 1	0	0 or 1	r_i and r_k are in different clusters $\rightarrow$ No relation between the applied functional splits of r_i and r_k

Table 4.5: Feasible combinations of Eq. (4.4j)

Variables			Description
$a_{i,n}$	$\overline{c}_{i,k}$	$a_{k,n}$	
1	1	1	r_i is anchored to BBU b_n & r_i and r_k are in same cluster $\rightarrow$ r_k is anchored to BBU b_n
0	1	0	r_i is not anchored to BBU b_n & r_i and r_k are in same cluster $\rightarrow$ r_k is not anchored to BBU b_n
0 or 1	0	0 or 1	r_i and r_k are in different clusters $\rightarrow$ No relation between the anchor BBU of r_i and r_k

4.3.2.2 Functional split

In terms of the constraints related to the functional split, Eq. (4.4g) identifies that each RRU shall apply one functional splits among the considered functional split options in $\mathcal{F}$. Further, Table 4.4 explains all feasible combinations of Eq. (4.4h), through which we can guarantee that the RRUs within the same RRU cluster will apply the same function split. Note that this "same functional split" constraint is necessary to enable the coordinated processing among RRUs in the same cluster.

4.3.2.3 BBU anchoring and FH network routing

In Eq. (4.4i), each RRU is guaranteed to be anchored to only one BBU. Then, all feasible combinations of Eq. (4.4j) are explained in Table 4.5 to ensure that the RRUs within the same cluster are anchored to the same BBU. Moreover, two common functions are defined for each vertex $v \in \mathcal{V}$ in the FH network, i.e., $\delta^+(v)$ and $\delta^-(v)$, in order to represent the outgoing and incoming edges of vertex v, respectively. Via utilizing these two functions, the standard flow conservation constraint is provided in Eq. (4.4k) from the source node at the i-th RRU (r_i) to the sink note at the n-th BBU (b_n) if the i-th RRU anchors to the n-th BBU (i.e., $a_{i,n} = 1$). Moreover, Eq. (4.4l) restricts the outgoing degree of each vertex can not be larger than 1. Finally, Eq. (4.4m) aims to avoid using some edges that can not be routed toward the destination leveraging $\overline{e}_{\epsilon,j}$ as the (ϵ, i) element within the defined $\overline{\mathbf{E}}$ matrix in Section 4.2.4.

4.3.2.4 Fronthaul traffic transportation

Among Eq. (4.4n) and Eq. (4.4o), the respective delay constraint and capacity constraint are provided for the FH traffic transportation. We can observe that the overall delay in Eq. (4.4n) can not exceed the maximum allowable delay in the uplink direction $T_{\text{max,rx}}$ and it can be decomposed into three components: (i) the RRU processing time $T_R(\cdot)$, (ii) a summation of FH link delay t_ϵ as explained in Table 4.2 over the FH routing path, and (iii) the BBU processing time $T_B(\cdot)$. Note that the first and the third components are related to the applied functional split, e.g., $\left\{f_{i,1}, \cdots, f_{i,|\mathcal{F}|}\right\}$ for the i-th RRU. Further, the capacity constraint in Eq. (4.4o) is to ensure that the summation of the FH throughput (i.e., $W_R(\cdot)$) from all RRUs that passes through the FH link ϵ (i.e., $\{r_i : \forall e_{i,\epsilon} = 1\}$) will not exceed the FH link capacity l_ϵ, as defined in Table 4.2. Note that the value of $W_R(\cdot)$ is dependent on the applied functional split and the user association, as explained beforehand.

4.3.3 SINR formulation

We hereby formulate the SINR of different coordination schemes that are applicable for each functional split. These SINR values are used extensively in the objective function of Eq. (4.4a). For simplicity, we assume that the power of the transmitted symbol from each user[3] and the power of additive white Gaussian noise (AWGN) at each receiving antenna are the same as $P_s = 1$ and N_0, respectively. We then denote the $(M \cdot |\mathcal{R}|) \times |\mathcal{U}|$ overall channel matrix $\mathbf{H}$ in Eq. (4.5), which is composed from all channel vectors $\mathbf{h}_{r_i,u_j}, \forall r_i \in \mathcal{R}, u_j \in \mathcal{U}$. The j-th column of this overall channel matrix is denoted as $\mathbf{h}_{u_j}$ and it will be utilized in the following SINR derivations. Afterwards, we formulate the SINR of our considered five functional splits for the j-th user as $s_{j,A}$, $s_{j,B}$, $s_{j,C}$, $s_{j,D}$, and $s_{j,E}$ in the following paragraphs: (a) **joint reception** over the time, frequency and user domains for Splits A, B, and C respectively (b) **soft symbol combination** for Split D, and (c) **transport block selection** for Split E.

$$\mathbf{H} = \left[\mathbf{h}_{u_1}, \mathbf{h}_{u_2}, \cdots, \mathbf{h}_{u_{|U|}}\right] = \begin{bmatrix} \mathbf{h}_{r_1,u_1} & \mathbf{h}_{r_1,u_2} & \cdots & \mathbf{h}_{r_1,u_{|U|}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{h}_{r_{|R|},u_1} & \mathbf{h}_{r_{|R|},u_2} & \cdots & \mathbf{h}_{r_{|R|},u_{|U|}} \end{bmatrix} \tag{4.5}$$

4.3.3.1 Splits A, B, and C: Joint reception

As for the joint reception over the time and the frequency domains, there is no major performance differences as they can be transformed to another domain using the DFT/IDFT operations. However, the joint reception over the user domain will process the allocated resource blocks for each user separately, and thus the extra inter-carrier interference (ICI) [198] may occur due to the different frequency offsets from different users. We then apply the similar approach adopted in [199] to model this interference between users, in which a portion (i.e., $0 \leqslant r_{ici} \leqslant 1$) of the desired signal power is viewed as the extra AWGN.

In the following, we elaborate on the SINR formulation for the j-th user. First of all, the extended associated user indicator matrix is formed as $\mathbf{X}_j^{ext}$ in Eq. (4.6), in which $\otimes$ is the Kronecker product operator. Then, we denote the effective channel indicator matrix as $\overline{\mathbf{X}}_j^{ext}$ via removing every all-zero row within $\mathbf{X}_j^{ext}$ and write the received signal vector from

[3]The transmitted symbols from different users are independent.

the j-th user toward all receiving antennas of the associated RRUs as $\mathbf{y}_j$ in Eq. (4.7a), where s_j is the j-th user transmitted symbol with zero mean and unit variance (i.e., $P_s = 1$), $\boldsymbol{\eta}_j$ is the independent and identically distributed AWGN with zero mean and variance N_0 for its elements, and $\tilde{\mathbf{h}}_{j,k} = \overline{\mathbf{X}}_j^{ext} \cdot \mathbf{h}_{u_k}$ is the effective channel vector from the k-th user to the associated RRUs for the j-th user. Note that $\mathbf{y}_j$ is the $M_j \times 1$ column vector with its size equals to $M_j = M \cdot \left(\mathbb{1}_{|R|\times 1} \cdot \mathbf{x}_j\right)$, where $\mathbf{x}_j$ is the j-th column of $\mathbf{X}$ as introduced previously in Section 4.2.2. We can notice that the interference is stemmed from other users that are not associated to the same RRU cluster, i.e., $\mathcal{N}_j = \{u_k : \overline{x}_{j,k} == 1\}$, using our pre-defined $\overline{\mathbf{X}}$. Then, we can use the minimum mean square error (MMSE) receiver design at the anchor BBU with the MMSE vector for the j-th user as $\mathbf{w}_j$ in Eq. (4.7b). Finally, the SINR results for Splits A and B are denoted as $s_{j,A}$ and $s_{j,B}$ in Eq. (4.7c).

$$\mathbf{X}_j^{ext} = \begin{bmatrix} x_{1,j} \otimes \mathbf{I}_M & \mathbf{0}_{M\times M} & \cdots & \mathbf{0}_{M\times M} \\ \mathbf{0}_{M\times M} & x_{2,j} \otimes \mathbf{I}_M & \cdots & \mathbf{0}_{M\times M} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{M\times M} & \mathbf{0}_{M\times M} & \cdots & x_{|R|,j} \otimes \mathbf{I}_M \end{bmatrix} \tag{4.6}$$

$$\begin{aligned} \mathbf{y}_j &= \overline{\mathbf{X}}_j^{ext} \cdot \mathbf{h}_{u_j} \cdot s_j + \sum_{u_k \in \mathcal{N}_j} \overline{\mathbf{X}}_j^{ext} \cdot \mathbf{h}_{u_k} \cdot s_k + \boldsymbol{\eta}_j \\ &= \tilde{\mathbf{h}}_{j,j} \cdot s_j + \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot s_k + \boldsymbol{\eta}_j \end{aligned} \tag{4.7a}$$

$$\mathbf{w}_j = \left(\tilde{\mathbf{h}}_{j,j} \cdot \tilde{\mathbf{h}}_{j,j}^H + \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j} \right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j} \tag{4.7b}$$

$$s_{j,A} = s_{j,B} = \frac{\mathbf{w}_j^H \cdot \tilde{\mathbf{h}}_{j,j} \cdot \tilde{\mathbf{h}}_{j,j}^H \cdot \mathbf{w}_j}{\mathbf{w}_j^H \cdot \left(\sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j} \right) \cdot \mathbf{w}_j} \tag{4.7c}$$

As for the split C, the received signal $\mathbf{y}_j$ can be written in Eq. (4.8a), where the desired signal from the j-th user is scaled by a factor of $\sqrt{1 - r_{ici}}$ and the extra additive interference due to aforementioned ICI is treated as $\mathbf{i}_j \sim CN\left(\mathbb{0}_{M_j\times 1}, \mathbf{Z}_j\right)$ with $\mathbf{Z}_j = r_{ici} \cdot E\left[\tilde{\mathbf{h}_{j,j}} \tilde{\mathbf{h}}_{j,j}^H\right]$. Then, the MMSE receiver vector is denoted as $\mathbf{w}_{j,C}$ in Eq. (4.8b) and the final SINR $s_{j,C}$ is formulated in Eq. (4.8c). We can observe that such SINR value will be the same as the ones of Splits A and B in Eq. (4.7c) when r_{ici} becomes 0.

$$\mathbf{y}_j = \sqrt{1 - r_{ici}} \cdot \tilde{\mathbf{h}}_{j,j} \cdot s_j + \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot s_k + \mathbf{i}_j + \boldsymbol{\eta}_j \tag{4.8a}$$

$$\mathbf{w}_{j,C} = \left(\frac{(1 - r_{ici}) \cdot \tilde{\mathbf{h}}_{j,j} \cdot \tilde{\mathbf{h}}_{j,j}^H + \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + \mathbf{Z}_j + N_0 \cdot \mathbf{I}_{M_j}}{\sqrt{1 - r_{ici}}} \right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j} \tag{4.8b}$$

$$s_{j,C} = \frac{(1 - r_{ici}) \cdot \mathbf{w}_{j,C}^H \cdot \tilde{\mathbf{h}}_{j,j} \cdot \tilde{\mathbf{h}}_{j,j}^H \cdot \mathbf{w}_{j,C}}{\mathbf{w}_{j,C}^H \cdot \left(\sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + \mathbf{Z}_j + N_0 \cdot \mathbf{I}_{M_j} \right) \cdot \mathbf{w}_{j,C}} \tag{4.8c}$$

4.3.3.2 Split D: Soft symbol combination

Due to the distributed equalization and demodulation processing at each RRU, the aforementioned *joint reception* scheme is not applied to the Split D. In this sense, we assume that the equalized soft symbols from RRUs within the same RRU cluster can be combined by the anchor BBU[4]. More specifically, the maximum ratio combination scheme[5] is applied in order to maximize the SINR value after the combination. Then, the received signal from the j-th user toward the i-th RRU can be formed as $\mathbf{y}_{i,j}$ in Eq. (4.9a)[6] and the corresponding MMSE vector is denoted as $\mathbf{w}_{i,j}$ in Eq. (4.9b). Finally, we can form the SINR before combination from the j-th user toward the i-th RRU as $s^{uc}_{i,j}$ in Eq. (4.9c) and the combined SINR as $s_{j,D}$ in Eq. (4.9d), which is the summation of the uncombined SINR values among the associated RRUs of the j-th user, i.e., $\Gamma_j = \{r_i : x_{i,j} == 1\}$.

$$\mathbf{y}_{i,j} = \mathbf{h}_{r_i,u_j} \cdot s_j + \sum_{u_k \in \mathcal{N}_j} \mathbf{h}_{r_i,u_k} \cdot s_k + \boldsymbol{\eta}_{i,j} \tag{4.9a}$$

$$\mathbf{w}_{i,j} = \left(\mathbf{h}_{r_i,u_j} \cdot \mathbf{h}^H_{r_i,u_j} + \sum_{u_k \in \mathcal{N}_j} \mathbf{h}_{r_i,u_k} \cdot \mathbf{h}^H_{r_i,u_k} + N_0 \cdot \mathbf{I}_M \right)^{-1} \cdot \mathbf{h}_{r_i,u_j} \tag{4.9b}$$

$$s^{uc}_{i,j} = \frac{\mathbf{w}^H_{i,j} \cdot \mathbf{h}_{r_i,u_j} \cdot \mathbf{h}^H_{r_i,u_j} \cdot \mathbf{w}_{i,j}}{\mathbf{w}^H_{i,j} \left(\sum_{u_k \in \mathcal{N}_j} \mathbf{h}_{r_i,u_k} \cdot \mathbf{h}^H_{r_i,u_k} + N_0 \cdot \mathbf{I}_M \right) \cdot \mathbf{w}_{i,j}} \tag{4.9c}$$

$$s_{j,D} = \sum_{r_i \in \Gamma_j} s^{uc}_{i,j} \tag{4.9d}$$

4.3.3.3 Split E: Transport block selection

For split E, all physical layer processing are distributed at RRU; therefore, the anchor BBU can select the decoded transport blocks among all RRUs within the same cluster that is successful in the HARQ process. In this sense, the SINR of split E can be viewed as selecting the maximum uncombined SINR among RRUs within the same cluster. Specifically, we can write the j-th user SINR as $s_{j,E}$ in Eq. (4.10), in which $s^{uc}_{i,j}$ is the corresponding uncombined SINR of the j-th user to the i-th RRU derived in Eq. (4.9c).

$$s_{j,E} = \underset{r_i \in \Gamma_j}{max}\ s^{uc}_{i,j} \tag{4.10}$$

4.3.3.4 Expected SINR

Among the aforementioned paragraphs, we detailedly formulate the instantaneous SINR for all considered functional splits. In the following, we go one step further via deriving the expected value of SINR, i.e., $E[s_{j,A}]$, $E[s_{j,B}]$, $E[s_{j,C}]$, $E[s_{j,D}]$, and $E[s_{j,E}]$ for the j-th user, that is used in the objective function of Eq. (4.4a).

For splits A and B, we first define $\beta_{j,A/B}$ in Eq. (4.11a) and write the variance of the numerator and denominator in its SINR of Eq. (4.7c) as $P_s \cdot \beta^2_{j,A/B}$ and $P_s \cdot \left(\beta_{j,A/B} - \beta^2_{j,A/B} \right)$ respectively

[4]The LLR combination can also be applied but its performance depends on the modulation scheme.

[5]Some simpler schemes such as the equal gain combination can be applied but with an inferior SINR.

[6]For simplicity, the same notation $\boldsymbol{\eta}_{i,j}$ is used here for the AWGN at the i-th RRU when receiving the j-th user data.

using the similar approach in [200]. Hence, the SINR value can be rewritten in Eq. (4.11b) following the similar approach adopted in [201], in which $\mathbf{V}_j^H \cdot \mathbf{\Lambda}_j \cdot \mathbf{V}_j = \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H$ is the eigenvalue decomposition. As $\mathbf{V}_j$ is a unitary matrix, the overall channel power is the same for both $\tilde{\mathbf{h}}_{j,j}$ and $\check{\mathbf{h}}_j = \mathbf{V}_j \cdot \tilde{\mathbf{h}}_{j,j}$, i.e., $\check{\mathbf{h}}_j^H \cdot \check{\mathbf{h}}_j = \tilde{\mathbf{h}}_{j,j}^H \cdot \tilde{\mathbf{h}}_{j,j}$; however, the statistic of each element of $\check{\mathbf{h}}_j$ relies on the corresponding power of desired signal and interference. In specific, the variance of the k-th element $\check{h}_{j,k}$ in $\check{\mathbf{h}}_j$ can be written as $\check{\sigma}_{j,k}^2 = E\left[\left|\check{h}_{j,k}\right|^2\right]$ in Eq. (4.11c). Hence, the SINR within Eq. (4.11b) will depend on the equivalent channel power (i.e., $\left|\check{h}_{j,k}\right|^2$), the AWGN variance (i.e., N_0), and the eigenvalues in $\mathbf{\Lambda_j}$ (i.e, from $\lambda_{j,1}$ to $\lambda_{j,|\mathcal{N}_j|}$). Afterwards, the expected SINR is formed in Eq. (4.11d) and the expectation term $E\left[\frac{1}{\lambda_{j,i}+N_0}\right]$ depends on the probability distribution function (PDF) of eigenvalues, i.e., $f\left(\lambda_{j,1}, \cdots, \lambda_{j,|\mathcal{N}_j|}\right)$, which is mostly known in random matrix theory [202] only for some particular forms[7]. To tackle the general case of arbitrary channel variance values, we collect this expectation term through the Monte Carlo simulations.

$$\begin{aligned}
\beta_{j,A/B} &= \mathbf{w}_j^H \cdot \tilde{\mathbf{h}}_{j,j} = \tilde{\mathbf{h}}_{j,j}^H \cdot \left(\tilde{\mathbf{h}}_{j,j} \cdot \tilde{\mathbf{h}}_{j,j}^H + \sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j} \\
&= \frac{\tilde{\mathbf{h}}_{j,j}^H \cdot \left(\sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j}}{1 + \tilde{\mathbf{h}}_{j,j}^H \cdot \left(\sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j}}
\end{aligned} \tag{4.11a}$$

$$\begin{aligned}
s_{j,A/B} &= \frac{\beta_{j,A/B}}{1-\beta_{j,A/B}} = \tilde{\mathbf{h}}_{j,j}^H \cdot \left(\sum_{u_k \in \mathcal{N}_j} \tilde{\mathbf{h}}_{j,k} \cdot \tilde{\mathbf{h}}_{j,k}^H + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j} \\
&= \tilde{\mathbf{h}}_{j,j}^H \cdot \left(\mathbf{V}_j^H \cdot \mathbf{\Lambda}_j \cdot \mathbf{V}_j + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \tilde{\mathbf{h}}_{j,j} = \check{\mathbf{h}}_j^H \cdot \left(\mathbf{\Lambda}_j + N_0 \cdot \mathbf{I}_{M_j}\right)^{-1} \cdot \check{\mathbf{h}}_j \\
&= \sum_{k=1}^{|\mathcal{N}_j|} \frac{1}{\lambda_{j,k}+N_0} \cdot \left|\check{h}_{j,k}\right|^2 + \frac{1}{N_0} \sum_{k=|\mathcal{N}_j|+1}^{M_j} \left|\check{h}_{j,k}\right|^2
\end{aligned} \tag{4.11b}$$

$$\check{\sigma}_{j,k}^2 = \begin{cases} \frac{\sum_{r_i \in \Gamma_j} \sigma_{r_i,u_j}^2 \cdot \sigma_{r_i,u_{k'}}^2}{\sum_{r_i \in \Gamma_j} \sigma_{r_i,u_{k'}}^2}, & k \leqslant |\mathcal{N}_j|, u_{k'} \text{ is the } k^{\text{th}} \text{ element in } \mathcal{N}_j \\ \frac{M \cdot \sum_{r_i \in \Gamma_j} \sigma_{r_i,u_j}^2 - \sum_{k=1}^{|\mathcal{N}_j|} \check{\sigma}_{j,k}^2}{M_j - |\mathcal{N}_j|}, & |\mathcal{N}_j| < k \leqslant M_j \end{cases} \tag{4.11c}$$

$$E\left[s_{j,A/B}\right] = \sum_{k=1}^{|\mathcal{N}_j|} E\left[\frac{1}{\lambda_{j,k}+N_0}\right] \cdot \check{\sigma}_{j,k}^2 + \frac{1}{N_0} \sum_{k=|\mathcal{N}_j|+1}^{M_j} \check{\sigma}_{j,k}^2 \tag{4.11d}$$

Following the similar steps, the expected SINR of split C is written in Eq. (4.12a), in which $z_{j,k}$ is the k-th diagonal element of $\mathbf{Z}_j$ formed in Section 4.3.3.1. We can observe that the ICI not only degrades the desired signal power by a factor of $(1 - r_{ici})$ but also introduces the interferences as the extra additive noise with the variance proportional to r_{ici}.

[7] Such as all unit variance channel vectors from all interferer users.

$$E\left[s_{j,C}\right] = (1 - r_{ici}) \cdot \left(\sum_{k=1}^{|\mathcal{N}_j|} E\left[\frac{1}{\lambda_{j,k} + z_{j,k} + N_0} \right] \cdot \check{\sigma}^2_{j,k} + \sum_{k=|\mathcal{N}_j|+1}^{M_j} \frac{\check{\sigma}^2_{j,k}}{z_{j,k} + N_0} \right) \tag{4.12a}$$

In the following, we investigate the expected SINR value for Splits D and E. The expected value of SINR before combination from the j-th user to the i-th RRU (i.e., $s^{uc}_{i,j}$ in Eq. (4.9c)) is presented in Eq. (4.13a), in which $\lambda^{uc}_{j,k}$ is the k-th eigenvalues of $\sum_{u_k \in \mathcal{N}_j} \mathbf{h}_{r_i,u_k} \cdot \mathbf{h}^H_{r_i,u_k}$. Based on these uncombined expected SINR values, the expected SINR of Splits D and E can be formed in Eq. (4.13b) and Eq. (4.13c), respectively. Note that the expected SINR of Split D is the same as the ones of splits A and B in Eq. (4.11d), only if there is no interference (i.e., $\mathcal{N}_j = \varnothing$) or there is only one RRU in the cluster (i.e., $M_j = M$). Otherwise, Splits A and B can have a better resolution through a larger number of receiving antennas, i.e., $M_j > M$, to balance the different interference impacts (cf. $\sigma^2_{r_i,u_{k'}}$ in Eq. (4.11c)) among multiple RRUs.

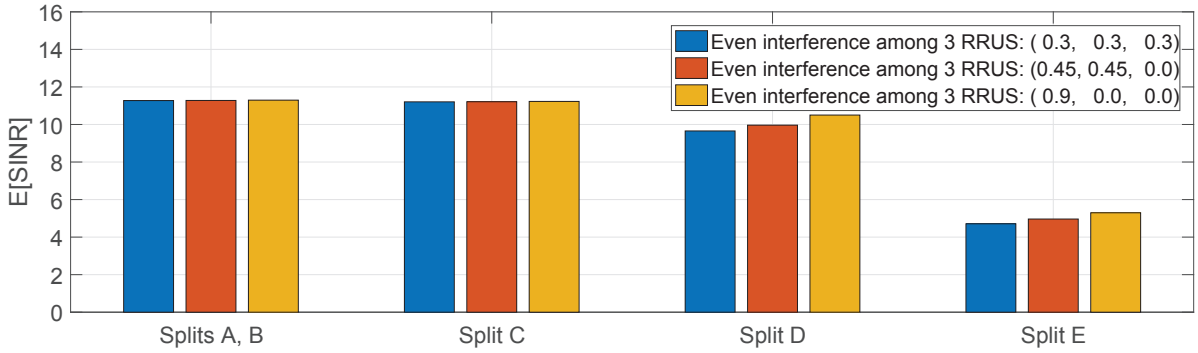
$$E\left[s^{uc}_{i,j}\right] = \sigma^2_{r_i,u_j} \cdot \left(\sum_{k=1}^{|\mathcal{N}_j|} E\left[\frac{1}{\lambda^{uc}_{j,k} + N_0} \right] + \frac{M - |\mathcal{N}_j|}{N_0} \right) \tag{4.13a}$$

$$E\left[s_{j,D}\right] = \sum_{r_i \in \Gamma_j} \sigma^2_{r_i,u_j} \left(\sum_{k=1}^{|\mathcal{N}_j|} E\left[\frac{1}{\lambda^{uc}_{j,k} + N_0} \right] + \frac{M - |\mathcal{N}_j|}{N_0} \right) \tag{4.13b}$$

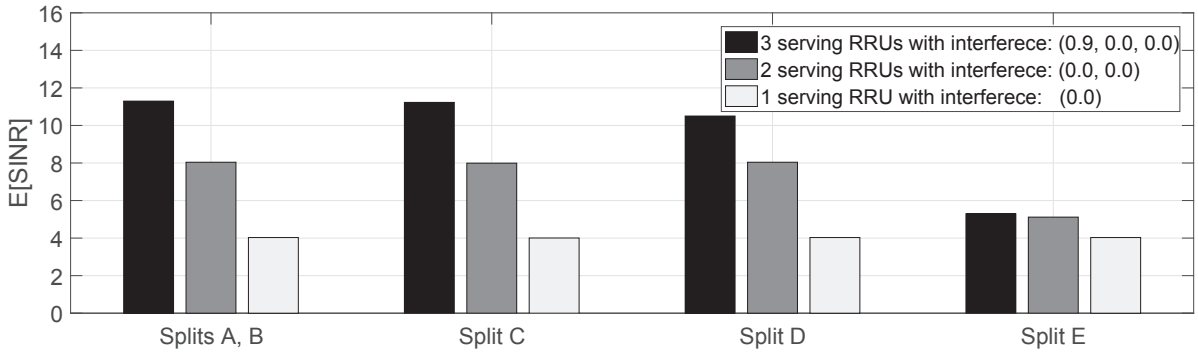
$$E\left[s_{j,E}\right] = \underset{r_i \in \Gamma_j}{max}\ E\left[s^{uc}_{i,j}\right] \tag{4.13c}$$

Before finishing this paragraph, we compare the expected SINR values among different functional splits in some iconic scenarios to highlight their strengths and weaknesses. The first considered scenario includes 1 user served by 3 RRUs with $\sigma^2_{r_1,u_1} = \sigma^2_{r_2,u_1} = \sigma^2_{r_3,u_1} = N_0 = 1$; however, there is another user generating the inter-user interference with interference power as $\sigma^2_{r_1,u_2}, \sigma^2_{r_2,u_2}, \sigma^2_{r_3,u_2}$. In Figure 4.3a, we compare the expected SINR values of different functional splits when distributing the interference power from that interference user is different manners: even ($\sigma^2_{r_1,u_2} = \sigma^2_{r_2,u_2} = \sigma^2_{r_3,u_2} = 0.3$), uneven ($\sigma^2_{r_1,u_2} = \sigma^2_{r_2,u_2} = 0.6$ and $\sigma^2_{r_3,u_2} = 0$) and mostly uneven ($\sigma^2_{r_1,u_2} = 0.9$ and $\sigma^2_{r_2,u_2} = \sigma^2_{r_3,u_2} = 0$) cases. We can see that Splits A, B, and C are less sensitive to the interference interference distribution, as expected in our model. In contrast, both Split D and Split E are sensitive to the interference distribution. Specifically, Split D shows a closer performance to the joint reception scenario (i.e., Splits A, B, and C) when interference is more uneven. The reason behind is that the growth of the eigenvalues in Eq. (4.13a) will have fewer impacts than reducing the number of interferers in $\mathcal{N}_j$.

On the other hand, some further cases are considered via gradually removing the serving RRUs from the most uneven case shown in Figure 4.3a. The results are depicted in Figure 4.3b, in which we first remove the most interfering RRU (i.e., r_1), then we remove another RRU in the cluster (i.e., r_2). After removing the most interfering RRU, we can see that Split D will have the same performance as Splits A, B, and C, since there is no interference anymore as expected beforehand (i.e., $\mathcal{N}_j = \varnothing$). Finally, all splits will have the identical performance when there is only one serving RRU in the RRU cluster, i.e., $M_j = M$.



(a) SINR comparison of different interference power distributions



(b) SINR comparison of different interference power levels

Figure 4.3: SINR comparison of different scenarios

4.3.4 Problem analysis

Within the formulated problem of Eq. (4.4), the aim is to allocate binary values toward all $|\mathcal{R}| \times (|\mathcal{R}| + |\mathcal{U}| + |\mathcal{F}| + |\mathcal{B}| + |\mathcal{E}|)$ entries of the involved variable matrices: RRU clustering ($\mathbf{C}$), user association ($\mathbf{X}$), functional split ($\mathbf{F}$), BBU anchoring ($\mathbf{A}$), and FH routing ($\mathbf{E}$). Specifically, this problem can be proved to have *NP-hard* complexity.

Theorem 4.1. *The problem of Eq.* (4.4) *is NP-hard to solve.*

Proof. This problem can be polynomially reduced into the multi-dimensional multiple-choice knapsack problem (MMKP) with NP-hard complexity [203]. First of all, we consider a specific instance of this problem with (i) fixed $\mathbf{C}$, $\mathbf{X}$ and $\mathbf{A}$ satisfying the constraints from Eq. (4.4b) to Eq. (4.4f), Eq. (4.4i), and Eq. (4.4j), (ii) a FH network guaranteeing there is always a routing path between each pair of RRU and BBU in order to satisfy the constraints from Eq. (4.4k) to Eq. (4.4m), (iii) sufficiently large capacity (e.g., $t_\epsilon \to \infty, \forall \epsilon$) and small delay (e.g., $l_\epsilon \to 0^+, \forall \epsilon$) for all FH links without impacting the satisfaction of Eq. (4.4n) and Eq. (4.4o), and (iv) affine function to model $T_R(\cdot)$, $T_B(\cdot)$ and $W_R(\cdot)$ in terms of the function split input.

In consequence, we can rewrite the constraints of Eq. (4.4g), Eq. (4.4h), Eq. (4.4n) and Eq. (4.4o) as the linear equalities and inequalities in term of $\overline{f}_{i,m}, \forall u_i \in \mathcal{U}, f_m \in \mathcal{F}$. Also, the objective function can be written as the linear function of $\overline{f}_{i,m}$. To sum up, this specific instance can be mapped into an MMKP with $|\mathcal{U}|$ classes of items and exactly one item shall be selected out of the $|\mathcal{F}|$ items for each class. □

4.4 Solution methodology

To efficiently deal with the formulated problem with an *NP-hard* complexity, we first revisit the overall problem and then propose a practical solution.

4.4.1 Problem revisiting

First of all, we can observe that the objective function is related to both $\mathbf{F}$ and $\mathbf{X}$ variables[8]; however, it is only convex in terms of $\mathbf{F}$ since the interference level within the expected SINR $E[\mathbf{S}]$ will be affected by $\mathbf{X}$ (cf. $\mathcal{N}_j$ in Eq. (4.11), Eq. (4.12) and Eq. (4.13)). Hence, one possible approach is to iteratively update the interference level in $E[\mathbf{S}]$ according to the previously-allocated $\mathbf{X}$ and the corresponding $\mathcal{N}_j$; therefore, the objective function can be viewed as a biconvex function for the continuously-relaxed $\mathbf{F}$ and $\mathbf{X}$, i.e., $f_{i,m}, x_{i,j} \in [0,1]$. Practically, such biconvex optimization can be solved through which two sets of variables are dealt alternatively using the known convex optimization approach while fixing another set of variables. Note that the continuous version of $\mathbf{X}$ can remove the mutually exclusive constraints and change the original constraint of Eq. (4.4f) into the constraint $\mathcal{C}_3$ of Eq. (4.14d). Finally, the rounding operation is taken after solving the problem and will be elaborated later on.

In correspondence to the aforementioned continuous relaxation approach, the impacts on other variables are evaluated. To clearly identify their relationship, we first define the FH routing path set $\mathcal{P}^m_{i,n}, \forall r_i \in \mathcal{R}, f_m \in \mathcal{F}, b_n \in \mathcal{B}$ that comprises all feasible FH routing paths from the i-th RRU to the n-th BBU using the m-th functional split satisfying constraints from Eq. (4.4k) to Eq. (4.4n). Take our interested five functional splits as examples, five respective sets are formed for each pair of RRU and BBU: $\mathcal{P}^A_{i,n} \subseteq \mathcal{P}^B_{i,n} \subseteq \mathcal{P}^C_{i,n} \subseteq \mathcal{P}^D_{i,n} \subseteq \mathcal{P}^E_{i,n}$, $\forall r_i \in \mathcal{R}$, $b_n \in \mathcal{B}$. Further, to replace the original FH routing variables $\mathbf{E}$, the new variables $\pi^m_{i,n,q}$ are defined to identify whether the q-th FH routing path within $\mathcal{P}^m_{i,n}$ is selected (i.e., 1) or not (i.e., 0) to deliver the FH traffic from the i-th RRU toward the n-th BBU using the m-th functional split. Hence, two corresponding new constraints are introduced to replace the original ones from Eq. (4.4k) to Eq. (4.4o): (i) $\mathcal{C}_6$ of Eq. (4.14g) represents the relation between the FH routing, the BBU anchoring and the functional split, and (ii) $\mathcal{C}_7$ of Eq. (4.14h) reserves the per-link capacity limitation using $I\left(\epsilon, P^m_{i,n,q}\right)$ function that will return 1 when FH link ϵ is within the path $P^m_{i,n,q}$ and will return 0 otherwise. Finally, this newly introduced variables can be relaxed in a continuous region, i.e., $\pi^m_{i,n,q} \in [0,1]$, to represent the portion of FH traffic goes through the FH routing path over $P^m_{i,n,q}$.

In contrast, the relaxation approach makes fewer senses for both BBU anchoring and RRU clustering variables. The first reason behind is that these two variables can not retain its definition after the relaxation. For instance, the relaxed $c_{i,k}$ is in contradiction to the aim of RRU clustering, since a fraction of a single RRU shall be viewed as another RRU to be in function individually. Similarly, the $a_{i,n}$ shall be kept in a binary form to centralize the processing at the same BBU (i.e., 1) or not (i.e., 0) and to retain the benefits of C-RAN. Another reason is because the relaxation can not reduce the computational complexity due to their involved non-linear constraints. Take the constraints of Eq. (4.14d) and Eq. (4.14g) as examples, the included $\overline{c}_{i,k} = \overline{c}_{k,i} = \sum_{j=1}^{|\mathcal{R}|} c_{i,j} \cdot c_{k,j}$ is the quadratic form of the RRU clustering variables. Also, the constraint of Eq. (4.14g) contains the product form of the BBU anchoring (i.e., $a_{i,n}$) and

[8]Note N_i is just used for normalization purpose.

the function split (i.e., $f_{i,m}$) variables, and thus it is neither convex nor concave. Hence, these two variables shall be dealt with the combinatorial optimization approach.

$$\max\ f(\mathbf{X},\mathbf{F}) = \sum_{u_j\in\mathcal{U}}\sum_{r_i\in\mathcal{R}}\sum_{f_m\in\mathcal{F}} x_{i,j}\cdot n_i\cdot f_{i,m}\cdot\log\left(1+E\left[s_{j,m}|\mathcal{N}_j\right]\right) \tag{4.14a}$$

$$s.t.\ \mathcal{C}_1\colon \sum_{r_i\in\mathcal{R}} x_{i,j}\cdot n_i - 1 = 0,\quad \forall u_j\in\mathcal{U} \tag{4.14b}$$

$$\mathcal{C}_2\colon x_{i,j}-\overline{q}_{i,j}\leqslant 0,\quad \forall r_i\in\mathcal{R},\ u_j\in\mathcal{U} \tag{4.14c}$$

$$\mathcal{C}_3\colon x_{i,j}\cdot\overline{c}_{i,k}-x_{k,j}\cdot\overline{c}_{k,i}=0,\quad \forall r_i\neq r_k\in\mathcal{R},\ u_j\in\mathcal{U} \tag{4.14d}$$

$$\mathcal{C}_4\colon \sum_{f_m\in\mathcal{F}} f_{i,m}-1=0,\quad \forall r_i\in\mathcal{F} \tag{4.14e}$$

$$\mathcal{C}_5\colon f_{i,m}\cdot\overline{c}_{i,k}-f_{k,m}\cdot\overline{c}_{k,i}=0,\quad \forall r_i\neq r_k\in\mathcal{R},\ f_m\in\mathcal{F} \tag{4.14f}$$

$$\mathcal{C}_6\colon \sum_{q=1}^{\left|\mathcal{P}_{i,n}^m\right|}\pi_{i,n,q}^m - a_{i,n}\cdot f_{i,m}=0,\ \forall r_i\in\mathcal{R},\ f_m\in\mathcal{F},\ b_n\in\mathcal{B} \tag{4.14g}$$

$$\mathcal{C}_7\colon \sum_{i=1}^{|\mathcal{R}|}\sum_{n=1}^{|\mathcal{B}|}\sum_{m=1}^{|\mathcal{F}|}\sum_{q=1}^{\left|P_{i,n}^m\right|}\pi_{i,n,q}^m\cdot I\left(\epsilon,P_{i,n,q}^m\right)\cdot W_R\left(f_{i,m},x_{i,j},\cdots,x_{i,|\mathcal{U}|}\right)-l_\epsilon\leqslant 0,\forall\epsilon\in\mathcal{E} \tag{4.14h}$$

$$\mathcal{C}_8\colon x_{i,j},f_{i,m},\pi_{i,n,q}^m\in[0,1],\forall r_i,\in\mathcal{R},x_j\in\mathcal{U},f_m\in\mathcal{F},b_n\in\mathcal{B},q\in\left[1,\left|\mathcal{P}_{i,n}^m\right|\right] \tag{4.14i}$$

In summary, the revisited problem is formulated in Eq. (4.14) with the objective function as $f(\mathbf{X},\mathbf{F})$ and eight sets of constraints from $\mathcal{C}_1$ to $\mathcal{C}_8$. These constraints are originated from the original problem (i.e., $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_4$ and $\mathcal{C}_5$ are from Eq. (4.4d), Eq. (4.4e), Eq. (4.4g) and Eq. (4.4h) respectively), modified due to the continuous relaxation (i.e., $\mathcal{C}_3$ is modified from Eq. (4.4f)), and added by newly-introduced variables (i.e., $\mathcal{C}_6$ and $\mathcal{C}_7$) To this end, the continuous variables $x_{i,j}, f_{i,m}, \pi_{i,n,q}^m$ are allocated given the binary values of BBU anchoring and RRU clustering variables.

4.4.2 Proposed solution

After problem revisiting, we now propose the two-level solution that is depicted in Figure 4.4 and will be elaborated in the following paragraphs.

4.4.2.1 High level

This level aims to continuously evaluate the potential update for the integer variables, i.e., $\mathbf{C}$ and $\mathbf{A}$, according to the low level optimization algorithm outcomes. Practically, the branch-and-bound (BB) method can be utilized to analyze each candidate that comprises a potential update among variables $c_{i,k}$ and $a_{i,n}, \forall r_i, r_k\in\mathcal{R}, b_n\in\mathcal{B}$ satisfying constraints of Eq. (4.4c), Eq. (4.4d), Eq. (4.4i) and Eq. (4.4j). Note that the aforementioned feasible FH routing path $\mathcal{P}_{i,n}^m$ is also formed for each pair of RRU and its anchor BBU as the input to the low-level optimization process. Then, the candidate with the most improvement will be selected to update the current solution. Nevertheless, no update is applied when either there is no candidate or all candidates can not improve the objective function anymore.

4.4.2.2 Low level

It takes the current solution of integer variables for $\mathbf{C}$ and $\mathbf{A}$ from the high level and applies the aforementioned continuous relaxation approach to (i) allocate continuous values to $x_{i,j}$, $f_{i,m}$, $\pi^m_{i,j,q}$, $\forall r_i \in \mathcal{R}, u_j \in \mathcal{U}, f_m \in \mathcal{F}, q \in \left[1, \left|\mathcal{P}^m_{i,j}\right|\right]$ through the biconvex optimization approach, and (ii) round these continuous values to the binary outcomes.

Based on the biconvex characteristic stated beforehand, two alternative stages are applied in the low-level optimization: (a) user scheduling (i,e, $x_{i,j}$), and (b) functional split and FH path routing (i.e., $f_{i,m}$ and $\pi^m_{i,j,q}$). Within each stage, the corresponding convex optimization problem is tackled with other variables being fixed. Specifically, in the first stage, the convex optimization is applied toward the objective function $f_{\mathbf{F}}(\mathbf{X})$ with constraint sets from $\mathcal{C}_1$ to $\mathcal{C}_3$ via fixing the values of $\mathbf{F}$ While the optimization in the second stage is over $f_{\mathbf{X}}(\mathbf{F})$ with constraint sets from $\mathcal{C}_4$ to $\mathcal{C}_7$ via fixing values of $\mathbf{X}$. Such alternation will be terminated when the optimization results of these two stages are converged.

Last but not least, another challenge is to properly rounding variables to the binary ones, and thus we can apply the dynamic rounding approach [204] to iteratively fix one more variable according to the ascending order of the Lagrangian function. Such Lagrangian function is composed of the objective function as well as the equality and inequality constraints multiplied with the corresponding Lagrangian multipliers. To better satisfy the equality constraints, we can set their respective Lagrangian multipliers to be large enough, and thus only eligible candidates are evaluated. In practice, the rounding operation can be applied after each stage via forming the Lagrangian function as $L_1(\mathbf{X})$ and $L_2(\mathbf{F}, \mathbf{\Pi})$ in Eq. (4.15) for the first and second stages respectively, in which $\mathbf{\Pi} = \left\{\pi^m_{i,n,q}, \forall i, n, m, q\right\}$ includes all variables $\pi^m_{i,n,q}$, and $l_{\mathbf{X},i,j}$, $l_{\mathbf{F},\epsilon}$ are the corresponding Lagrangian multipliers for the constraint $c_{2,i,j}$ and $c_{7,\epsilon}$. Note that $c_{2,i,j}$ is the corresponding constraint for the i-th RRU and the j-th user within set $\mathcal{C}_2$, and $c_{7,\epsilon}$ is the corresponding constraint for the FH link ϵ within $\mathcal{C}_7$. Finally, the candidate that has the minimum Lagrangian function value is selected and the corresponding binary values are fixed. Such dynamic rounding process will lead to consequent subproblems until all variables are binarized.

$$L_1(\mathbf{X}) = -f_{\mathbf{F}}(\mathbf{X}) + \sum_{r_i \in \mathcal{R}} \sum_{u_j \in \mathcal{U}} l_{\mathbf{X},i,j} \cdot c_{2,i,j}(\mathbf{X}) \tag{4.15a}$$

$$L_2(\mathbf{F}, \mathbf{\Pi}) = -f_{\mathbf{X}}(\mathbf{F}) + \sum_{\epsilon \in \mathcal{E}} l_{\mathbf{F},\epsilon} \cdot c_{7,\epsilon}(\mathbf{\Pi}) \tag{4.15b}$$

4.5 Simulation results

Due to the time limitation, several extensive simulations and insight surveys are still undergoing; therefore, we will provide a full results in the final manuscript. In the following, we show the results of a simple C-RAN topology in Figure 4.5, in which there are 8 users, 4 RRUs, 6 forwarding nodes, and 2 BBUs. The bold and thin dashed line between each user and RRUs indicate the relationship of anchor RRU (e.g., u_3 and r_2) and associable RRU (e.g., u_3 and r_1), respectively. Moreover, detailed parameters can be found in Table 4.6 and we evaluate our results using the self-built MATLAB simulator.

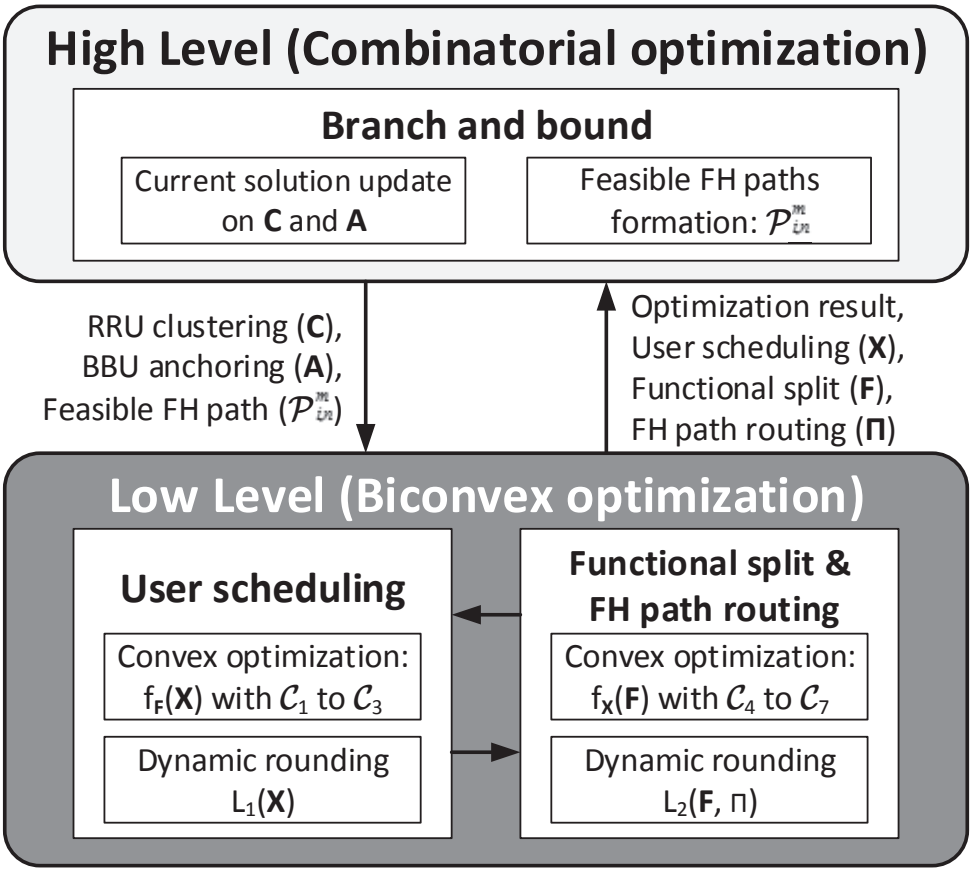


Figure 4.4: Proposed solution of the formulated problem

Table 4.6: Parameters of considered C-RAN topology

Parameter	Value
Variance of channel vector	Anchor RRU: 4, Associable RRU: 2, Other RRUs: 0.1
γ_{th}	2
P	1
N_0	0.1
C_{max}	3
γ_{ici}	0
t_ϵ $(\forall \epsilon \in \mathcal{E})$	∞
l_ϵ $(\forall \epsilon \in \mathcal{E})$	0^+

First of all, we can apply the exhaustive search scheme to find the optimal solution and our provided solution can reach that optimal solution, as the rank 1 listed in Table 4.7. However, to detailedly understand the effect of RRU clustering, user association and functional split on the spectral efficiency, we go through other combinations listed in the Table 4.7. For simplicity, U_{c_i} and F_{c_i} represent the set of associated user and applied functional split for the i-th RRU cluster.

$$U = \begin{Bmatrix} u_1, u_2, u_3, u_4, \\ u_5, u_6, u_7, u_8 \end{Bmatrix} \quad R = \{r_1, r_2, r_3, r_4\} \quad V = \{v_1, v_2, v_3, v_4, v_5, v_6\} \quad B = \{b_1, b_2\}$$

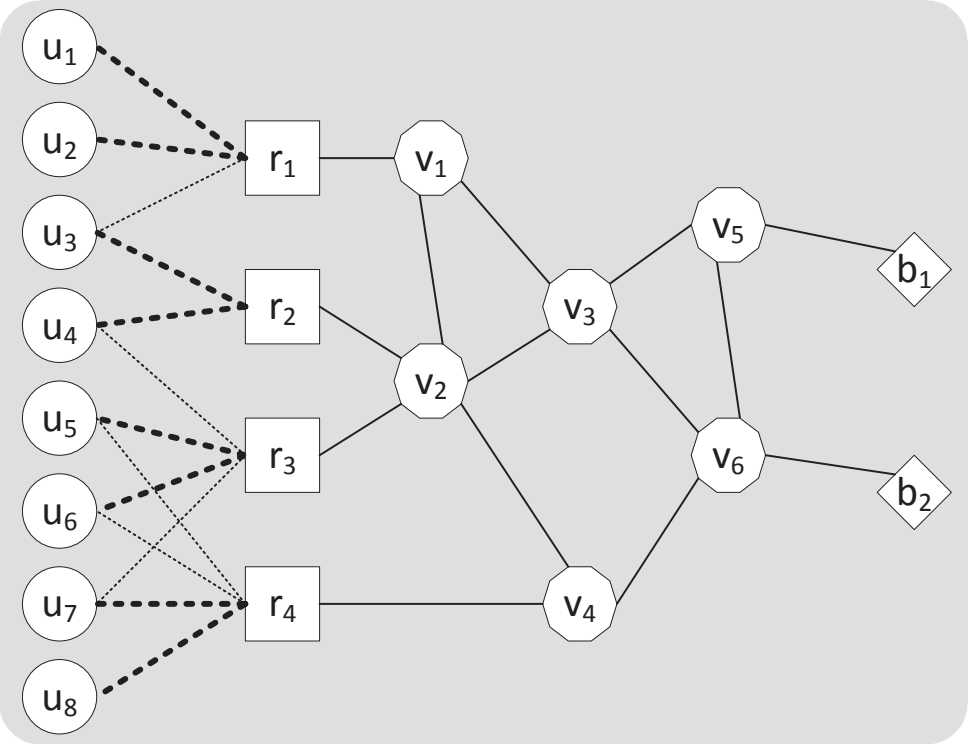


Figure 4.5: Considered C-RAN topology

We can see that Splits A, B and C have the same performance due to no ICI is considered here (cf. Table 4.6). First of all, the formulation of a large cluster can achieve a better spectral efficiency (see the first 4 ranked results all with 3 RRUS in c_1). Moreover, a large cluster that can covers all users is also even desired (see rank 1 and 2) due to more signal power are included within the cluster. Furthermore, via comparing the results of ranked 6 and 7, we can see that moving a user (u_4 in this case) into a larger cluster might benefits via sacrificing the performance of other users (i.e., $\{u_1, u_2, u_3, u_4\}$) but obtaining gains from another group of users (i.e., $\{u_5, u_6, u_7, u_8\}$). Last but not least, the approach that forms RRU cluster only bases on the number of included user is not the good option. For instance, a cluster that can cover 7 users is only ranked 15, while the clusters that can evenly serve 4 users is ranked 7. Such phenomenon is due to the loss of not including the desired signal power form the user toward its anchor RRU (i.e., from u_3 to r_2, from u_5 to r_3 and from u_6 to r_3).

4.6 Discussions

In this chapter, we extensively survey the impacts of the considered five factors on the network spectral efficiency in more details. As the next steps, following possible directions are highlighted.

Based on the considered five design issues, one potential extension is to investigate which

Table 4.7: Simulation results in network example

Rank	Spectral efficiency	RRU clustering	User association	Functional split
1	49.02	$c_1 = \{r_1, r_3, r_4\}$ $c_2 = \{r_2\}$	$U_{c_1} = \mathcal{U}$ $U_{c_2} = \varnothing$	$F_{c_1} = \{\text{A,B,C}\}$ $F_{c_2} = \varnothing$
2	46.21	$c_1 = \{r_1, r_2, r_4\}$ $c_2 = \{r_3\}$	$U_{c_1} = \mathcal{U}$ $U_{c_2} = \varnothing$	$F_{c_1} = \{\text{A,B,C}\}$ $F_{c_2} = \varnothing$
3	40.05	$c_1 = \{r_1, r_2, r_3\}$ $c_2 = \{r_4\}$	$U_{c_1} = \mathcal{U} \setminus \{u_8\}$ $U_{c_2} = \{u_8\}$	$F_{c_1} = \{\text{A,B,C}\}$ $F_{c_2} = \{\text{A,B,C,D,E}\}$
4	38.72	$c_1 = \{r_2, r_3, r_4\}$ $c_2 = \{r_1\}$	$U_{c_1} = \mathcal{U} \setminus \{u_1, u_2\}$ $U_{c_2} = \{u_1, u_2\}$	$F_{c_1} = \{\text{C}\}$ $F_{c_2} = \{\text{A,B,C,D,E}\}$
6	33.58	$c_1 = \{r_1, r_2\}$ $c_2 = \{r_3, r_4\}$	$U_{c_1} = \{u_1, u_2, u_3\}$ $U_{c_2} = \mathcal{U} \setminus \{u_1, u_2, u_3\}$	$F_{c_1} = \{\text{D}\}$ $F_{c_2} = \{\text{A, B, C}\}$
7	33.33	$c_1 = \{r_1, r_2\}$ $c_2 = \{r_3, r_4\}$	$U_{c_1} = \{u_1, u_2, u_3, u_4\}$ $U_{c_2} = \{u_5, u_6, u_7, u_8\}$	$N_{c_1} = \{\text{A,B,C}\}$ $F_{c_2} = \{\text{A,B,C}\}$
15	30.94	$c_1 = \{r_1, r_4\}$ $c_2 = \{r_2, r_3\}$	$U_{c_1} = \mathcal{U} \setminus \{u_4\}$ $U_{c_2} = \{u_4\}$	$F_{c_1} = \{\text{A,B,C}\}$ $F_{c_2} = \{\text{C,D}\}$

subset of them are needed when deploying the C-RAN for different services. For instance, the infrastructures of BBU and FH may be shared between other tenants; as a results, we can only optimize the network performance in terms of RRU clustering, user association, and functional split. Another case is to fix the RRU clustering and BBU anchoring decisions due to their coarse time granularity, compared with other three variables. Via considering a subset of these factors, some other solutions can be utilized with a decreased complexity.

Additionally, one extension approach is to investigate the impacts of modifying these five factors in terms of service continuity. We can notice that the changing of these variables shall not introduce any catastrophic effect (e.g., radio link failure) on providing a consistent service experience. Among these five factors, we can observe that the modifications of RRU clustering and BBU anchoring may have such discontinuous side-effect. To this end, further works are needed toward this direction.

4.7 Conclusions

In this chapter, we detailedly elaborate five critical design issues to deliver the E2E RAN service: (1) user association, (2) RRU clustering, (3) functional split, (4) FH routing, and (5) BBU anchoring. To maximize the overall network spectral efficiency, we formulate the optimization problem and introduce different coordinated processing schemes for different functional splits. However, due to its ultra high time complexity, we revisit the problem and provide an efficient solution. Due to the time limitation, several extensive simulations are still undergoing; therefore, we will provide a full results in the final manuscript.

Chapter 5

RAN Runtime Slicing System for Flexible and Dynamic Service Execution

5.1 Introduction

As detailed in Chapter 1, 5G mobile network is a paradigm shift beyond the new radio and new and wider spectrum with the objective of improving the overall network efficiency and flexibility. In is also the evolution of computing for wireless networks (e.g., central offices become data centers) and delivering networks on an *as-a-service* basis, when compared to the "one-size-fits-all" 4G approach. Support of vertical industries [205] is one of the main driving factors behind this evolution to empower the business and value creation for 5G. The underlying idea being to support multiple services and/or virtual networks on a common physical network with different service requirements in terms of service definition and agreement, network control and management, and also user performance.

Through the aforementioned 5G vision, naturally the network infrastructure providers (e.g., operators and data center owners), service providers (e.g., over-the-top [OTT] and verticals), and network function/application providers (e.g., vendors) are decoupled to allow a cost-effective network composition and sharing model to reduce both CAPEX and OPEX. Figure 5.1 illustrates the relationship between different providers and the transformation of the value-chain in the telecommunication industry being extended from the high-level role models presented by 3GPP in [206]. For example, network infrastructure may be provided by the operator as an intermediary between the vendors and data center owners or by a combination of network equipments from vendors, data centers from information technology (IT) industry, and transport network from operators. Furthermore, a blend of CPU, graphics processing unit (GPU), network processing unit (NPU), field-programmable gate array (FPGA) can be leveraged by the data centers to provide a sophisticated and customized computing environment tailored to the 5G service requirements (e.g., ultra-reliability and low-latency). A service is built through the composition of multi-vendor PNFs/VNFs, that not only shall meet the requirements of service providers such as performance and cost but also thats of network infrastructure providers in terms of PNF/VNF interoperability and compatibility when the service is running on different underlying infrastructures.

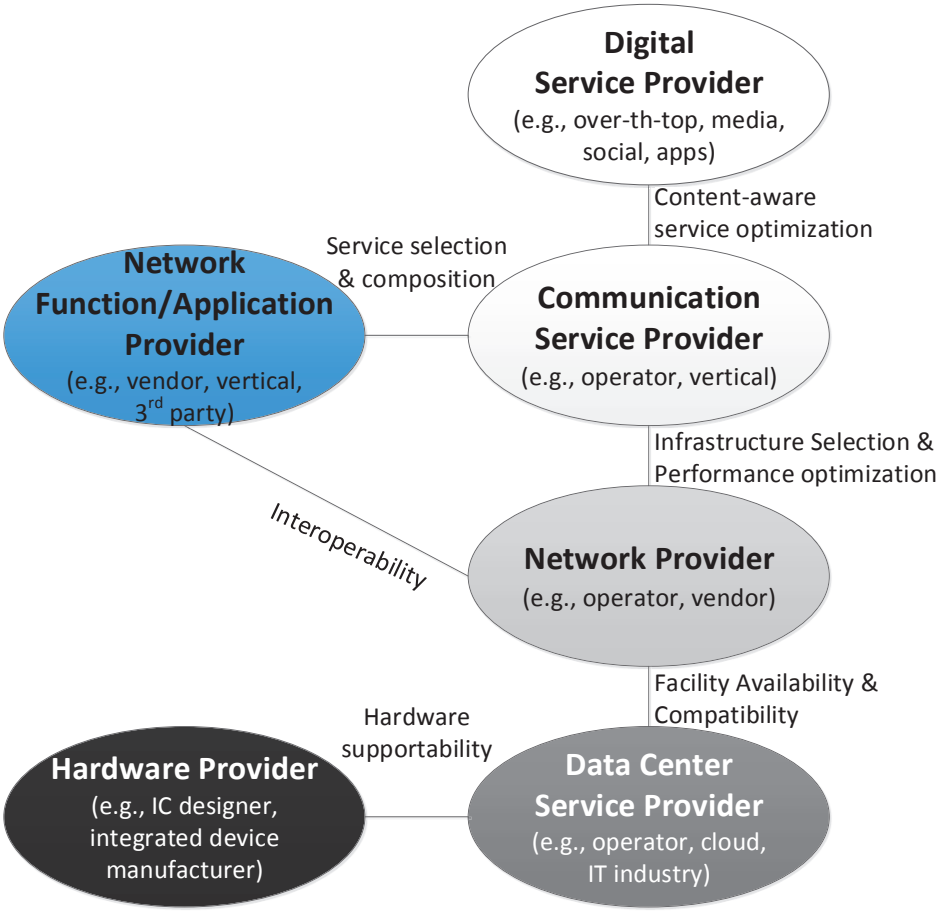


Figure 5.1: 5G vision in the evolution of telecommunication industry

Network slicing is one of the key enablers to flexibly deliver networks on an as-a-service basis. It enables the composition and deployment of multiple logical networks over a shared physical infrastructure, and their delivery as a service or slice. A slice can either be completely *isolated* from other slices down to the different sets of spectrum and cell site (as in most of current 3G and 4G deployment), or be *shared* across all types of resources including radio spectrum and network functions (e.g., all network layers of protocol stack), or be *customized* for a subset of UP and CP processing with an access to a portion of radio resources in a *virtualized/physical* form. To enable these options, a flexible execution environment is needed to host slice service instance over the network resources provided by the underlying infrastructures. Hence, different levels of *isolation* and *sharing* across the *domain-specific resources* spanned by a slice shall be naturally supported. Note that the concept of different isolation levels not only provides the dedicated resources but also brings different service chains and potentially different execution environment.

Further, domain boundaries could be administrative (e.g., between operators), network segment (e.g., RAN, CN, TN), radio access technology (e.g., 4G, 5G) among the others, and resources could be of different types including computing, storage, network, hardware, radio, spectrum, network functions and applications. For instance, a slice can be composed of dedi-

cated CN with isolated control functions, while using the virtualized radio resource in a shared spectrum. However, another slice can be composed of fully isolated resources (except computing ones) and network functions like the FLARE solution provided in [207]. To this end, *softwarization*, *virtualization*, and *disaggregation* are the key slicing enablers to flexibly customize a slice, automate its life-cycle management, and ease the development of network functions and applications with the objective to accommodate the E2E service requirements.

Several standardization bodies and industry forums outline the crucial role of an E2E service architecture to fulfill the *service-oriented* visions of 5G as already stated in Chapter 1, e.g., ITU [110], 3GPP [208], NGMN alliance [111] and GSMA [112]. Many architectures and prototypes have been proposed for CN slicing [209–211] and RAN slicing [212,213]. The challenges for CN slicing has been also addressed by 3GPP, and realized through the dedicated core network (DECOR) [214] and evolved DECOR (eDECOR) [215] as 5G core network (5GC). Nevertheless, RAN slicing remains a challenge in *providing different levels of isolation and sharing to allow a slice owner to customize its service across UP, CP, and CL while increasing the resource utilization of RAN infrastructure*. The CL refers to the logic that makes the decisions for a particular CP/UP function, e.g., CL decides on user handover and CP performs the corresponding handover action following the standardized protocol stack. Note that the RAN slicing is different from the legacy RAN sharing notion in that the focus is only on the efficient sharing on cell sites, passive (e.g., antenna mast) and active (e.g., TN infrastructure) network elements, radio spectrum, network function and application, and baseband processing.

Based on the four technology enablers of RAN slicing mentioned in Chapter 1, several works study the RAN slicing vision. The RadioVisor proposed in [216] can isolate the control channel messages, elementary resources such as CPU and radio resource to provide the customized service for each slice. A fully isolation solution as FLARE is provided in [207] with different virtual BSs representing different slices; however, there is no benefits in terms of the radio resource allocation multiplexing and network function sharing. The Hap-SliceR radio resource slicing framework proposed in [217] considers resource utilization and slice utility requirements; however, its main focus is on the resource customization for haptic communication. The authors of [218] propose a RAN slicing architecture among infrastructure and spectrum for both public safety and commercial services via resource virtualization and dedicated control functions. In [219], the radio resource scheduling of a BS is separated into the intra-slice scheduler and inter-slice scheduler; however, the resource abstraction/virtualization is not included and only a portion of functions are isolated. In [212], a RAN slicing architecture is proposed allowing radio resource management (RRM) policies to be enforced at the level of PRBs through providing the virtualized resource blocks (vRBs) by a novel resource visor toward each slice; however, it neither considers function isolation nor resource customization/abstraction per slice request. In [220], different approaches to split radio resources are compared in terms of the resource granularity and the degrees of isolation and customization; nonetheless, the resource multiplexing capability among slices is not considered. The Orion solution provided in [213] introduces the BS hypervisor to simultaneously isolate slice-specific control logics and share the radio resources. Moreover, the underlying PRBs are grouped into vRBs to be provided only to the corresponding slice. Such work exploits the prerequisites of function isolation and resource virtualization, while it does not consider customization of CP/UP functions in both monolithic and disaggregated RANs. In [221], the proposed RAN slicing framework can base on the service descriptions to flexibly share RAN functions over different network layers; however, it only considers physical resource partitioning without resource virtualization and multiplexing.

Table 5.1: RAN slicing state-of-the-arts comparison

Authors (Year)	Solution level	Radio resource	CP function	UP function
Nikaein *et al.* [158] (2015)	Network level	-	Dedicated	Dedicated
Kokku *et al.* [121] (2012)	BS level	Physical or virtualized resource sharing	-	-
Mahindra *et al.* [124] (2013)	Gateway and BS levels	Physical or virtualized resource sharing	-	-
Kokku *et al.* [125] (2013)	Gateway level	Virtualized resource sharing	-	-
He *et al.* [126] (2015)	Gateway level	App-oriented virtualized resource sharing	-	-
Aijaz [217] (2017)	Gateway level	Learning-based virtualized resource sharing	-	-
Zaki *et al.* [155] (2017)	BS level	Physical resource sharing	Dedicated	Dedicated
Foukas *et al.* [154] (2016)	BS level	Physical or virtualized resource sharing	Shared	Shared
Gudipati *et al.* [216] (2014)	BS level	Physical 3D resource sharing	Dedicated	Dedicated till programmable radio
Nakao *et al.* [207] (2017)	BS level	Dedicated spectrum allocation	Dedicated	Dedicated
Marabissi and Fantacci [218] (2017)	BS level	Virtualized resource sharing	Dedicated	Dedicated
Sallent *et al.* [220] (2017)	BS level	Physical resource sharing	Split into tenant-specific and common	Shared
Rost *et al.* [219] (2017)	BS level	Physical resource sharing	Split into cell and user-specific	Dedicated till real-time RLC
Ksentini and Nikaein [212] (2017)	BS level	Flexible resource sharing	Dedicated	Shared
Foukas *et al.* [213] (2017)	BS level	Virtualized resource sharing	Split into cell and user-specific	Dedicated till PHY
Ferrús *et al.* [221] (2018)	BS level	Physical resource sharing	Dedicated	Dedicated or shared till PHY

Table 5.1 summarizes the solution level and compares several related works in three aspects: (a) radio resource allocation model, (b) control plane function, and (c) user plane function. In order to serve various flavors of slice, the flexibility and effectiveness of these three aspects shall be achieved simultaneously through a unified RAN slicing solution. To this end, our proposed RAN runtime slicing system aims to flexibly support various slice requirements (e.g., isolation) and elastically improve multiplexing gains (e.g., sharing) in terms of (1) the new set of radio resource abstractions, (2) network service composition and customization for modularized RAN, and (3) flexibility and adaptability to different RAN deployment scenarios ranging from monolithic to disaggregated.

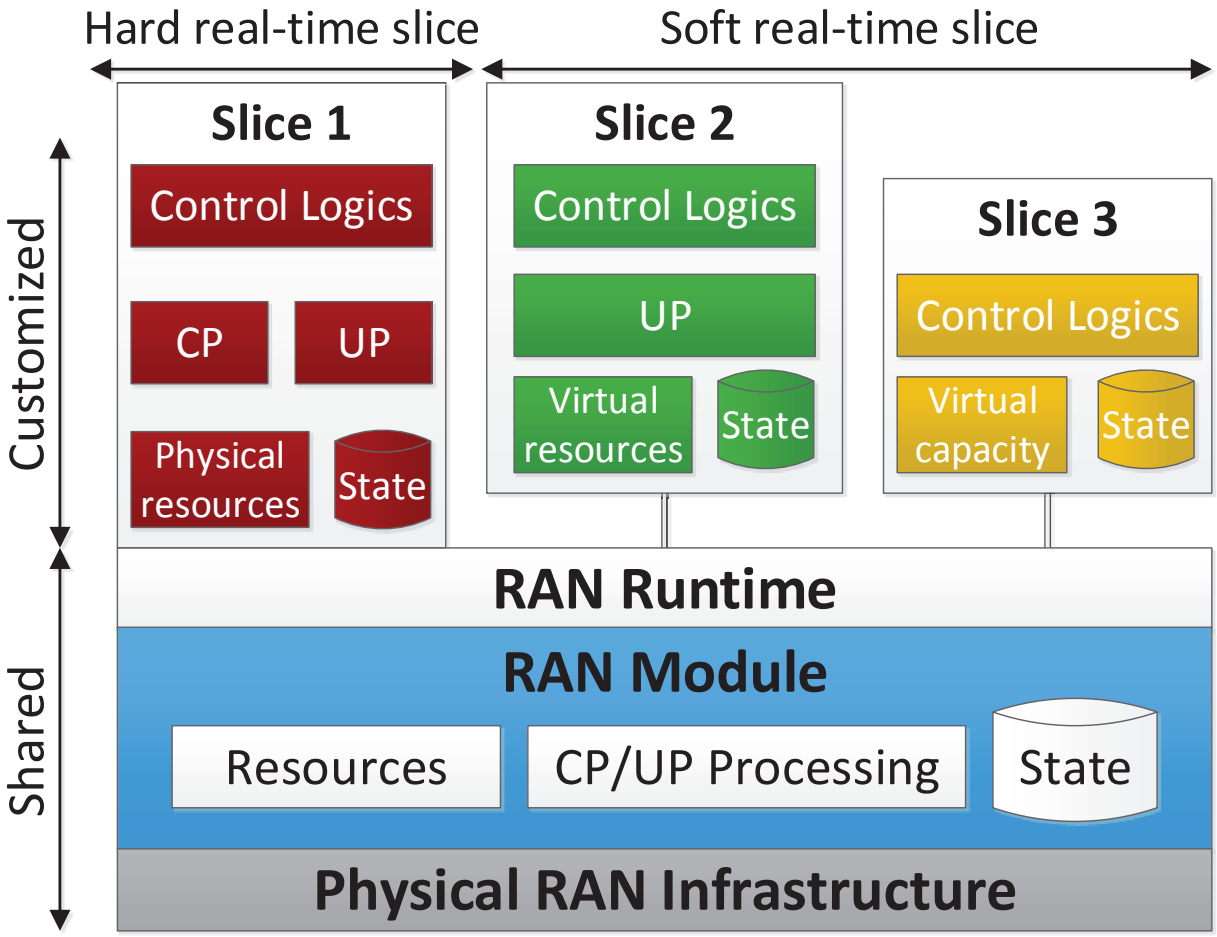


Figure 5.2: High-level architecture of RAN runtime slicing system

5.2 RAN runtime slicing system

The proposed RAN runtime slicing system provides a flexible *execution environment* to run multiple virtualized RAN instances with the requested levels of isolation and sharing of the underlying RAN modules and resources. It allows the slice owners to (a) create and manage their slices, (b) perform their customized CLs (e.g., handover decision) and/or customized UP/CP processing (e.g., PDCP and RLC functions), and (c) operate on a set of virtualized resources (e.g., resource block or frequency spectrum) or performance indicators (e.g., rate) and access to their CP/UP state (e.g., user identity) that are revealed by the RAN runtime. The isolation and customization properties provided by the RAN runtime is in favor of the slice owners allowing them to control the slice compositions and the behaviors of the underlying RAN module as per service requirements, while the sharing is in favor of the infrastructure provider that enables the efficient and dynamic multiplexing among multiple tenants over resources, processing, and states of the shared RAN module. The RAN module refers to a unit that comprises a subset of RAN functions and performs a portion or all of RAN processing.

The proposed RAN runtime slicing system is shown in Figure 5.2, with the RAN runtime

being the core component by which each running slice can interact with the RAN modules to access resources and state, and control the underlying RAN behavior. From the slice owner perspective, the RAN runtime provides an execution environment through which a slice can perform the customized processing, request the resources, and access the states. At the same time, it enables infrastructure provider to manage the underlying RAN module, enforce the slice-specific policies, and perform the access and admission control. The RAN runtime by itself is in charge of managing the life-cycle of instantiated slices, abstracting the radio resources and states, and applying changes into the underlying RAN module to customize each slice. It also implements a set of RAN runtime APIs for the bidirectional interactions between each slice and the underlay RAN module in order to monitor or control the CP/UP processing, resources, and states, while retaining the isolation among slices.

A slice is formally represented to the RAN runtime by a slice descriptor that defines the slice service requirements in terms of the resources, customized processing, and the performance. It is generally provided by the service orchestrator during the creation or update of a slice, and indicates for each slice how radio resources are allocated, reserved, preempted, or shared, how the CP/UP processing is pipelined, and what are the expected throughput and/or latency. The customization feature provided by the RAN runtime allows a slice owner to only contain a portion of resources and processing within the slice boundary and to multiplex the remaining ones in the underlying RAN module. To realize a flexible trade-off between the isolation and sharing, the states of CP and UP processing are maintained in a database[1] allowing to update the processing pipeline (e.g., from the customized one to the multiplexed one or vice versa) on-the-fly, while retaining the service continuity and isolation on the input/output data streams. By maintaining the state, the network functions are virtually turned into the stateless processing which allows to update the service and to recover the state through the RAN runtime.

In addition, the overall CP processing of a BS is logically separated into the slice-specific functions and the BS-common ones to exploit the function sharing benefits. Note that the CP processing is separated in terms of their functionalities. For instance, the master information block (MIB) and system information blocks (SIBs) are broadcasted commonly to all users with in a cell coverage and are categorized into the BS-common one, while the random access process may be customized by each slice to reduce the latency generated by the BS-common random access procedure. Moreover, the CLs of each slice can be developed/deployed independently tailored to the service requirement. For example, the handover control decisions can be programmed to improve slice-specific QoE and the RAN runtime will provide a feasible policy toward the underlying RAN module.

A monolithic RAN example with three slice instances is depicted in Figure 5.3 over the RAN runtime. For slice 1, both CP and UP processing are separated into the customized (radio resource control [RRC], service data adaptation protocol [SDAP] RLC, MAC layers) and the shared ones (Physical layer [PHY]), while slice 2 only customizes its SDAP function for UP processing. In contrast, slice 3 relies on the shared CP/UP processing without any customization. This CP/UP processing composition for each slice is facilitated by the RAN runtime. Moreover, the control logics of each slice can be programmed in a customized manner, e.g., handover (slice 1), RRM (slice 1 and slice 2) and monitoring (all three slices). Through the RAN runtime, these customized control logics will be elaborated and their conflicts will be resolved that will be further detailed in the next section.

[1]This is regardless of whether the network function is stateful or stateless [222].

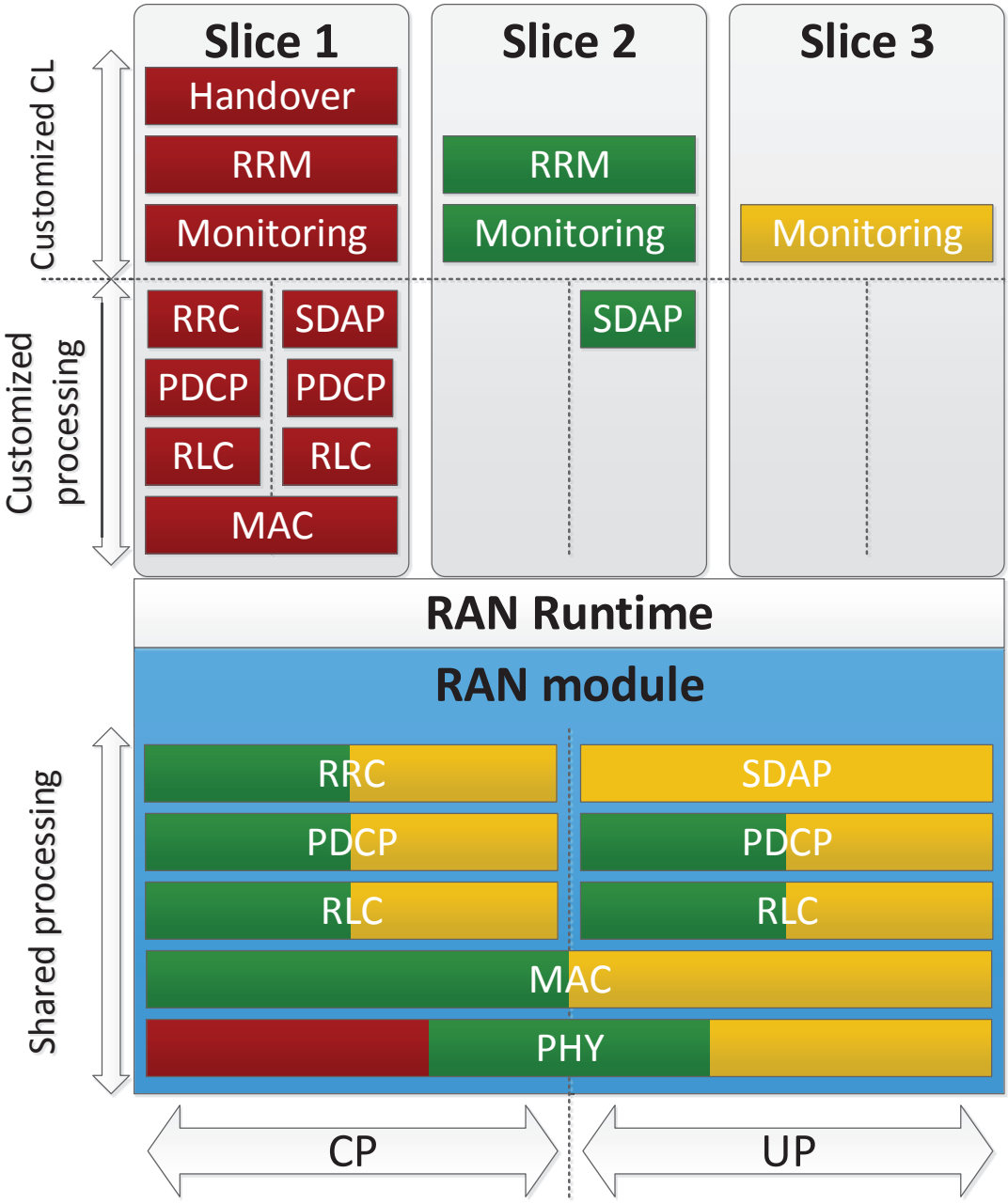


Figure 5.3: A monolithic RAN example with three instantiated slices over the RAN runtime

In summary, in the proposed RAN runtime slicing system, RAN functions are pipelined to compose the desired RAN module, i.e., monolithic or disaggregated RAN instances, either via multiplexed or customized CP/UP functions and CLs as per slice requirement. The RAN runtime acts as the intermediate between the customized slices and the underlying shared RAN module and infrastructure providing a unified execution environment with substantial flexibility to achieve the required levels of isolation and sharing.

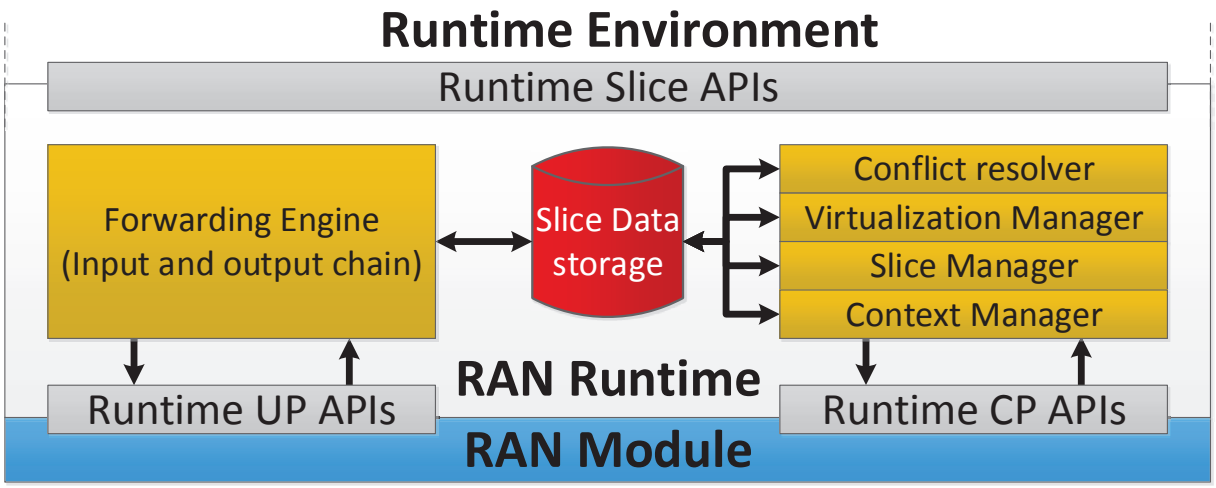


Figure 5.4: Architecture of the RAN runtime slicing system

5.3 Design elements of RAN runtime

Based on the overview of the RAN runtime slicing system in the previous section, we hereby introduce the design challenges and provide more details on the main components of the RAN runtime slicing system in this section, namely the slice data, RAN runtime services, and RAN runtime APIs.

5.3.1 Design challenge

Before examining the components of the RAN runtime, we first highlight the challenges that shall be resolved the RAN runtime as following:

- Allow each slice to interact with the underlying RAN module and change the CP/UP behaviors that are dynamically determined during its execution subject to the access control (Section 5.3.2 and Section 5.3.3).
- Provide different levels of isolation and sharing to allow a slice owner to flexibly compose the slice-specific RAN resources and processing from the multiplexed or customized resources and CP/UP functions, respectively. Note that the multiplexing gain is also considered for the underlying radio resources and RAN modules (Section 5.3.3).
- Provide the APIs to enable the slice-specific CP, UP and control decisions to be realized for both soft and hard real-time requirements (Section 5.3.4).

Figure 5.4 illustrates the three main building blocks of the RAN runtime: (a) slice data storage, (b) CP and UP functions to provide the RAN runtime services, and (c) RAN runtime API, that are described in the following paragraphs.

5.3.2 Slice data storage

Slice data is the entity that stores both *slice context* and *RAN module context* under the control of the context manager within the RAN runtime. They are used to customize and manage a

Table 5.2: Slice context maintained by the RAN runtime

Slice Context	Descriptions
Slice identity	Represents a unique slice identifier
Service registry identity	Identifies to which RAN runtime services a slice is e.g., slice manager, context manager, virtualization manager, conflict resolver, and forwarding engine
Slice SLA and policy	Describes a business agreement between the slice owner and the infrastructure provider in terms of performance, resource, access control, and priority level of the slice
Customized processing	Specifies the customized CP/UP processing functions of the slice. If not specified, the default processing is applied to this slice.
User context	Identifies which pair of BSs and slices a user belongs to and the mapping between traffic flow and dedicated radio bearers (DRBs)[2]

slice in terms of the required RAN runtime services, resources, processing, states, and users.

The *slice context* describes the basic information and prerequisites to instantiate a slice service and to manage corresponding users. It is provided by the service orchestrator and may be updated by the corresponding slice (cf. Figure 5.2) following the agreement between the slice owner and the infrastructure provider. Table 5.2 describes the slice context information maintained by the RAN runtime in the slice data.

The *module context* includes the CP and UP state information (belongs to the slice owners), module life-cycle management primitives such as start, configure, and stop service (belongs to the network function/application provider), and resources (belongs to the infrastructure provider). Unlike input or output data streams of the RAN module that can be pipelined, the control and data state are maintained separately by the RAN runtime and revealed to each slice in real-time to allow the efficient and isolated slice-specific processing. In addition, such state may be shared among multiple slices subject to the access control, for instance, when coordinated processing and/or decision making are required in the case of the handover decision of a user belonging to two or more slices. Note that in general case, states only include the user-specific functions in both RRC CONNECTED mode and RRC INACTICE-CONNECTED [223] mode, and not necessarily the BS-common functions that are executed independently from the number of instantiated slice, i.e., even with no instantiated slices or when operating in the RRC IDLE mode (cf. Table 5.3).

5.3.3 RAN runtime services

In the following paragraphs, we elaborate on five RAN runtime service that can be provisioned for each slice as shown in Figure 5.4. To utilize these RAN runtime services, each slice is registered and identified with its identity over the RAN runtime among disaggregated RAN entities.

[2]The 1:n:m relation of user-to-slice-to-BS mapping will make use of RAN runtime CP APIs for network slice selection operation.

Table 5.3: BS-common and user-specific control plane functions

Process	BS-common functions	User-specific functions
Location tracking and paging	Tracking area update, CN paging	RAN Paging
Handover and cell re-selection	Cell (re-)selection criterion	Measurement configuration, handover processing
Random access	Common random access	Dedicated random access
User attach procedures	-	Slice-based user association control
QoS maintenance and admission control	-	QoS flow maintenance and slice-based admission control
Security function	Common BS key management	User/Slice-specific CP/UP key management
Bearer management	Signaling radio bearer procedure	Default/Dedicated radio bearer procedure
Radio resource allocation	Common BS signals such as cell-specific reference signal and synchronization signals	Per-slice dedicated resource partitioning and accommodation
System Information	Broadcast non-access stratum, MIB, and SIB information	-

5.3.3.1 Context manager

This service manages both slice context and module context by performing the CRUD[3] operation on the slice data. To create a slice context, the context manager firstly performs the slice admission control based on the provided network service descriptor (NSD) that defines the required processing, resources, and states (as agreed between the slice owner and the infrastructure provider). Upon the slice admission control, module context is used by the context manager to register slice-specific life-cycle primitives to the slice manager and the requested resources and/or performance to the virtualization manager. The former allows the customized CP/UP processing to be applied on the input/output data streams, while the latter enables the resource partitioning and abstraction to be performed among multiple slices. At this stage, a slice can start to consume the RAN runtime services not only to manage its service but also to interact with the underlying RAN module through the RAN runtime CP/UP APIs. Then, the context manager can handle the real-time CP/UP state information within the slices and the underlying RAN module so as to keep the slice data in-sync with the instantaneous state.

Note that many slices can be deployed at a single RAN runtime following the multi-tenancy approach to enable scalable service deployment. However, the maximum number of slices that can be deployed depends on (1) the overhead of the RAN runtime, (2) the available resource in terms of compute, memory and networking link, (3) the requested SLA and resource by each slice, (4) the resource over-provision percentage, and (5) the workload of each slice.

[3]CRUD includes four basic operations: create, read, update, and delete.

5.3.3.2 Slice manager

The slice manager entity is responsible for managing the life-cycle of a slice when instructed by the slice owner or by the service orchestrator. Through the slice manager, slice life-cycle operations can be triggered, which in turn enables both slice owner and infrastructure provider to control and update slice service definition as per need and agreement. Based on the service definition and slice context, the slice manager determines the CP/UP processing chain for each slice and each traffic flow, and programs the forwarding engine through a set of rules allowing to direct the input and output streams across the multiplexed processing operated by underlying RAN module and the customized processing performed by the slice. Unlike the context manager that handles the local slice context, the slice manager operates on an E2E RAN service in support of service continuity when the slice service definition is updated. For example, a slice owner that performs the customized UP processing can opt in for the multiplexed pipelined processing to reduce its OPEX, which causes changes in its slice service definition. In addition, when the slice requirements are violated (e.g., performance degradation), the slice manager may change the number of requested resources, resource allocation type, resource partitioning period, or even update the service definition to comply with the service requirements.

Slice manager is also in charge of taking a set of actions when detecting any conflicts among multiple slices based on a set of policy rules. Such conflict can happen at the level of slice when service definition is changed or at the level of user when it is associated to multiple slices (e.g., 1:n or m:n user-slice relationships). For instance, reserving the resources and/or changing the resource allocation type of a slice may violate the performance of another slice that requires a high bandwidth. Another example is when different user measurement events are requested by different slices which will require the coordination to reconfigure the measurement with the largest common parameters and the least denominator. To this end, slice manager relies on a set of policy rules defined by the infrastructure provider to decide whether to preempt one slice, reject another slice, or multiplex the requests.

5.3.3.3 Virtualization manager

This service is in charge of providing the required levels of isolation and sharing to each slice as depicted in Figure 5.5. It *partitions* on resources and states based on the slice and module contexts, *abstracts* the physical resources and states to/from the virtualized ones, and reveals the *virtual* views to a slice that is customized and decoupled from the exact physical resources and states. Afterwards, the *intra-slice operation* is done on this virtual view by each slice instance. Finally, multiple intra-slice scheduling and allocation results will be *accommodated* and *mapped* to the physical resources and states. In the following, we focus on the resource aspect and omit state partitioning and abstraction as they can be realized through some well-known approaches such as database partitioning and control access.

Inter-slice resource partitioning Resource partitioning is an update process that happens in varying allocation window of T [121, 213]. It allows to distribute resources among multiple slices based on the resource requirements expressed in the slice context that is stored in the slice data and their aggregated workload. Radio resource descriptor has three elements: (1) *resources type* defines whether the requested resources are of type physical/virtual radio

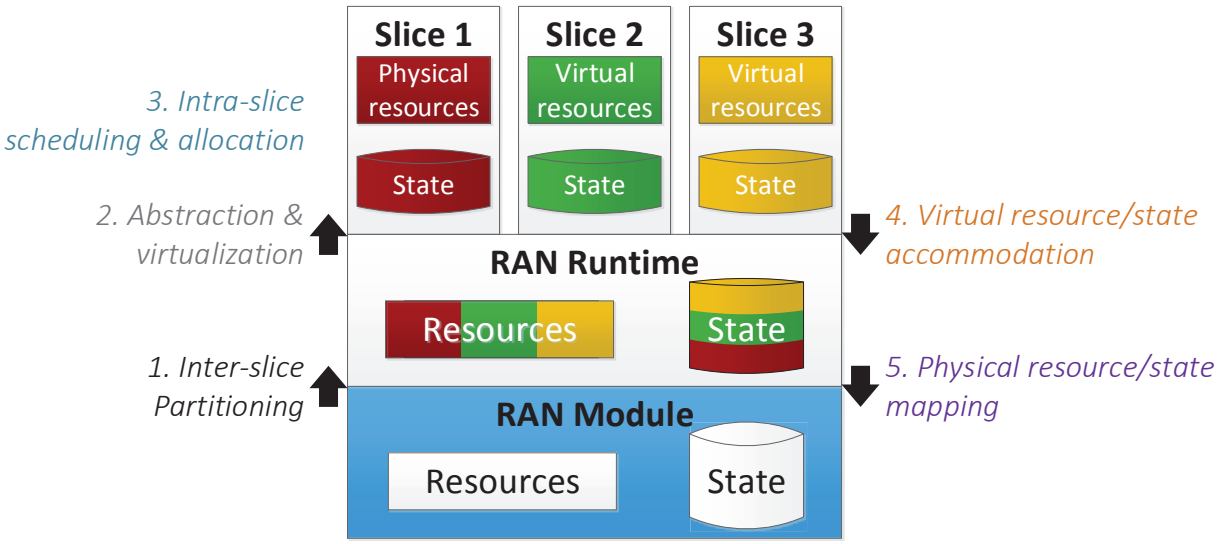


Figure 5.5: Steps of virtualization manager

resources in time and frequency domains[4], or performance indicators in data rate or latency, (2) *abstraction type* that maps physical radio resource allocation types, namely fixed position, contiguous, non-contiguous, or minimum resource block groups (RBGs), to the virtual RBGs (vRBG) or virtual transport block size (vTBS), and (3) *resource structure* that contains the applicable frame structure numerologies in time and frequency domains. Specifically, different numerologies in terms of the TTI and the sub-carrier spacing (SCS) can be applied depending on the deployed frequency band and/or maximum user mobility in order to mitigate the impacts of wireless channel non-idealities (e.g., Doppler shift due to the user mobility) for each slice service [224]. For instance, only one type of SCS, i.e., 15 kHZ, is applied in LTE system, while there are five applicable SCSs, i.e., 15, 30, 60, 120, and 240 kHz, defined by 3GPP in [225] with their corresponding frame structures.

Besides aforementioned radio resource requirements provided by the slice owner, the resource allocation shall also respect the policy defined by the infrastructure provider, for instance, the allowable resource allocation types of underlying RATs. Take the DL resource allocation of LTE system for instance, there are three types of resource allocation: (i) Type 0 allocation is based on the minimum granularity as RBG that comprises multiple RBs, (ii) Type 1 categorizes RBGs into different subsets and only allocates RBs within the same subsets, and (iii) Type 2 allocates contiguous vRBs that can be physically contiguous (localized vRB) or non-contiguous (distributed vRB). For UL, there are two resource allocation types: (a) UL type 0 allocates physical RBG (PRBG) in a contiguous manner, and (b) UL type 1 allocates non-contiguous RBGs within two distinct clusters. Then, four abstraction types are introduced with RBG as the minimum resource granularity, and their respective mapping to the DL/UL resource allocation types are shown in Table 5.4. Moreover, via leveraging the vBS typs 0 in Table 5.4, different types of resource can be abstracted, e.g., virtual capacity or virtual latency, to match service requirements in terms of minimum required data rate or maximum allowed latency. Even

[4]It can be extended to the dimensions of component carrier and antenna.

Table 5.4: Mapping between resource abstraction type and allocation type

Requested resources	Abstraction types (Resource granularity)	DL Resource allocation type	UL Resource allocation type
Resource block	vRBG Type 0 (Non-contiguous)	Type 0, Type 1, Type 2 distributed	Type 1
	vRBG Type 1 (Contiguous)	Type 0, Type 2 localized	Type 0
	vRBG Type 2 (Fixed position allocation)	Type 2 localized	Type 0
Performance indicator	vTBS Type 0 (RBGs with minimum granularity)	All types	All Types

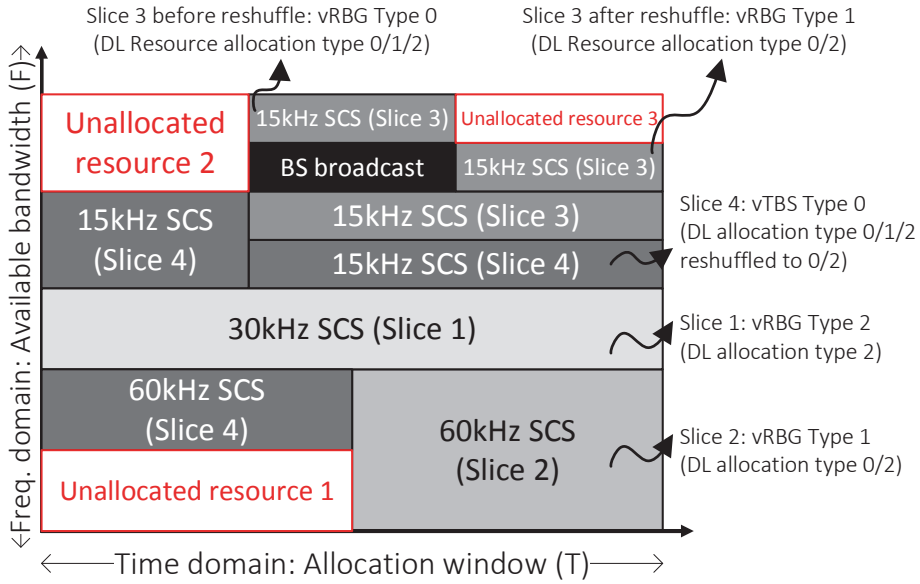
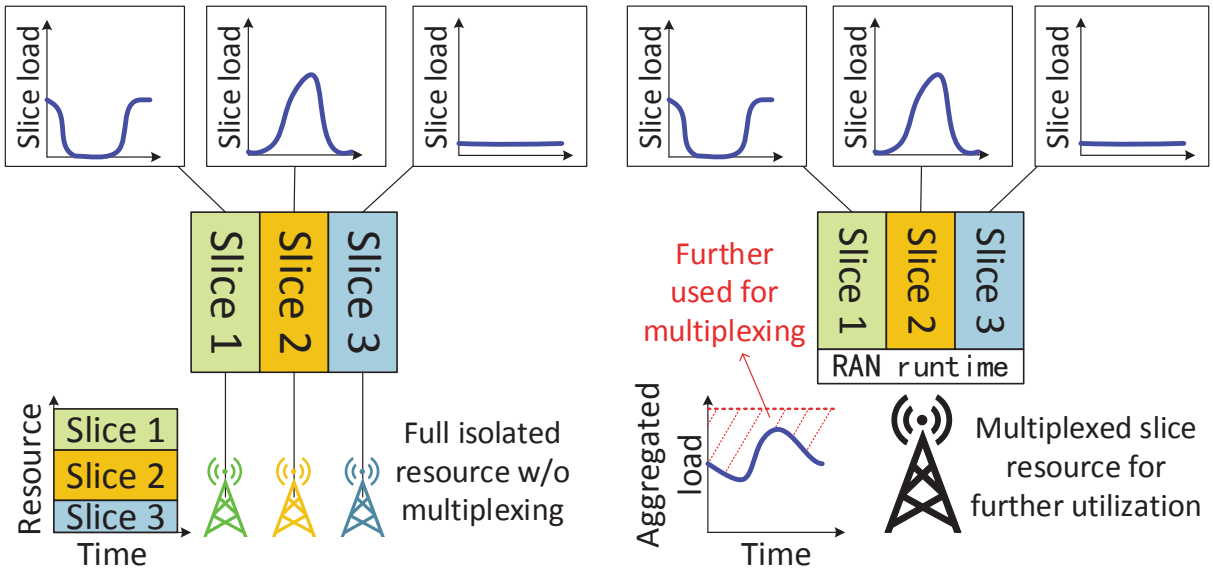


Figure 5.6: Resource partitioning with different resource abstraction types

though both virtual capacity and virtual latency can be mapped to any resource allocation types (i.e., DL type 0/1/2 and UL type 0/1 of LTE system); however, only the latter one have higher a higher priority and a shorter allocation window due to its latency requirements. In summary, the proposed vRBG and vTBS form a superset of legacy resource allocation types, providing the required flexibility for both inter-slice resource partitioning and accommodation as detailed in Section 5.4.

Figure 5.6 then illustrates an example of resource partitioning among four slices over one allocation window T with different types of abstraction. The proposed resource abstraction scheme allows to dynamically change the mapping between different resource allocation types, for instance, changing from allocation type 0/1/2 to allocation type 0/2 for both slices 3 and 4. Such resource reshuffle can increase the flexibility of resource allocation for the infrastructure provider and maintain the resource abstraction requested by the slice owner.



(a) Without resource multiplexing **(b)** With resource multiplexing

Figure 5.7: Multiplexing of slice resources

Radio resource abstraction Based on the aforementioned inter-slice resource partitioning, all available radio resources can be fragmented following different abstraction types. Such resource abstraction serves for two purposes: (1) isolate resources by presenting a virtual view of the resources that is decoupled from the exact physical locations, and (2) increase multiplexing gain by adjusting allocation types and sharing the unused resources. The former simplifies the inter-slice resource partitioning operation and prevents other slices to access or even infer the resources allocated to others (in favor of slice owner), and the latter allows to increase the resource utilization efficiency (in favor of infrastructure provider). Specifically, if slice resources are not multiplexed, we can observe in Figure 5.7a that no other slices can utilize the unallocated resources even the slice load variates from time to time. In contrast, in Figure 5.7b, the unallocated resources due to the time-varying load can be multiplexed to deploy more services at a single RAN infrastructure. Such multiplexing gain across tenants is anticipated by the infrastructure provider when deploying scalable numbers of slices.

Take the 3 MHz case of LTE system as an example in Figure 5.8a, where the number of total PRB is 15 and the PRBG granularity is 2 PRBs, giving a total of 8 PRBGs and the last PRBG only contains 1 PRB. These PRBGs are firstly partitioned for each slice based on the number of required resources and then they are abstracted according to the abstraction types mentioned in Table 5.4, i.e., fixed, contiguous, non-contiguous, minimum granularity. Afterwards, the resulted PRBGs, vRBGs and vTBSs are provided to each slice for the intra-slice resource scheduling. For instance, fixed position resources is requested by slice 1 and hence no virtualization is performed (i.e., PRBG). Whereas slice 4 requests a number of capacity as its performance indicator, and thus its PRBGs are abstracted into vTBS with the capacity value computed from the measured channel state information. The PRBGs of slice 2 and slice 3 are virtualized into vRBGs via abstracting the exact frequency/time locations as well as other dimensions (e.g.,

carrier frequency and antenna index) and are pooled together to maintain the relative frequency dependencies among the virtualized resources without revealing any absolute physical relations. Take the slice 3 that uses resource allocation type 0 as an example, only PRBGs within the same subset can be scheduled at the same time. In that sense, vRBGs are *pooled* in order to indicate such exclusive condition between vRBG pool 1 (i.e., PRBG0, PRBG6) and vRBG pool 2 (i.e., PRBG 5), and thus the intra-slice resource scheduler of slice 3 will allocate resources to each user from either vRBG pool 1 or vRBG pool 2.

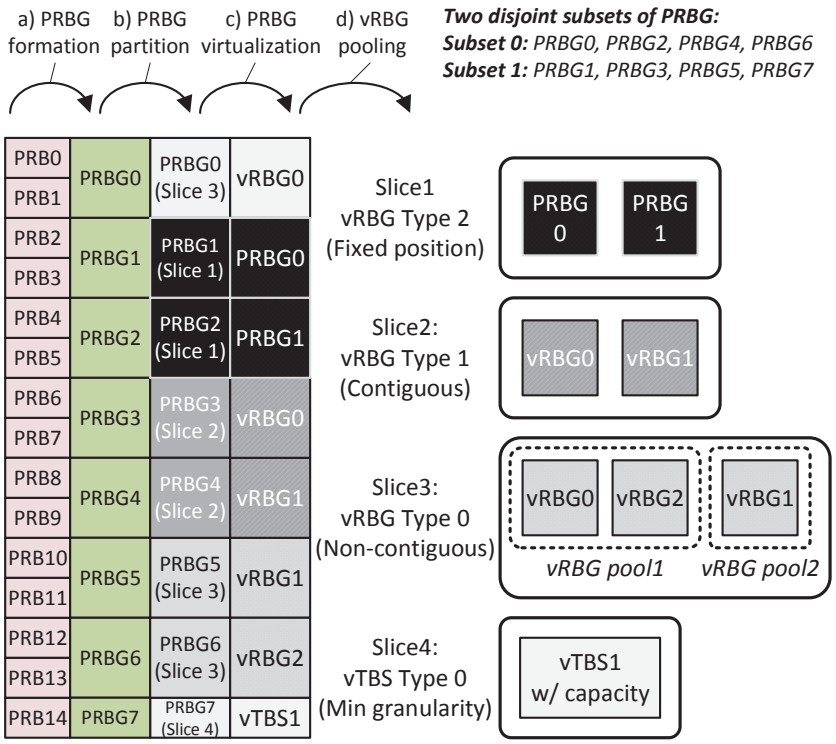
Radio resource accommodation and multiplexing After the radio resource partitioning and abstraction, each slice can perform the intra-slice resource scheduling to its associated users and the scheduling decisions will be accommodated into PRBs as shown in Figure 5.8b. Such accommodation does not necessary follow the partitioned resources of the inter-slice resource partitioning (cf. Figure 5.8a) to allow a better utilization of available resources. For instance, the vRBG1 for both slice 2 and slice 3 are accommodated to their vRBG0 in the partitioning stage respectively so as to have a larger contiguous unallocated region (i.e., PRBG4 to PRBG6) that can be shared to other slices. The unallocated resource can be shared to other slices (e.g., vTBS2 of slice 4) that request more resources[5] or to some new services. Moreover, the preemption mechanism can also be applied by reallocating the inter-slice scheduling decisions of other low-priority slices to boost the perceived performance of high-priority slices[6]. Finally, the RAN runtime will allocate the corresponding control channel elements (CCEs) to transport the DL/UL control information (CI) based on the aforementioned DL/UL resource allocation types in the last step (cf. step h in Figure 5.8b). These CIs are used to indicate the user about the positions of allocated PRB as well as the necessary physical layer information (e.g., modulation and coding scheme, new data indication) for successful user data reception or transmission. With a limited control region to accommodate CCEs, the RAN runtime can also leverage the unallocated resources to carry these CIs.

5.3.3.4 Conflict resolver

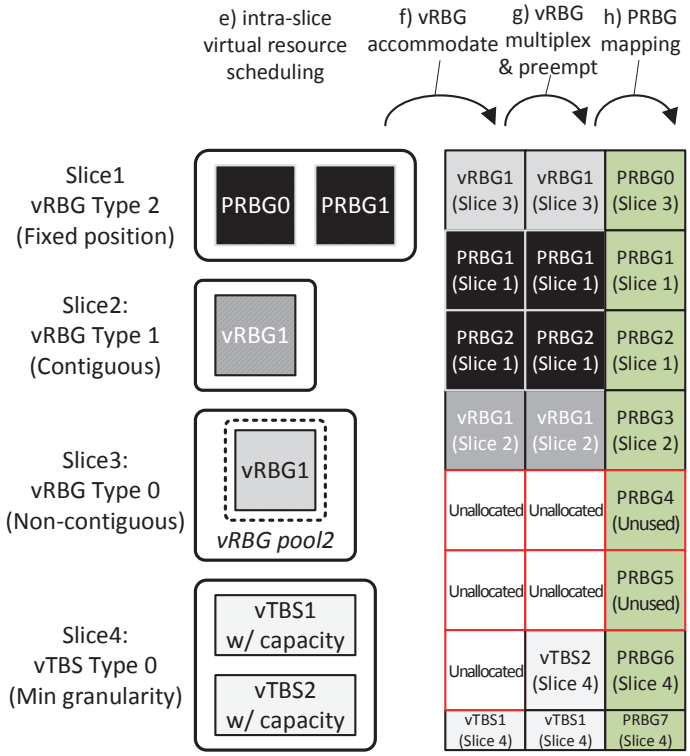
The conflict resolver provides a shared control logics for multiple slices. It aims to accommodate the customized control logics from different slice-specific control applications, resolve their conflicts, and enforce a feasible policy to underlying RAN module. For instance, the control logics of two customized RRM applications of slice 1 and slice 2 in Figure 5.3 will leverage the inter-slice conflict resolution and to provide a unified control logic for underlying RAN module. Moreover, the control logics of different control applications may also introduce conflict, e.g., both load balancing and handover optimization applications may simultaneously adjust the same handover parameter for their different purposes.

To resolve the conflict, there are some possible methods. First, the policy-based method can be used to apply the provided policy from the slice manager to decide which control application to be executed. However, the drawback of policy-based methods is that they may suffer from high variance in the quality of results [226]. Another approach is through the learning-based method to generate the predictive model from historical data to optimize multiple control application objectives simultaneously.

[5]Such multiplexing may not be allowed by slices requesting fixed position.
[6]The preemption characteristic shall be described in the slice context.



(a) Stages for building vRBG and vRBG pool



(b) Stages for accommodating vRBG to PRBG

Figure 5.8: Different stages for virtualized radio resources slicing

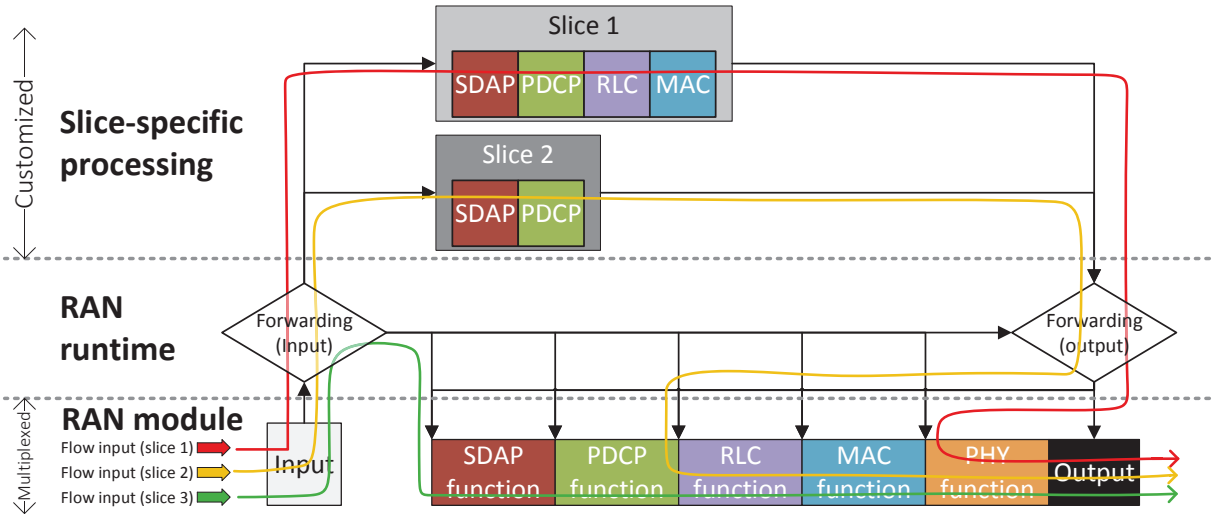


Figure 5.9: Forwarding engine and UP forwarding path

5.3.3.5 Forwarding engine

The forwarding engine manages the input and output streams of CP and UP, or simply data streams, between RAN and users across multiplexed and/or customized processing. Figure 5.9 shows an example of how the forwarding engine manages the UP processing chain in the DL direction (i.e., from RAN to user) across several network layers: service data adaptation protocol (SDAP), PDCP, RLC, MAC, and PHY[7]. Input flows of the RAN module for each slice are forwarded either to the customized (i.e., slice 1 and slice 2) or the multiplexed (i.e., slice 3) processing chain based on the rules applied by the slice manager. After the first stage of processing, the output flows are further forwarded to the corresponding entry points in the multiplexed chain (i.e., slice 2) or the output endpoint (i.e., slice 1). Note that more complex forwarding rules can be applied if required, for instance, the customized MAC function to manage the intra-slice scheduling while multiplexing other functions. Further, the per-flow customization within a slice can be applied in order to differentiate the customized processing for each flow with corresponding QoS requirement. Such forwarding engine can leverage the match-action abstraction following SDN principles to establish the input/output forwarding path between the RAN runtime and slices in both directions [227].

Furthermore, the forwarding engine is able to direct data not only in a monolithic RAN but also in a disaggregated RAN, where a single RAN module is decomposed into CU, DU, and RU as mentioned in Chapter 2. Note that in the proposed RAN slicing model, RAN disaggregation and functional splits are controlled and maintained by the infrastructure provider, whereas the RAN service customization is managed by the slice owner. Figure 5.10 shows the input/output forwarding path between CU, DU, and RU to compose a distributed UP processing chain using 3GPP function split option 2 between CU and DU and option 6 between DU and RU. The input and output endpoints of each RAN module will perform the infrastructure-dependent packet processing like encapsulation , sub-header including, and switching/routing

[7]Further decompositions within each layer are possible like splitting the PHY into high-PHY and low-PHY.

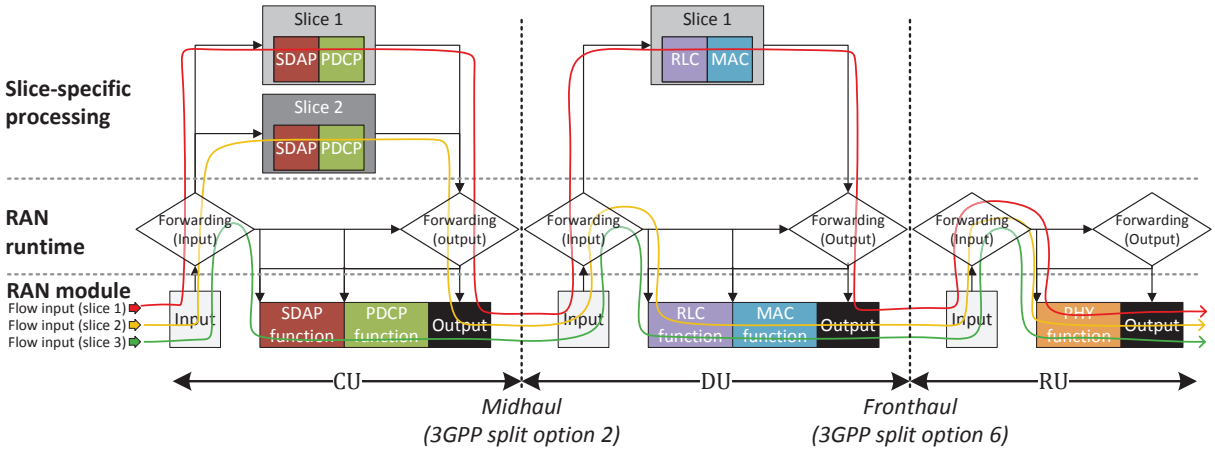


Figure 5.10: UP forwarding path in three-tier disaggregated RAN (CU, DU, RU)

for the fronthaul/midhaul transportation which is transparent for the slice owner[8]. Moreover, when adopting the flexible function split and placement [228], the CP/UP state information has to be efficiently shared among disaggregated RANs to flexibly deploy and chain functions in between. Table 5.5 indicatively summarizes the main UP state information that shall be maintained and shared in the slice data. Also note that these aforementioned chains are applied for the DL direction, while the same forwarding engine can be utilized for the UL direction with different chain compositions.

5.3.4 RAN runtime APIs

The RAN runtime APIs are exposed both in the north-bound toward each slice and in the south-bound toward the underlying RAN module, allowing to manage a slice and control the underlying RAN module (cf. Figure 5.4). In the north-bound, the RAN runtime slice APIs provides interfaces and communication channels to connect a slice to the RAN runtime as a separate process, whether it is local or remote. Hence, each slice can be executed in isolation from each other either at the host or guest level leveraging the well-know OS and virtualization technologies, such as container or virtual machine. Such north-bound APIs allow the slice owner to register and consume the aforementioned RAN runtime services, manage its service in coordination with the RAN runtime and service orchestrator, and customize the CP/UP processing. In the south-bound, the RAN runtime CP/UP APIs enable a slice to take the control of its service by requesting virtualized/physical resources, applying its control decisions, and accessing the virtualized state information. When a slice is deployed locally, the RAN runtime APIs may exploit the inter-process communication mechanism to allow a slice to perform the real-time operation (e.g., MAC scheduling function) with hard guarantees (cf. slice 1 in Figure 5.2). Remote slices, on the other hands, communicate with the RAN runtime through the asynchronous communication interface and can perform the non-time-critical operation (e.g., PDCP function) like slice 2 and 3 in Figure 5.2.

[8]It can be customized for each service but needs agreements between slice owner and infrastructure provider.

Table 5.5: UP network functions and the decoupled states

Layer	Network function	Network state
PHY	RF processing	Carrier frequency, Spectrum bandwidth
	DFT/IDFT	Point of DFT, Output indexes
	Multi-antenna processing	Transmission mode, Beamforming matrix
	(De-)Modulation	Modulation order, Reference symbol information
	Bit-rate processing	Information of coding, scrambling, rate matching, and cycle redundancy check
MAC	HARQ process	HARQ index, User identity, Redundancy version
	(De-)Multiplexing	(De-)Mutltiplexed logic channel identities
	Dynamic scheduling and priority handling	Priorities between logic channels and users
RLC	ARQ error correction	Status report parameters, Polling information
	Segmentation and reassemble	Size of corresponding protocol data unit and service data units
	SDU discard	Discard criterion, e.g., window information
PDCP	Header (de-)compression	Header compression profile, state and parameter
	Integrity protection and verification	Integrity protection algorithm and related key parameters
	(De-)Ciphering	Ciphering algorithm and parameters
	Reordering and duplicate detection	Sequence number of queued PDUs
SDAP	Mapping between QoS flows and DRBs	QoS flow identity, QoS profile, mapping policy
	Marking QoS flow identity	QoS flow identity

5.3.5 Summary

In summary, the five proposed RAN runtime services can provide different levels of isolation and sharing and the correlated message flows between them are depicted in Figure 5.11. Note that these message flows between the RAN runtime services and the slice data storage are omitted for simplicity. These message flows can be utilized together with other known mobile network messages of different interfaces to provide a complete set of slice-specific processing, e.g., the customized handover process between BSs using the X2 interface tailored to the slice service requirements. Further, they can be utilized for the service orchestration and management purpose to orchestrate virtualized infrastructures and VNFs for newly-instantiated services. They can also be utilized by European telecommunications standards institute (ETSI) management and orchestration (MANO) architectural framework to collect functional blocks, data repositories and related interfaces.

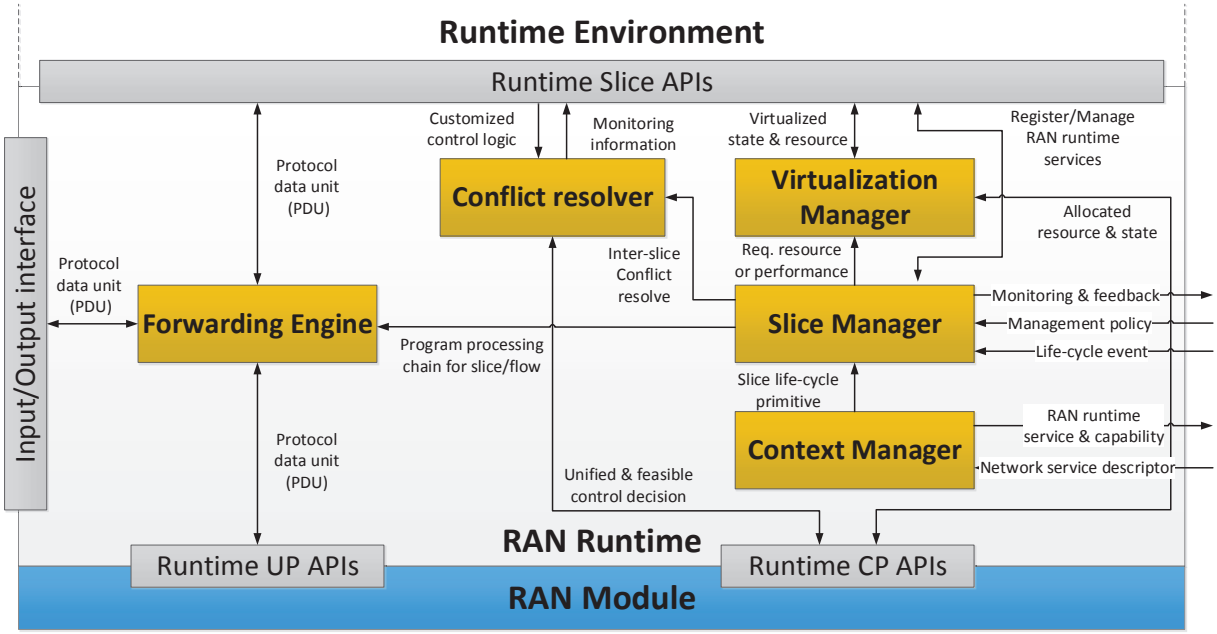


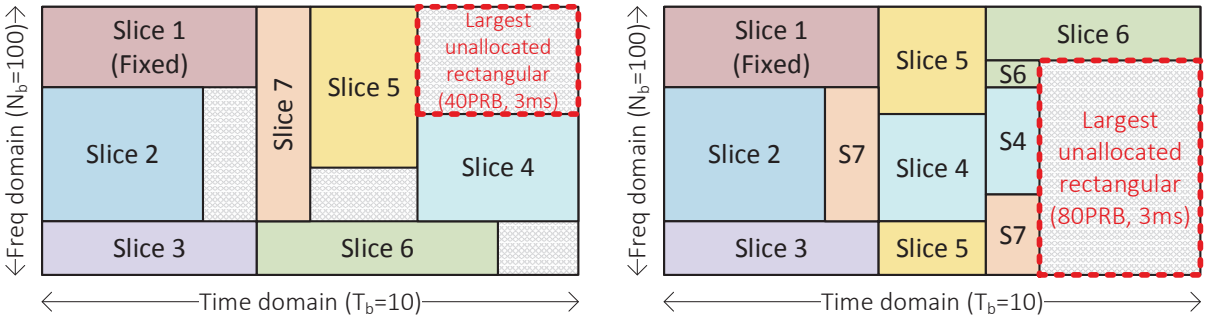
Figure 5.11: Message flows between RAN runtime services

5.4 Resource partitioning and accommodation

In this section, we focus on the inter-slice radio resource partitioning and accommodation, as the intra-slice resource scheduling can utilize several known scheduling algorithms, such as proportional fair or round robin, configured by the slice orchestrator [212] to provide the slice-specific customization. Specifically, we provide the algorithm of the inter-slice partitioning and accommodation, evaluate its performance, and formulate the overall multiplexing gain.

5.4.1 Inter-slice resource partitioning

The radio resources are partitioned by the RAN runtime periodically within an allocation window T (in ms) in the time domain and F (in Hz) in the frequency domain. These resources can be specifically quantized into a resource grid map Map with T_b TTIs in time domain and N_b PRBs in the frequency domain with respect to the base SCS (SCS_b) used by the infrastructure provider, e.g., a 20 MHz LTE radio bandwidth in a 10 ms allocation window is quantized into 100 PRBs in the frequency domain and 10 TTIs in the time domain. There are $|S|$ slices requesting the radio resources $\mathcal{S} = \{s_1, \cdots, s_{|S|}\}$. For the k-th slice (i.e., s_k), its radio resource requirements include: (a) SCS_k set comprises the applicable SCSs, (b) T_k and N_k are the number of requested resource in time (ms) and frequency (Hz) domain respectively, and (c) g_k is the granularity which can be contiguous, non-contiguous, fixed position (with its fixed starting position denoted as FN_k and FT_k in frequency and time domain) or minimum granularity (with its request data rate as R_k in bps) as mentioned in Table 5.4. The fixed position granularity inherently isolates resources as its partitioned resources are physical ones without any virtualization. The contiguous one is more suitable for quasi-constant traffic patterns (e.g., streaming) since it can reduce the latency and minimize the CI signaling overhead. The non-contiguous one, on the other hand,



(a) Contiguous granularity **(b)** Slice-specific granularity

Figure 5.12: Examples of radio resource partitioning

accommodates better for variable traffic patterns as it can allocate fragmented resources. The minimum granularity can be utilized by those slices that request only capacity (i.e., vTBS), which allowing for all feasible partitioning.

A resource partitioning example is depicted in Figure 5.12 with 7 slices (i.e., $|S| = 7$). Each slice has different resource granularities: $g_1 = \text{FixPos}$, $g_2 = g_3 = \text{Con}$, $g_4 = g_5 = \text{NonCon}$, and $g_6 = g_7 = \text{Min}$. Note that the largest rectangular of the unallocated resource is highlighted, which is an important criterion for further resource multiplexing. Since such largest unallocated rectangular region may potentially fit in any data transportation numerology in time (i.e., TTI) and frequency (i.e., SCS) domains, and can be either shared by different slices or utilized commonly for CI transportation and BS broadcasting. Additionally, such radio resource defragmentation can provide a better slice performance in terms of delay and throughput. It is observed from Figure 5.12a and Figure 5.12b that although both resource partitions can satisfy the requested resources among all seven slices, while only the latter one can achieve a larger unallocated rectangular region. Such compact resource packing in Figure 5.12b utilizes different resource granularities, i.e., s_4 and s_5 can be discontinuous in frequency and time separately, and s_6 and s_7 can leverage the minimum granularity.

5.4.1.1 Problem formulation

Based on the aforementioned observations, the inter-slice resource partition has two complementary goals: (a) satisfy as many slice resource requests as possible, and (b) maximize the size of unallocated rectangular region for further multiplexing. In following, we introduce the control variables for the problem formulation: (1) $m_{i,j,k} \in \mathcal{M}$ is binary, indicating whether the resource substrate $i \in \{1, \cdots, T_b\}$ and $j \in \{1, \cdots, N_b\}$ in time and frequency domain is partitioned for the k-th slice (i.e., 1) or not (i.e., 0), (2) $b_{i,j,k} \in \mathcal{B}$ is binary, representing whether the resource substrate $i \in \{1, \cdots, T_b\}$ and $j \in \{1, \cdots, N_b\}$ is the unique beginning position for the k-th slice (i.e., 1) or not (i.e. 0), (3) $a_{i,k} \in \mathcal{A}$ is binary, using by the non-contiguous slices (i.e., $g_k = \text{NonCon}$) to indicate whether the i-th TTI is partitioned for the k-th slice (i.e., 1) or not (i.e., 0), and (4) $c_k \in \mathcal{C}$ is the applied SCS for the k-th slice. Based on such applied SCS for the k-th slice, the input resource request (i.e., T_k, N_k) can be further computed correspondingly as $T_k^{c_k}$ and $N_k^{c_k}$. The overall resource partitioning problem is formulated as follows.

$$\max_{\mathcal{M},\mathcal{B},\mathcal{A},\mathcal{C}} \sum_{i=1}^{T_b}\sum_{j=1}^{N_b}\sum_{k=1}^{|\mathcal{S}|} b_{i,j,k} + w \cdot \text{MaxUn}\left(\mathcal{M}\right) \tag{5.1a}$$

$$s.t. \sum_{k=1}^{|\mathcal{S}|} m_{i,j,k} \leq 1,\ \forall i,j \tag{5.1b}$$

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} b_{i,j,k} \leq \begin{cases} b_{FT_k^{c_k},FN_k^{c_k},k}, & \text{if } g_k = \text{Fix} \\ 1, & \text{else} \end{cases}, \forall k \tag{5.1c}$$

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} \left(m_{i,j,k} - T_k^{c_k} \cdot N_k^{c_k} \cdot b_{i,j,k}\right) = 0,\ \forall k \tag{5.1d}$$

$$\sum_{p=i}^{i+T_k^{c_k}-1}\ \sum_{q=j}^{j+N_k^{c_k}-1} m_{p,q,k} \geq b_{i,j,k} \cdot N_k^{c_k} \cdot T_k^{c_k}, \forall i,j,g_k = \text{Fix}||\text{Con} \tag{5.1e}$$

$$\sum_{j=1}^{N_b} m_{i,j,k} = N_k^{c_k} \cdot a_{i,k},\ \forall i, g_k = \text{NonCon} \tag{5.1f}$$

$$\left(\sum_{j=1}^{N_b} b_{i,j,k}\right)\left(\sum_{p=i}^{T_b} a_{p,k} - T_k^{c_k}\right) = 0,\ \forall i, g_k = \text{NonCon} \tag{5.1g}$$

$$\left[N_k^{c_k}, T_k^{c_k}\right] = \text{SCSMap}\left(N_k, T_k, R_k, CSI_k, c_k, SCS_b\right),\ \forall k \tag{5.1h}$$

$$\left[FN_k^{c_k}, FT_k^{c_k}\right] = \text{SCSMap}\left(FN_k, FT_k, 0, 0, c_k, SCS_b\right), \forall k \tag{5.1i}$$

$$m_{i,j,k} \in \{0,1\}, b_{i,j,k} \in \{0,1\}, a_{i,k} \in \{0,1\}, c_k \in \text{SCS}_k \tag{5.1j}$$

We can first observe the objective function of Eq. (5.1a) includes two distinguished goals. The first one indicates whether the unique beginning point can be found, while the second relies on the MaxUn $(\cdot)$ function to output the largest unallocated rectangular based on the control variable set $\mathcal{M}$. A weight $w > 0$ can balance these two goals. Eq. (5.1b) guarantees that each resource can be partitioned no more than one slice, while Eq. (5.1c) ensures at most one beginning point is indicated for each slice. Noted that the fixed position slices can only set $b_{FT_k,FN_k,k}$ to be 1. Eq. (5.1d) restricts up to $T_k^{c_k} \times N_k^{c_k}$ resource are partitioned for the k-th slice, and Eq. (5.1e) provides the dimensional constraints for fixed-position and contiguous granularity slices. Moreover, Eq. (5.1f) and Eq. (5.1g) are the corresponding dimensional constraints for non-contiguous granularity slices via utilizing $a_{i,k}$ to indicate $T_k^{c_k}$ TTI instances each with $N_k^{c_k}$ PRBs. For minimum granularity slices, there is no dimensional constraint. Finally, Eq. (5.1h) and Eq. (5.1i) use the SCSMap $(\cdot)$ function to compute the quantized resource request (i.e., $T_k^{c_k}$, $N_k^{c_k}$) and fixed position (i.e., $FT_k^{c_k}$, $FN_k^{c_k}$) based on the selected SCS c_k. Note that the quantized resource request for minimum granularity slice is derived based on the requested rate R_k and channel state information (CSI) CSI_k.

We notice that such problem can be mapped to the *NP-hard* two-dimensional knapsack problem [229] when all resource granularities are continuous, which makes the complexity to find the optimal solution be non-polynomial. Hence, finding the optimal inter-slice resource partition is expensive when the number of slice increases. Prior works provide the heuristic

algorithms [230, 231], but they only focus on one special case that considers a single SCS with contiguous granularity. We hereby propose the granularity-based heuristic algorithm that can sequentially partition radio resources as explained in the following paragraph.

5.4.1.2 Proposed algorithm

The overall proposed algorithm is presented in Algorithm 5.1 that sequentially prioritizes the k-th slices, i.e., s_k, based on the pre-defined prioritization policy (i.e., *priority*) and then partitions resources according to its granularity, i.e., g_k. As each slice can support more than one SCSs, the remapping operation from the base SCS (SCS_b) to another SCS (scs) is necessary for the number of requested resources (N_k^{scs}, T_k^{scs}) and fixed position (FN_k^{scs}, FT_k^{scs}) through the SCSMap $(\cdot)$ function shown in Algorithm 5.1. Note that the requested data rate R_k can be mapped to the number of requested radio resources using the per-slice aggregated CSI, i.e., CSI_k[9] as well as the corresponding MCS index. Moreover, we use a unified resource grid map for the granularity-based partitioning, i.e., $Map_{i,j}^{scs}$. Hence, the granularity-based partitioning algorithms include the ones for the fixed position (Algorithm 5.2), contiguous (Algorithm 5.3), non-contiguous (Algorithm 5.4) and minimum granularity (Algorithm 5.5). When a slice is satisfied (i.e., $Sat_k == 1$), a resource grid remapping through the $RGMap\,(\cdot)$ function in Algorithm 5.1 aims to map the resource grid from the selected SCS for the k-th slice (i.e., $SCS\,[k]$) to other SCSs. Further, the mapping from the unified resource grid map to the control variables $\mathcal{M}$ and $\mathcal{B}$ are done through $MMap\,(\cdot)$ and $BMap\,(\cdot)$ correspondingly. Finally, all satisfied slices after partitioning are included in the set $\mathcal{S}_p$.

When applying the fixed position algorithm (cf. Algorithm 5.2), the FindFRe $(\cdot)$ function checks the feasibility of the fixed position allocation (i.e., starts from FN_k^{scs} and FT_k^{scs} in frequency and time domain respectively) and outputs 1 when feasible (0 otherwise). While the FindRe $(\cdot)$ function is used in the contiguous algorithm (cf. Algorithm 5.3) and it outputs a set of 2-tuples comprising all possible contiguous positions in frequency and time domain, respectively, i.e., PN and PT. Specifically, PN set comprises first entry of the 2-tuple set, while PT set includes the second entry. Then, we pick the position with the largest unallocated rectangular using the aforementioned MaxUn $(\cdot)$ function over the unified resource grid map. In non-contiguous algorithm (cf. Algorithm 5.4), the FindUnRe $(\cdot)$ function outputs all available positions in a set of 2-tuples (i.e., PN set includes the first entries and PT set contains the second entries) without requiring a contiguous fragment. Then, we allocate sequentially in time domain following the decreasing order of available resources over the frequency domain using the sorting function sort $(\cdot)$ shown in Algorithm 5.4. Specifically, all possible time indexes (i.e., from 1 to $T_b \cdot \frac{scs}{SCS_b}$) are ranked based on the number of available frequency domain resource at that time index (i.e., aN). The minimum granularity algorithm of Algorithm 5.5 also applies the same FindUnRe $(\cdot)$ function to find all available positions and uses the InMaxRe $(\cdot)$ to check that these available positions are within the largest rectangular region (output 1 in In) or not (0 otherwise). Finally, all possible positions (i.e., indexed from 1 to $Size$) are sorted in the ascending order based on whether they are in the maximum rectangular or not (i.e., In) for later resource partitioning.

[9] It can base on the average CSI among its served users.

Algorithm 5.1: Inter-slice Resource Partition Algorithm

Input : T_b and N_b are resource grid size in time and frequency domains
$\mathcal{S}$ is the set of slices

Output: $\mathcal{M}$ is the set of slice grid map
$\mathcal{C}$ is the set of slice applied SCS
$\mathcal{B}$ is the set of slice beginning indicator
$\mathcal{S}_p$ is the set of satisfied slice

begin

- $\mathcal{S}_p = \varnothing$; /* Initialize the set of satisfied slice */
- **foreach** $s_k \in \mathcal{S}$ **do**
  - **for** $i = 1$ *to* T_b **do**
    - **for** $j = 1$ *to* N_b **do**
      - $b_{i,j,k} = 0$; /* Initialize the beginning indicator of slice s_k */
      - $m_{i,j,k} = 0$; /* Initialize the resource grid map of slice s_k */
  - $c_k = 0$; /* Initialize the selected SCS of slice s_k */
  - $Sat_k = 0$; /* Initialize satisfaction indicator of slice s_k */
  - **foreach** $scs \in SCS_k$ **do**
    - /* Map requested resource and fixed position to all SCSs */
    - $[N_k^{scs}, T_k^{scs}] = \text{SCSMap}\,(N_k, T_k, R_k, CSI_k, scs, SCS_b)$;
    - $[FN_k^{scs}, FT_k^{scs}] = \text{SCSMap}\,(FN_k, FT_k, 0, 0, scs, SCS_b)$;
- **foreach** $scs \in SCS$ **do**
  - **for** $i = 1$ **to** $N_b \cdot scs/SCS_b$ **do**
    - **for** $j = 1$ **to** $T_b \cdot SCS_b/scs$ **do**
      - $Map_{i,j}^{scs} = 0$; /* Reset unified resource grid map */
- **while** $isempty(\mathcal{S}) == false$ **do**
  - $s_k = \text{prioritize}\,(\mathcal{S}, priority)$; /* Get most prioritized slice s_k */
  - **switch** g_k **do**
    - **case** *Fix* **do**
      - $[Sat_k, c_k, Map] = \text{FixPos}\,(s_k, Map)$; (cf. Algorithm. 5.2)
    - **case** *Con* **do**
      - $[Sat_k, c_k, Map] = \text{Con}\,(s_k, Map)$; (cf. Algorithm 5.3)
    - **case** *NonCon* **do**
      - $[Sat_k, c_k, Map] = \text{NCon}\,(s_k, Map)$; (cf. Algorithm 5.4)
    - **case** *Min* **do**
      - $[Sat_k, c_k, Map] = \text{Min}\,(s_k, Map)$; (cf. Algorithm 5.5)
  - **if** $Sat_k == 1$ **then**
    - $Map = \text{RGMap}\,(Map, c_k)$; /* Remap the unified resource grid map for all SCSs */
    - $\mathcal{B} = \text{BMap}\,(Map, c_k)$; /* Map the unified resource grid map to beginning indicator */
    - $\mathcal{M} = \text{MMap}\,(Map, c_k)$; /* Map the unified resource grid map to slice grid map */
    - $\mathcal{S}_p = \text{SetUnion}\,(\mathcal{S}_p, s_k)$; /* Add slice into the satisfied slice set */
  - $\mathcal{S} = \text{SetDiff}\,(\mathcal{S}, s_k)$; /* Remove prioritized slice */

Algorithm 5.2: Fixed Position Resource Partition (FixPos)

Input : s_k is target slice
$IMap$ is the input unified resource grid map
Output: Sat is the slice satisfaction index
c_k is the selected SCS for target slice
$OMap$ is the output unified resource grid map

begin
  $MaxRec = 0$; /* Initialize the maximum unallocated rectangular */
  $Sat = 0$; /* Initialize the satisfaction index */
  foreach $scs \in SCS_k$ **do**
    if $FindFRe(N_k^{scs}, T_k^{scs}, scs, IMap^{scs}, FT_k^{scs}, FN_k^{scs}) == 1$ **then**
      $Sat = 1$; /* Current slice is satisfied*/
      $tMap = IMap^{scs}$;
      for $i = 0$ **to** $N_k^{scs} - 1$ **do**
        for $j = 0$ **to** $T_k^{scs} - 1$ **do**
          $tMap_{i+FN_k^{scs}, j+FT_k^{scs}} = k$;
      $tRec = \text{MaxUn}(tMap)$; /* Find the maximum unallocated rectangular */
      if $tRec > MaxRec$ **then**
        $c_k = scs$;
        $MaxRec = tRec$;
        $OMap^{scs} = tMap$;

5.4.1.3 Complexity analysis

The overall inter-slice resource partition algorithm of Algorithm 5.1 is composed of four granular-specific ones as explained beforehand from Algorithm 5.2 to Algorithm 5.5. In the following paragraphs, we analyze the complexity of each granular-specific algorithm and then summarize the overall inter-slice resource partitioning algorithm complexity.

In Algorithm 5.2, the most complex operation is to find the largest rectangular in the resource grid, i.e., MaxUn $(\cdot)$, for all available SCSs, and thus its complexity equals to $\mathcal{O}\left(|SCS| \cdot (N_b \times T_b)\right)$. In Algorithm 5.3, the complexity is proportional to the number of available SCSs, the size of possible locations (i.e., $|PN|$), and the operation to find the largest rectangular. In the worst case, $|PF|$ will equal to the size of resource grid; therefore, the complexity of Algorithm 5.3 is written as $\mathcal{O}\left(|SCS| \cdot (N_b \times T_b)^2\right)$. The complexity of Algorithm 5.4 depends on the operation of finding the largest rectangular as well as the sorting operation. The former is proportional to $(N_b \times T_b)$ as mentioned before, while the latter is proportional to T_b^2 in the worst case as there are T_b elements to be sorted. Thus, the complexity will be $\max\left(\mathcal{O}\left(|SCS| \cdot (N_b \times T_b)\right), \mathcal{O}\left(|SCS| \cdot T_b^2\right)\right)$. The most complex operation of Algorithm 5.5 is to sort all available positions, i.e., $|PN|$ elements As $|PN|$ in worst case will equal to the size of resource grid, i.e., $(N_b \times T_b)$; therefore, the complexity of such algorithm can be written as $\mathcal{O}\left(|SCS| \cdot (N_b \times T_b)^2\right)$.

Moreover, the complexity of the overall algorithm is proportional to (1) the number of slices, (2) the slice prioritization policy, and (3) the highest complexity among the aforementioned four algorithms. Note that only constant time will be spent when a pre-defined prioritization

Algorithm 5.3: Contiguous Resource Partition (Con)

Input : s_k is the target slice
$IMap$ is the input unified resource grid map
Output: Sat is the slice satisfaction index
c_k is the selected SCS for target slice
$OMap$ is the output unified resource grid map

begin
  $MaxRec = 0$; /* Initialize the maximum unallocated rectangular */
  $Sat = 0$; /* Initialize the satisfaction index */
  foreach $scs \in SCS_k$ **do**
    /* Find the possible positions (PN/PT) in frequency and time domains */
    $[PN, PT] = \text{FindRe}\,(N_k^{scs}, T_k^{scs}, scs, IMap^{scs})$;
    for $p = 1$ **to** $|PN|$ **do**
      $Sat = 1$; /* Current slice is satisfied*/
      $tMap = IMap^{scs}$;
      for $i = 0$ **to** $N_k^{scs} - 1$ **do**
        for $j = 0$ **to** $T_k^{scs} - 1$ **do**
          $tMap_{i+PN[p],j+PT[p]} = k$;
      $tRec = \text{MaxUn}\,(tMap)$; /* Find the maximum unallocated rectangular */
      if $tRec > MaxRec$ **then**
        $c_k = scs$;
        $MaxRec = tRec$;
        $OMap^{scs} = tMap$;

policy is applied (e.g., according to the SLAs)[10]. Thus, the overall complexity is written as $\mathcal{O}\left(|\mathcal{S}| \times |SCS| \times (N_b \times T_b)^2\right)$. Finally, as the number of SCS numerologies is limited, e.g., up to 5 options defined by 3GPP in [225], this complexity can be rewritten as $\mathcal{O}\left(|\mathcal{S}| \times (N_b \times T_b)^2\right)$,

5.4.1.4 Performance evaluation

As mentioned before, the resource partitioning is based on the prioritization policy (i.e., *priority* in Algorithm 5.1); hence, high priority slices will impact the available positions for low priority ones. In the following, five different prioritization policies are evaluated:

1. *Optimal*: Search all possible permutations to optimize the objective function in Eq. (5.1a).
2. *Random*: Randomize the slice ordering in each allocation window T.
3. *Greedy*: Greedily prioritize the slice that can generate the largest unallocated rectangular.
4. *Granularity*: Sort slices in the following order: Fix, Con NonCon, and Min.
5. *Granular & Greedy*: Use the two-sequential sorting, in which the first sort is based on *granularity* and the second is based on the *greedy* method.

[10]Extra complexity is required when adopting dynamic searching in the policy.

Algorithm 5.4: Non-contiguous Resource Partition (NCon)

Input : s_k is the target slice
$IMap$ is the input unified resource grid map
Output: Sat is the slice satisfaction index
c_k is the selected SCS for target slice
$OMap$ is the output unified resource grid map

begin
  $MaxRec = 0$; /* Initialize the maximum unallocated rectangular */
  $tIdxCount = 0$; /* Initialize the time index counter */
  $Sat = 0$; /* Initialize the satisfaction index */
  foreach $scs \in SCS_k$ **do**
    /* Find the unallocated resources position (PN/PT) in $IMap^{scs}$ */
    $[PN, PT] = \text{FindUnRe}\left(IMap^{scs}\right)$;
    for $j = 1$ *to* $T_b \cdot scs/SCS_b$ **do**
      $aN\left[j\right] = \text{find}\left(PT == j\right)$; /* Count available resources at time j */
      if $|aN\left[j\right]| \geq N_k^{scs}$ **then**
        $tCount = tCount + 1$; /* Increase the time index counter */
    if $tCount \geq T_k^{scs}$ **then**
      $Sat = 1$; /* Current slice is satisfied*/
      $tMap = IMap^{scs}$;
      /* Sort time indexes base on the descending order of aN */
      $Torder = \text{sort}\left(1 : T_b \cdot \frac{scs}{SCS_b}, aN, \text{'descend'}\right)$;
      for $j = 1$ *to* T_k^{scs} **do**
        $NIdx = PN\left[aN\left[Torder\left[j\right]\right]\right]$;
        for $i = 1$ **to** N_k^{scs} **do**
          $tMap_{i+NIdx[i],Torder[j]} = k$;
      $tRec = \text{MaxUn}\left(tMap\right)$; /* Find the maximum unallocated rectangular */
      if $tRec > MaxRec$ **then**
        $c_k = scs$;
        $MaxRec = tRec$;
        $OMap^{scs} = tMap$;

We evaluate our results using the self-built system-level MATLAB simulator. The results are shown in Figure 5.13 with 7 slices. Each slice can serve a number of users and it requests a time-varying uniformly-distributed aggregated resources with $N_k \sim \text{Uniform}\,(1.6, 9)$ MHz and $T_k \sim \text{Uniform}\,(1, 10)$ ms, $\forall s_k \in \mathcal{S}$. Note that the granularities of all seven slices are the same as the ones shown in Figure 5.12, and the applicable SCS set for the k-th slice is $SCS_k = \{15, 30, 60\}$ kHz, $\forall s_k \in \mathcal{S}$. As for the RAN infrastructure, the radio bandwidth is 20 MHz with $SCS_b = 15$ kHz and allocation window is 10 ms. Figure 5.13a then shows the slice satisfaction ratio for all seven slices or for each granularity type (i.e., fixed, contiguous, non-contiguous and minimum granularity). The *optimal* policy reaches the highest satisfaction ratio (82% on average for all 7 slices) but with much higher time complexity (e.g., 1 day for the considered scenario). From the figure, one can observe that the *Granular & Greedy* one (81%) outperforms the others

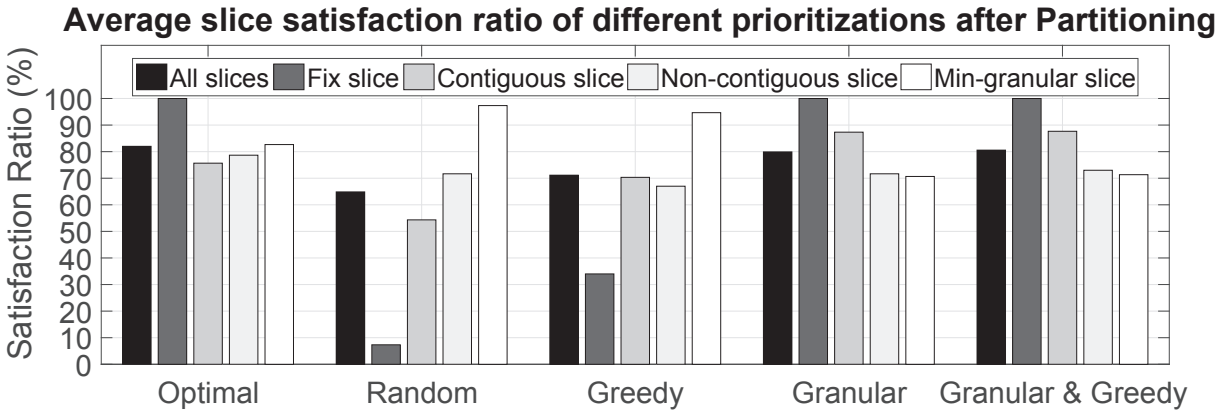
Algorithm 5.5: Min granularity Resource Partition (Min)

Input : s_k is the target slice
$IMap$ is the input unified resource grid map
Output: Sat is the slice satisfaction index
c_k is the selected SCS for target slice
$OMap$ is the output unified resource grid map

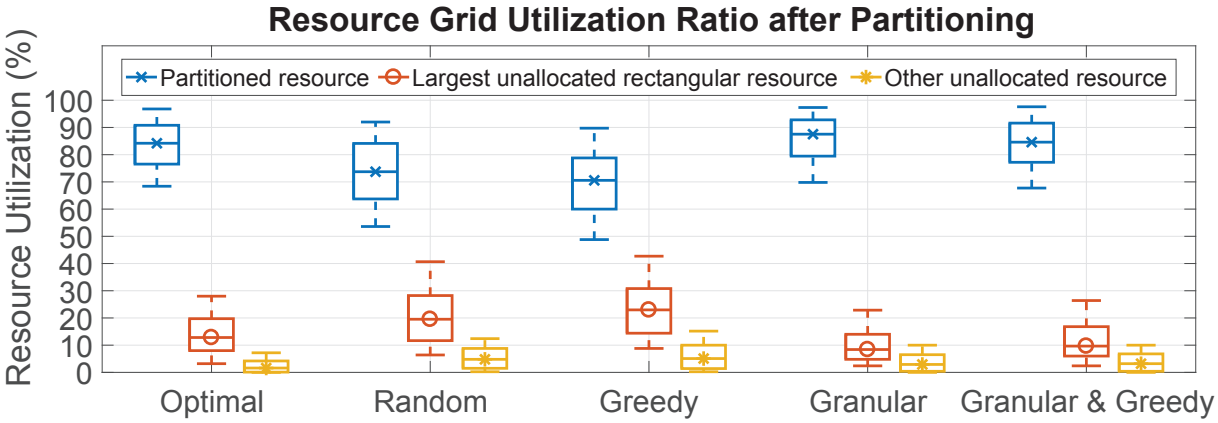
begin
  $MaxRec = 0$; /* Initialize the maximum unallocated rectangular */
  $Sat = 0$; /* Initialize the satisfaction index */
  foreach $scs \in SCS_k$ **do**
    /* Find the unallocated resources position (PN/PT) in $IMap^{scs}$ */
    $[PN, PT] = \text{FindUnRe}\,(IMap_{scs})$;
    $Size = |PN|$; /* The size of all available positions */
    $[In] = \text{InMaxRe}\,(PN, PT)$; /* Indicate resources are in the maximum rectangular or not */
    /* Sort resources base on whether they are in the largest rectangular*/
    $Order = \text{sort}\,(1 : Size, In, \text{'ascend'})$;
    if $|PN| \geq N_k^{scs} \times T_k^{scs}$ **then**
      $Sat = 1$; /* Current slice is satisfied*/
      $tMap = IMap^{scs}$;
      for $pos = 1$ **to** $N_k^{scs} \times T_k^{scs}$ **do**
        $tMap_{PN[Order[pos]],PT[Order[pos]]} = k$;
      $tRec = \text{MaxUn}\,(tMap)$; /* Find the maximum unallocated rectangular */
      if $tRec > MaxRec$ **then**
        $c_k = scs$;
        $MaxRec = tRec$;
        $OMap^{scs} = tMap$;

and is very close to the optimal policy as it not only follows the elasticity of resource granularity (i.e., granularity) but also seeks for the largest unallocated region (i.e., greedy) at the meantime. This approach is designed to follow the two complementary goals in the objective function of Eq. (5.1a).

Moreover, the resource grid utilization ratio over the unified resource grid map Map is shown in Figure 5.13b with three components: (1) the partitioned resources for all slices, (2) the largest unallocated rectangular, and (3) other unallocated resource in box plot. Both *random* and *greedy* policies have a larger unallocated rectangular ratio (20% and 23% on average) at the cost of a significantly lower slice satisfaction ratio (73% and 71% on average) shown in Figure 5.13a, i.e., more unallocated resources are due to the lower slice satisfaction ratio. Conversely, the percentage of the largest unallocated rectangular is close between the case that uses the optimal policy (12%) and the case that applies the *Granular & Greedy* policy (10%), which confirms the performance of the proposed algorithm. Finally, the *Granular & Greedy* prioritization policy takes much less execution time, i.e., polynomial time, to provide such close performance, which justifies its efficiency and applicability.



(a) Average slice satisfaction ratio

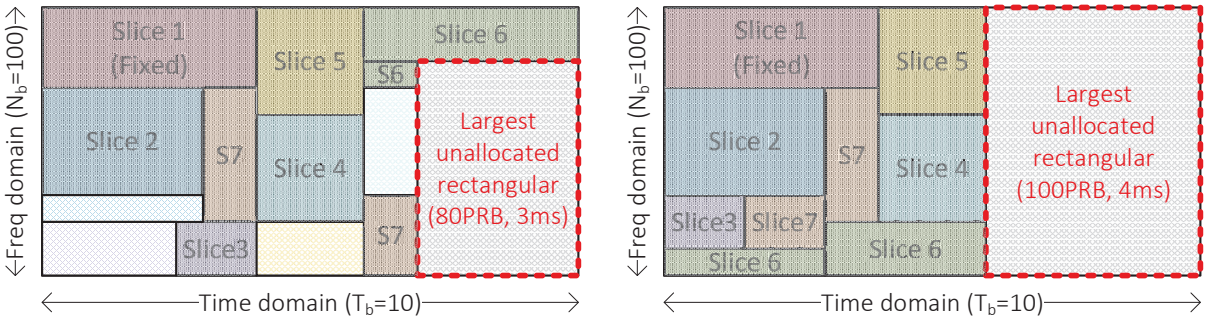


(b) Resource grid occupation ratio

Figure 5.13: Performance of different slice prioritization policies in resource partitioning.

5.4.2 Radio resource accommodation

Based on the stages depicted in Figure 5.8b, the intra-slice resource scheduling will be done in a per-slice manner based on revealed virtual view by the RAN runtime. Afterwards, these intra-slice resource scheduling results will be accommodated by the virtualization manager (cf. step f in Figure 5.8b). An example is provided in Figure 5.14 based on the previous inter-slice partitioning results of Figure 5.12b. First of all, the intra-slice scheduling results of all seven slices are shown in Figure 5.14a, where the gray portions are the scheduled parts and the transparent portions are the unused resources, i.e., unallocated by the intra-slice resource scheduler. We can notice that there are several unused resources distributed in the resource grid map. via proper inter-slice resource accommodation, a larger unallocated rectangular is formed in Figure 5.14b. Such advantage relies on the *resource abstraction* scheme as mentioned in Section 5.3.3.3. In this sense, the inter-slice accommodation is not necessary mapped to the same physical partitioned resource except for the slices that request the fixed position granularity. Finally, such unallocated rectangular region can be further utilized to increase the multiplexing gain via satisfying more slices (cf. step g in Figure 5.8b).



(a) Intra-slice scheduling results **(b)** Compact inter-slice accommodation

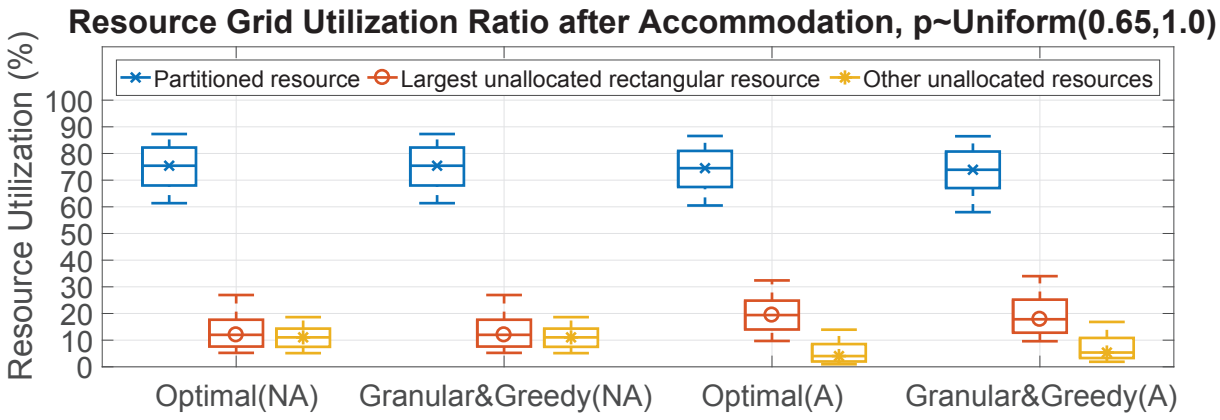
Figure 5.14: Examples of inter-slice resource accommodation.

Like the inter-slice resource partitioning, our objective of resource accommodation contains the two complementary goals as expected in Eq. (5.1a). In this sense, we can apply almost the same algorithm as proposed in Algorithm 5.1 with following modifications:

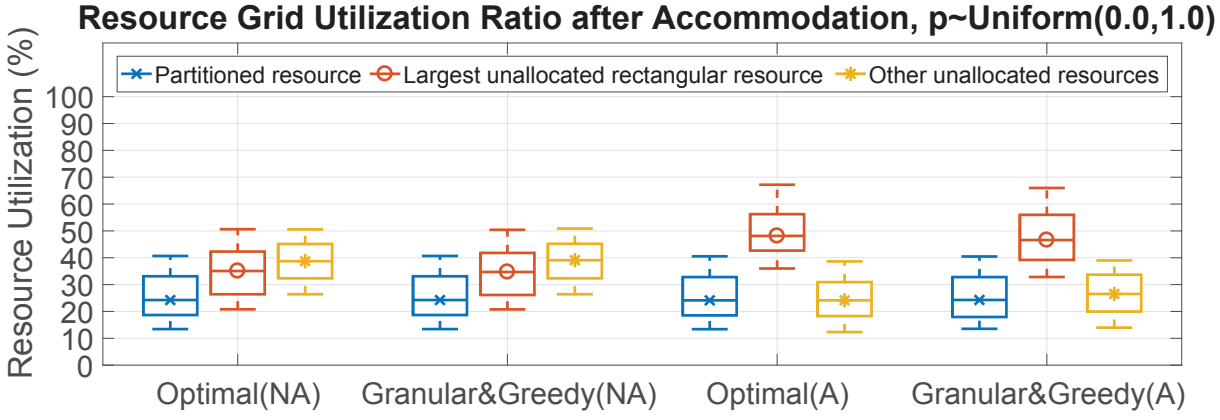
1. Apply the selected SCS in the inter-slice partitioning, i.e., $\mathcal{C}$, from Algorithm 5.1,
2. Replace the number of requested resource (i.e., T_k and N_k) with the number of allocated resource and denote the previous results from the inter-slice partitioning as $\overline{T}_k^{c_k}$ and $\overline{N}_k^{c_k}$,
3. Prohibit other slices to be accommodated into the resources partitioned for the fixed-position granularity slices and write the original constraints for the fixed-position slices of Eq. (5.1c) into Eq. (5.2).

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} b_{i,j,k} \leq \sum_{p=0}^{\overline{T}_k^{c_k}-T_k^{c_k}} \sum_{q=0}^{\overline{N}_k^{c_k}-N_k^{c_k}} b_{FT_k^{c_k}+p,FN_k^{c_k}+q,k}, \forall g_k = \text{Fix} \tag{5.2}$$

In the following, we evaluate the performance of two slice prioritization policies, i.e., *Optimal*, *Granular & Greedy*, in Figure 5.15 considering two cases: (a) no resource abstraction (denoted as NA in Figure 5.15), and (b) resource abstraction is applied except for fixed-position slices (denoted as A in Figure 5.15). Moreover, the aggregated traffic arrival rate of each slice is assumed to be proportional to the number of requested radio resource (i.e., $N_k \times T_k$) that is further multiplied with a time-varying uniformly-distributed *traffic arrival ratio* p. In the no resource abstraction case, all intra-slice scheduling decisions are accommodated within the partitioned resource (e.g., Figure 5.14a), while the resource abstraction scheme allows more freedom when accommodating (e.g., Figure 5.14b). In Figure 5.15a, we can see that no abstraction case only shows $\sim 2\%$ increasing in terms of the largest unallocated rectangular when comparing with the inter-slice partitioning result (cf. Figure 5.13b). In contrast, with the resource abstraction scheme, the optimal and *Granular & Greedy* prioritization policies provides $\sim 9.8\%$ and $\sim 8.2\%$ enhancement, respectively. Such benefit is further boosted when the average traffic arrival ratio p is decreased as shown in Figure 5.15b, i.e., p is changed from Uniform $(0.65, 1.0)$ to Uniform $(0.0, 1.0)$. These results show the resource abstraction advantages in terms of avoiding fragmentation of the overall resource grid. Finally, more slices can be further satisfied and we denote the final set of satisfied slices after accommodation and multiplexing as $\mathcal{S}_a$.



(a) Traffic arrival ratio $p \sim$ Uniform $(0.65, 1.0)$



(b) Traffic arrival ratio $p \sim$ Uniform $(0.0, 1.0)$

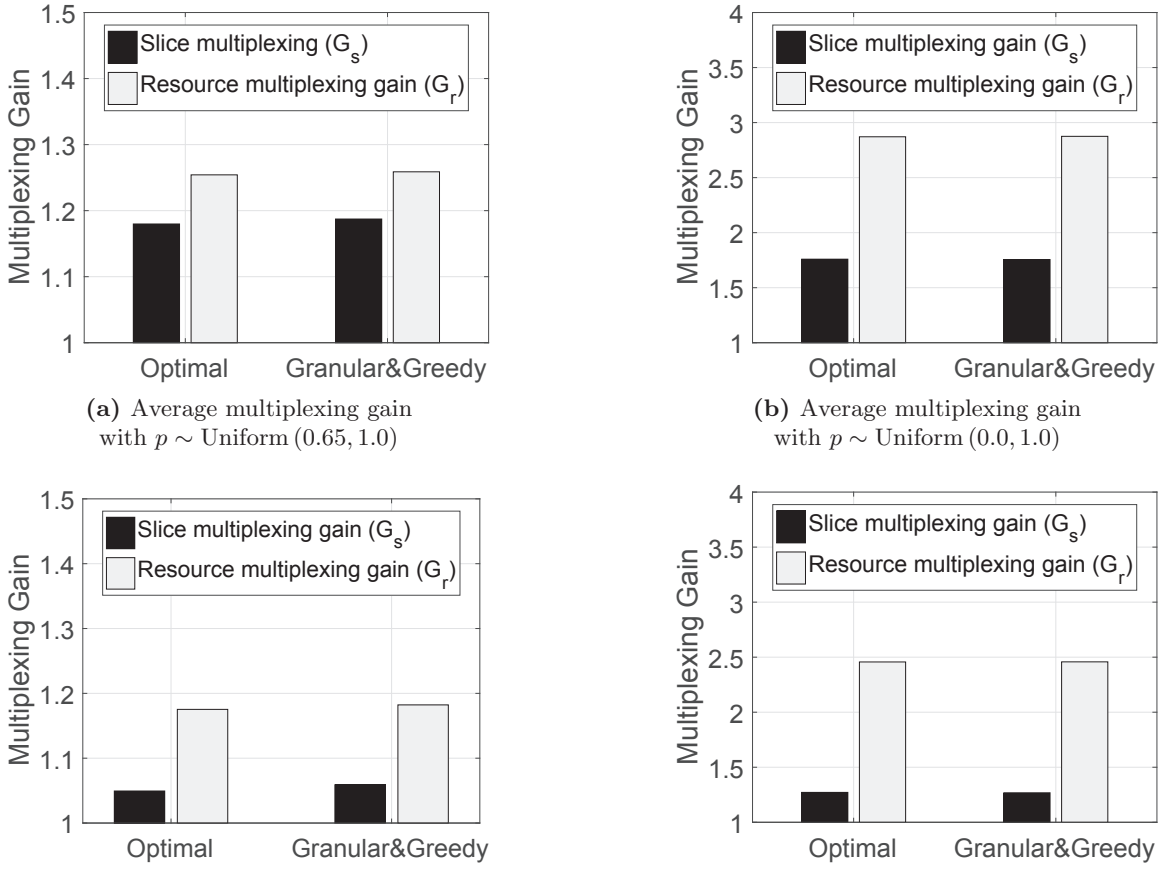
Figure 5.15: Performance of different slice prioritization policies and resource abstraction in resource accommodation.

5.4.3 Multiplexing gain

To explicitly represent the level of multiplexing benefits, we formulate the statistical multiplexing gain in two aspects: (1) slice instance, and (2) radio resource block. First of all, based on the previous results, we can see that the number of satisfied slices after the inter-slice partitioning (i.e., $|\mathcal{S}_p|$) is smaller than the number of satisfied slices after the inter-slice accommodation and multiplexing (i.e., $|\mathcal{S}_a|$) via utilizing the unallocated resource. Hence, the statistical multiplexing gain in terms of the number of supported slice can be written as

$$G_s = \frac{\text{Number of satisfied slices after accommodation and multiplexing}}{\text{Number of satisfied slices after partitioning}} = E\left[\frac{|\mathcal{S}_a|}{|\mathcal{S}_p|}\right], \quad (5.3)$$

where $\mathcal{S}_p$ and $\mathcal{S}_a$ are introduced beforehand to include the slices that are satisfied after partitioning and accommodation, respectively. Another aspect is to view the multiplexing gain in terms of the number of radio resource blocks via dividing the number of utilized resource blocks after accommodation and multiplexing by the number of scheduled resource blocks after the



(a) Average multiplexing gain with $p \sim \text{Uniform}\,(0.65, 1.0)$

(b) Average multiplexing gain with $p \sim \text{Uniform}\,(0.0, 1.0)$

(c) Worst case multiplexing gain with $p \sim \text{Uniform}\,(0.65, 1.0)$

(d) Worst case multiplexing gain with $p \sim \text{Uniform}\,(0.0, 1.0)$

Figure 5.16: Slice and radio resource multiplexing gain among different cases

intra-slice resource scheduling:

$$\begin{aligned} G_r &= \frac{\text{Number of utilized resource after accommodation and multiplexing}}{\text{Number of intra-slice scheduled resources}} \\ &= E\left[\frac{\sum_{s_k \in \mathcal{S}_a} T_k^a \cdot N_k^a}{\sum_{s_k \in \mathcal{S}_p} T_k^a \cdot N_k^a}\right], \end{aligned} \tag{5.4}$$

where T_k^a and N_k^a are the number of allocated resource blocks. Note that these two multiplexing gain formulations in Eq. (5.3) and Eq. (5.4) depend not only on the results of inter-slice resource partitioning and accommodation but also on the characteristics of extra slices to be satisfied, i.e., their traffic patterns and resource granularities.

In Figure 5.16, we show the multiplexing gain based on the inter-slice partitioning and accommodation results (i.e., optimal and *Granular & Greedy* from Figure 5.15), and utilize the aforementioned traffic pattern, i.e., *traffic arrival ratio* p, as introduced in the previous paragraph for each extra slice. Two resource granularity cases are considered for each extra slice: (1) contiguous granularity (i.e., $g_k = \{\text{Con}\}$), and (2) random granularity between contiguous,

non-contiguous and minimum (i.e., $g_k = \{\text{Con}, \text{NonCon}, \text{Min}\}$). The former shows that the multiplexing gain in its *worst case* as all extra slices require the exact contiguous resources, while the latter one shows the *average* multiplexing gain. The average multiplexing gain in Figure 5.16a is approximately 1.18 (G_s) and 1.26 (G_r) for both optimal and *Granular & Greedy* policies. When the average traffic arrival ratio is decreased (i.e., p), more unused resources can be multiplexed, and thus the multiplexing gain shown in Figure 5.16b are increased to 1.75 (G_s) and 2.87 (G_r).

Based on the above results, several observations are found. First of all, when comparing the average case (Figure 5.16b and Figure 5.16a) with the worst case (Figure 5.16d and Figure 5.16c), the slice multiplexing gain is reduced more significantly than the resource multiplexing gain. The reason behind is that these extra slices are rejected majorly due to their requested contiguous granularity rather than the their requested number of resources. Moreover, when comparing the results of different traffic arrival rates, the resource multiplexing gain is more significantly increased than the slice multiplexing gain since the bottleneck is in terms of the number of requested resource. To sum up, these two formed multiplexing gains have different characteristics: the slice multiplexing gain is more related to the resource granularity (e.g., Con, NonCon, Min), while the latter one concerns more on the traffic characteristic (e.g., N_k, T_k, p).

5.5 Proof-of-concepts

To validate the concept of RAN runtime slicing system and explore different use cases, we implemented an LTE-based prototype of RAN runtime following the aforementioned design descriptions. The RAN runtime is developed based on the FlexRAN agent[11] over the OAI platform [164], and each instantiated slice is built on top of the FlexRAN controller[12] with the customized CL. The main functionalities of the proposed RAN runtime services and CP/UP APIs are implemented and integrated within the agent. Slice selection for each user is done based on the the public land mobile network (PLMN) information, as a part of the unique international mobile subscriber identity (IMSI) in order to allow each user to associate to one slice. Note that as specified by 3GPP in [232], a single-network slice selection assistance information (S-NSSAI) can identify a slice and it comprises the (1) slice/service type (SST) and (2) slice differentiator (SD) to differentiate the slice service. Afterwards, each user can transport the NSSAI information that includes up to 8 S-NSSAIs to identify it preference(s) for slice selection.

The current implementation of a slice service descriptor is shown in Listing 5.1. It can describe a slice by its BS name (i.e., `name`), cell identifier (i.e., `cell_id`), and service types (i.e., `service_types`), where each service is defined by a set of service policies (i.e., `service_policy`) in both DL and UL directions that will be applied when a slice is created or updated. Note that such descriptor can support asymmetric slicing in uplink and downlink directions. Specifically, the service policies are expressed in terms of (1) the number of requested resources (i.e., vRBGs) and performance (i.e., rate and latency), (2) slice isolation requirement, and (3) slice priority as shown in listing 5.1. In terms of the requested resource abstraction types, currently the vRBG type 0/1 and vTBS type 0 are available and they can utilize the DL and UL resource allocation type 0 (see Table 5.4). Hence, each slice will be mapped with a vRBG pool in each TTI, and the overall resource partitioning is updated in every allocation window T. This slice isolation

[11] `https://gitlab.eurecom.fr/oai/openairinterface5g`
[12] `https://gitlab.eurecom.fr/flexran/flexran-rtc`

Listing 5.1: Slice service descriptor.

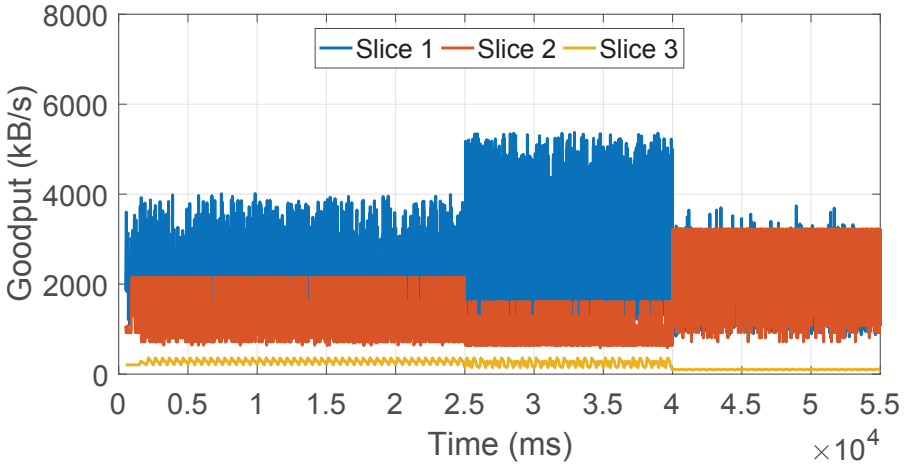
```
enb_slices :
  - name: BS1
    cell_id: val
    service_types: [ST1, ST2, ST3]
  - name: BS2
    ...
service_policy :
    ST1 :
        UL :
            requested_vrbg: val
            requested_rate: val
            requested_latency: val
            requested_priority: val
            requested_isolation: val
        DL :
            requested_vrbg: val
            requested_rate: val
            requested_latency: val
            requested_priority: val
            requested_isolation: val
    ST2:
        ...
    ST3:
        ...
```

property (cf. `requested_isolation`) can allow a slice to reserve its resources in the fixed position (i.e., without any multiplexing), whereas the slice priority (cf. `requested_priority`) is used to accommodate the resources and to preempt resources from other slices.

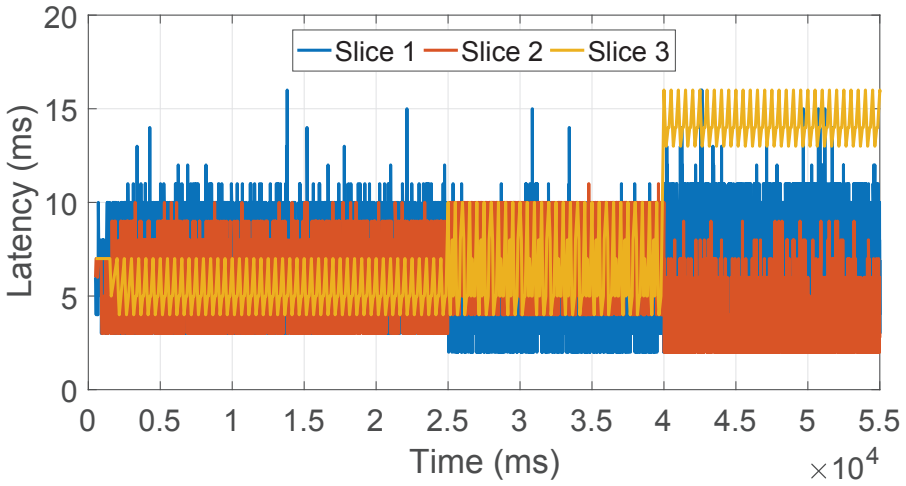
Using the aforementioned slice service descriptor, three slices are created on the top of the a single BS. They communicate with the RAN runtime using the asynchronous communication channels. Each slice embeds the control logics and operates on the virtualized resources and states based on the modified version of FlexRAN controller and its SDK. In following, we describe the experiment setup for each use case and present the respective results demonstrating the slice performance trade-off between isolation and sharing as well as the flexibility in terms of changing the RAN service definition dynamically.

5.5.1 Radio resource and control logic isolation

To demonstrate the impacts of inter-slice resource partitioning, we deploy three slices with different traffic patterns as follows: slice 1 with a variable bit rate emulating burst 720p video streaming, slice 2 with a compressed variable bit rate emulating a surveillance IP camera with 30 frame per second (FPS), and slice 3 with constant low bit rate emulating periodical sensing data. Each slice serves 5 different users (i.e., user 1 to user 5 belong to slice 1, user 6 to user 10 belong to slice 2, and user 11 to user 15 belong to slice 3), and each user transmits UL and DL data on the default radio bearer. Then, three different resource partitioning manners are applied at different time intervals: (a) fair partitioning that allocates 33% of total vRBGs to each slice before time instance $t_1 = 25s$, (b) greedy partitioning between t_1 and $t_2 = 40s$ that allocates 60% of vRBGs to slice 1, and 20% per slice 2 and slice 3, and (c) proportional partitioning after



(a) Aggregated good-put

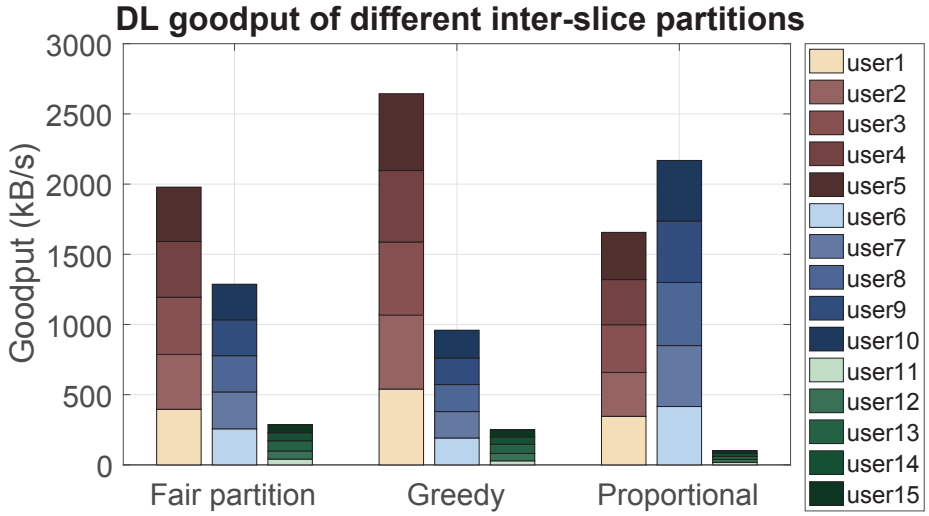


(b) Average latency

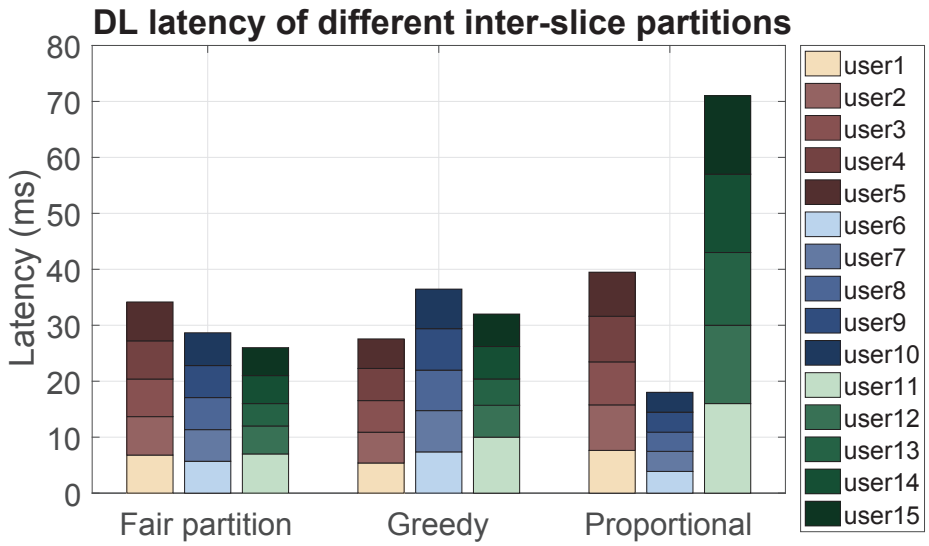
Figure 5.17: Slice performance of dynamic inter-slice partitioning

t_2 that allocates 50% of vRBGs to slice1, 40% to slice 2, and 10% for slice 3. Note that our focus in this experiment is on the number of requested resources (i.e., `requested_vrbg`), and thus the impacts of priority and isolation are not taken into account. As for the intra-slice scheduling, we apply a simple fair scheduling among users.

From the results presented in Figure 5.17, it can be observed that the slice aggregated good-put and average latency can significantly fluctuate when applying different inter-slice resource partitioning policies. However, any change of the inter-slice partitioning has no impact on the intra-slice scheduling policy as shown in Figure 5.18, in which all users are scheduled and the fairness can be preserved. The above results confirm the capability of RAN runtime in providing isolation among slices and performance guarantee, matching the challenge listed in Section 5.3.1. Further, it implies that the inter-slice partitioning and intra-slice scheduling can be decoupled and developed individually to meet the respective requirements of infrastructure provider and slice owner, i.e., a two-level scheduling approach as shown in Figure 5.5.



(a) Per-user average Good-put



(b) Per-user average latency

Figure 5.18: User performance of dynamic inter-slice partitioning

5.5.2 Radio resource preemption and multiplexing

In this experiment, we demonstrate the impacts of resource multiplexing and preemption, i.e., `requested_priority` and `requested_isolation` in Listing 5.1, on the perceived performance in a scenario with three slice instances, each hosting one user. Besides the applied resource abstraction/virtualization scheme, different slice service policies are explored: (a) slice 1 can preempt resources of all other slices when the actual (aggregated) rate exceeds the requested rate, (b) slice 2 can only increase its multiplexing gain by utilizing the unallocated resources, and (c) slice 3 may sustain its requested data rate as it can neither preempt nor multiplex resources but is subject to the preemption from high priority slice (i.e., slice 1).

We firstly show the box plot of measured RTT distribution in Figure 5.19 with different packet sizes ranging from 64 to 8192 bytes and inter-packet departure time from 0.2 to 1 second. We can observe that the smallest RTT with the lowest variability is achieved for slice 1, as such slice has the ability to preempt resources from others, and hence it can utilize available radio resources to meet its instantaneous traffic dynamics. On the other hand, slice 2 is able to maintain the average RTT compared with slice 1 with opportunistic improvement when there are some unallocated resources to be multiplexed (cf. Figure 5.8b). However, it still suffers from the delay variability caused by the scheduling delay. In contrast, slice 3 experiences the largest average RTT (almost twice as slice 1) with the highest variability, and it represents a typical best effort service.

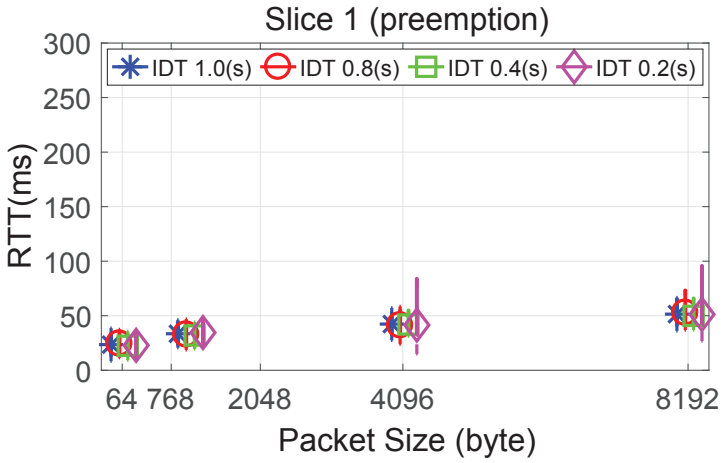
Besides, the relations between the measured RTT and the characteristics of traffic are observed in the following. We can see that there is a positive correlation between the RTT and the packet size for slice 1, as such slice does not experience any scheduling delay due to the resource preemption scheme, and thus its RTT is only proportional to the size of packet. As for the slice 2, there is no straightforward relation between inter-packet departure time and RTT since it can utilize some unused resource opportunistically. In contrast, an extra positive correlation is observed for slice 3 between the inter-packet departure time and the measured RTT. The reason being that the longer inter-departure packet time traffic of slice 3 suffers from a longer scheduling delay as it can neither preempt others' resource nor multiplex unused resources to reduce its RTT.

When examining the slice aggregated good-put and delay-jitter in Figure 5.20, it can be seen that slice 1 can flexibly adapt its data rate as a function of its instantaneous workload by preempting the resources from other slices, i.e., from 3 Mbps to 6 Mbps, while slice 2 experiences a data rate drop from 10 Mbps to 8 Mbps. The same trend is observed in the delay jitter, in which slice 1 experiences the minimum jitter as its highest priority and slice 3 suffers from the largest delay jitter due to its lowest priority (i.e., neither preemption nor multiplexing).

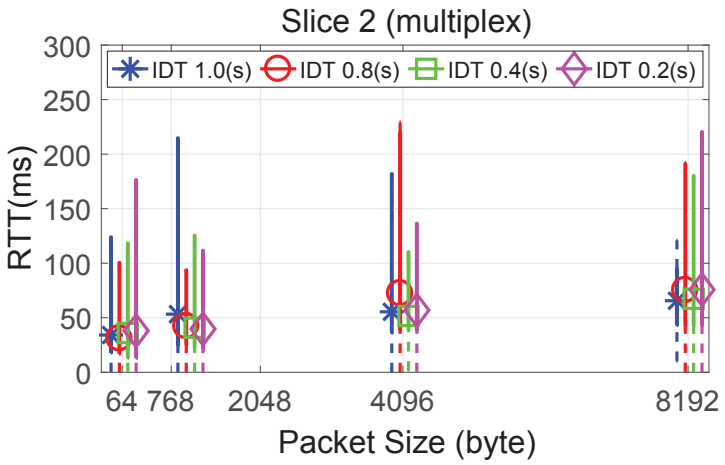
The above results reveal that the impacts of slice policy, generated from the service-specific SLA, in terms of the multiplexing and the priority when creating different slices. They enable resource reservation and preemption to potentially meet the slice-specific QoS requirements as well as the resource multiplexing to increase the efficiency of resource utilization by sharing the unused resources. Finally, such slice policy can be utilized together with the inter-slice partitioning mentioned in Section 5.5.1 to customizedly serve the needs of a slice, e.g., leveraging the greedy partitioning with the preemption/multiplexing to further strength its characteristic.

5.5.3 Network function and state flexibility

We then show the capability of the RAN runtime to change the underlying RAN module between monolithic and disaggregated RAN deployments from the infrastructure provider perspective. Particularly, we consider three possible RAN deployments at different time instances without instantiating any slice: (a) monolithic RAN deployment at t_1, (b) disaggregated RAN deployment using Split A at t_2, and (c) disaggregated RAN deployment using Split B at t_3. Our considered disaggregated RAN deployment uses UDP/IP based Ethernet transportation over the FH interface with one switch between RU and DU to route the traffic. To this end, the related RAN module parameters shall be reconfigured such as FH interface (e.g., IP address, transportation protocol, packetization) and functional split parameters.



(a) Slice 1



(b) Slice 2

(c) Slice 3

Figure 5.19: Impact of preemption and multiplexing on RTT

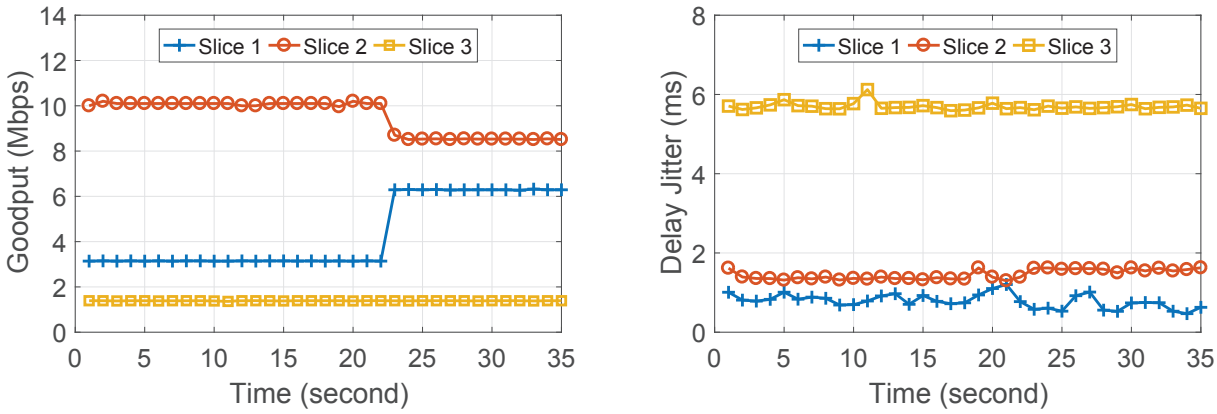


Figure 5.20: Impact of preemption and multiplexing on good-put and delay jitter

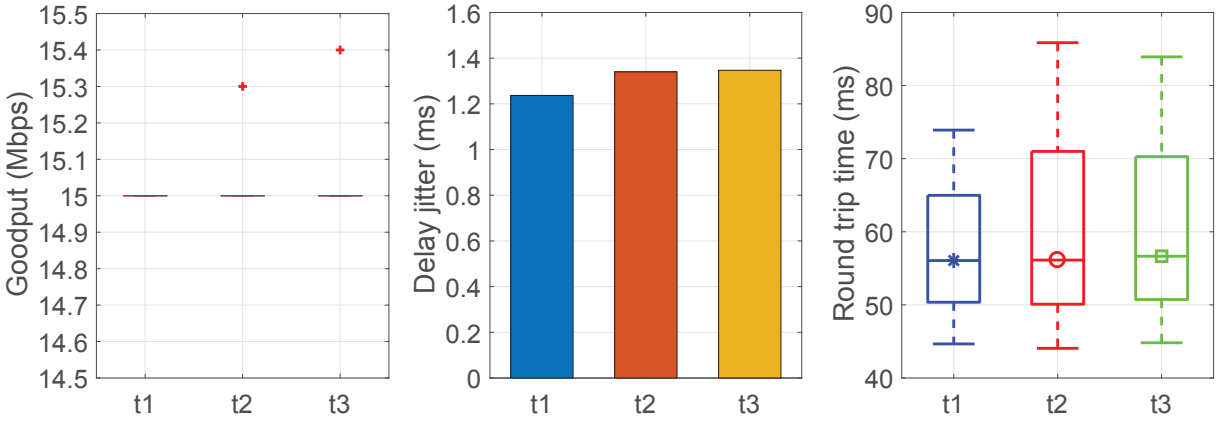


Figure 5.21: Flexible RAN deployment impacts on good-put, delay jitter and RTT

Moreover, we measure the UP performance to observe the impacts on the service owner. The UP measurement results are shown in Figure 5.21 in terms of the good-put, delay jitter and RTT when a 15 Mbps traffic flow is transferred in the DL direction. We can see that these is no good-put drop when the infrastructure provider modifies the RAN deployment. This is because the considered two functional splits only require the RAN module reconfiguration, i.e., DFT/IDFT network functions, without any state synchronization (cf. Figure 2.2). Nevertheless, both delay jitter and RTT are increased at t_2 and t_3 due to the Ethernet packet loss as well as the extra computation time spent for the packet processing along the FH link. Based on this performance degradation, the RAN deployment change is applicable only when no service-specific SLA violation is observed.

Although the changes of RAN deployment are mainly reserved for the infrastructure provider to ensure the network service performance; however, the update of split can be possibly made for a slice owner by appropriately customizing the CP/UP functions at each RAN module (cf. Figure 5.9 and Figure 5.10). In such a case, the RAN runtime shall make sure that the SLA is maintained when there is any change in the service descriptor, and transfer the CP/UP states between disaggregated RAN modules (i.e., RU, DU, CU), as listed in Table 5.5.

5.6 Discussions

In the aforementioned work, the RAN runtime slicing system is proposed to provide a number of RAN runtime service and to provide different levels of isolation and sharing for each slice. As for the next steps, three potential directions are provided in the following.

The first direction is to investigate the interfaces and applications of the RAN runtime slicing system with other subsystem. For instance, to enable the RAN-domain service management and orchestration, some extra interfaces are necessary between the RAN runtime and the dedicated/shared RAN-domain management functions defined by 3GPP and ETSI. Another possible approach is to expose the SDK on top of the RAN runtime to enable the development and deployment of the slice-specific control applications.

Secondly, to extend our investigated inter-slice partitioning and accommodation approaches, extra consideration on the QoS assurance is needed. Moreover, the resource over-provisioning approach can be take into account to find a balance between multiplexing gain and QoS satisfaction. Last but not least, more resource abstraction schemes can be considered corresponding in the future to the newly-introduced service performance indicators, e.g., latency and reliability.

The third direction aims to extend the provided forwarding engine for customized granularity. For each function of different network layers, a flexible composition among the customized one and the shared one is desired for efficiency and performance. Moreover, to better satisfy the per-flow QoS requirements, the forwarding engine can be extended in the per-flow manner at the cost of extra overhead.

5.7 Conclusions

In this chapter, we propose the RAN runtime slicing system that serves as a flexible execution environment to run multiple customized slice instances with the required levels of isolation, while sharing the underlying RAN modules and infrastructure. We elaborate on the design of such system and identify the its functionalities in both CP and UP. A new set of radio resource abstractions are defined to efficiently provide resource isolation among different slices. We also propose the inter-slice resource partitioning and accommodation approach that can satisfy the requests of different granularities and maintain a significant multiplexing gain with acceptable complexity. Finally, we implement the proposed RAN runtime slicing system over the OAI platform in three use cases that exactly match aforementioned RAN slicing challenges.

Chapter 6

Slice Orchestration for Multi-Service Disaggregated RAN

6.1 Introduction

Based on the aforementioned RAN runtime architecture in Chapter 5, different levels of isolation and sharing across processing, state and resource can be provided in a customized manner. Nevertheless, there is still one open question in terms of how to orchestrate network slice instances (NSIs) over the RAN domain through the envisioned RAN runtime slicing system. Such issue is critical when deploying an E2E service architecture spanning all relevant domains (i.e., RAN, CN) in an orchestrated manner for service performance and scalability. More precisely, the service orchestrator needs to translate the high-level definitions of services and applications into the lower-level building components and handles the life-cycle management tasks like service deployment and upgrade. In this sense, it requires an architecture framework that is not bounded to a particular use case or scenario and is adaptable with few manual interventions (by either infrastructure provider or slice owner). Such architecture framework also needs to support the customization for multiple tenants across multiple domains leveraging the notion of modularization and virtualization.

To orchestration network services, several standardization bodies define their slicing information models, e.g., 3GPP, ETSI NFV, and ONF in [206], [233], and [234], respectively, as mapped in the Figure 6.1. Take the 3GPP aspect as an example, a network slice contains one or more network slice subnets, each of which contains one or more network functions and can also contain other network slice subnets. These network functions can be managed as VNFs and PNFs of different subnets. More specifically, the network slice subnet instance (NSSI) in introduced for the NSI management. An NSSI may contains core network and/or access network functions and it can be dedicated to one NSI or shared by more than one NSIs/NSSIs. Thus, the ETSI NFV network service can be regarded as a resource-centric view of a 3GPP network slice. Also, the 3GPP network slice subnet can be represented by the nested ETSI NFV network services or directly attaching VNFs/PNFs to the ETSI NFV network service. On the other hand, the client context and server context of the SDN controller defined by the ONF can also be mapped to the ETSI NFV network service and NFV infrastructure. To leverage these aforementioned network slicing information models, one design issue is to enable the proper interactions between our proposed RAN runtime and the defined network management framework.

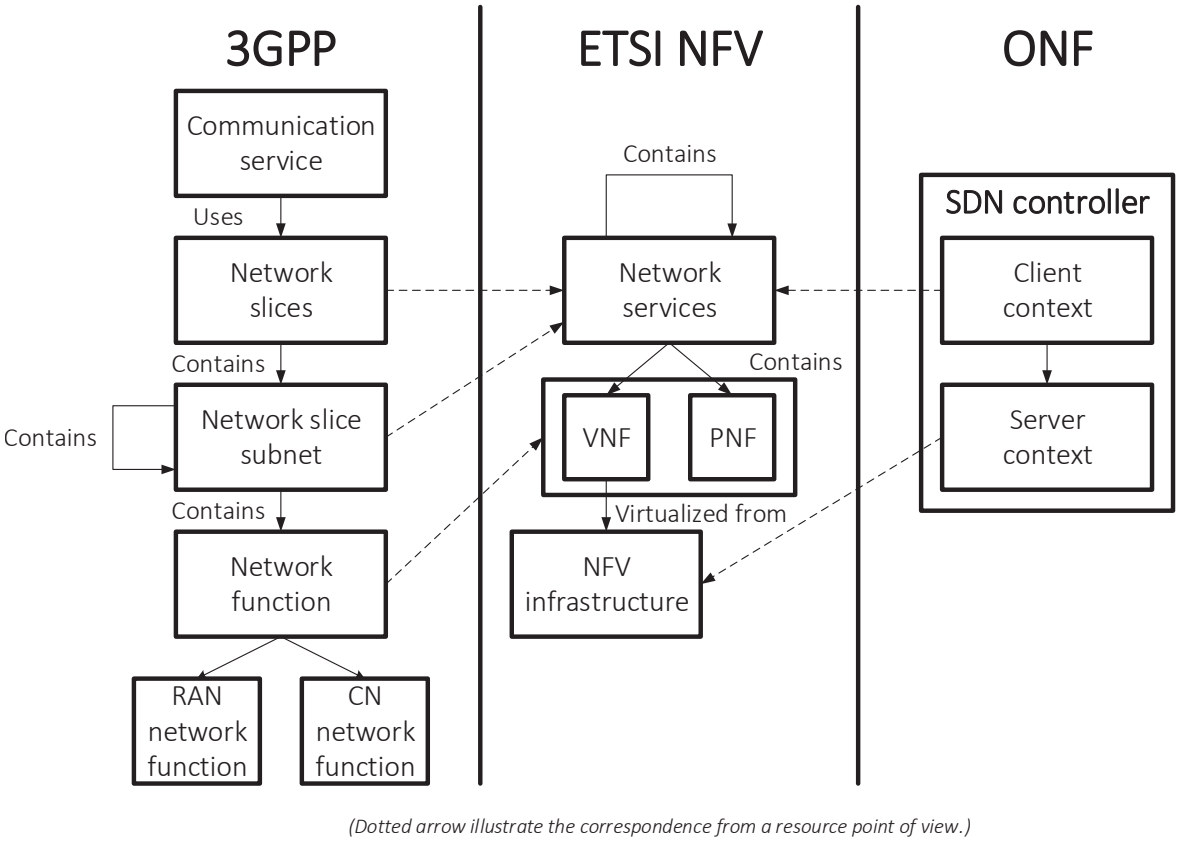


Figure 6.1: Mapping between the slicing information models of 3GPP, ETSI and ONF

To further elaborate on the aforementioned design issue, an example is depicted in Figure 6.2. We can see that there are three instantiated NSIs leveraging the underlying four NSSIs, i.e., NSSI 1 and NSSI 2 are of RAN with different RATs while NSSI 3 and NSSI 4 contain dedicated and shared CN components respectively. We can observe that the RAN runtime is the intermediated between the instantiated NSIs/NSSIs and the underlying RAN module, and thus several interfaces shall be designed to enable the aforementioned RAN runtime services in Chapter 5. Moreover, when orchestrating multiple services at the RAN, efficient resource usage and service scaling are mandatory, while considering specific slice requirements in terms of performance and customization. Note that even current solutions such as ETSI open source management and orchestration (OSM)[1], open platform for NFV (OPNFV)[2], and M-CORD[3], allow slice customization; however, they only consider isolation by running the RAN service in a virtualized environment and fail to meet the trade-off between customization and sharing. Furthermore, none of the above studies investigate the network slices orchestration in the case of disaggregated RAN. To be complementary with these works, we identify the following three challenges in this chapter to design the slice-enabled disaggregated RAN:

[1] `https://osm.etsi.org/`
[2] `https://www.opnfv.org/`
[3] `https://www.opennetworking.org/solutions/m-cord`

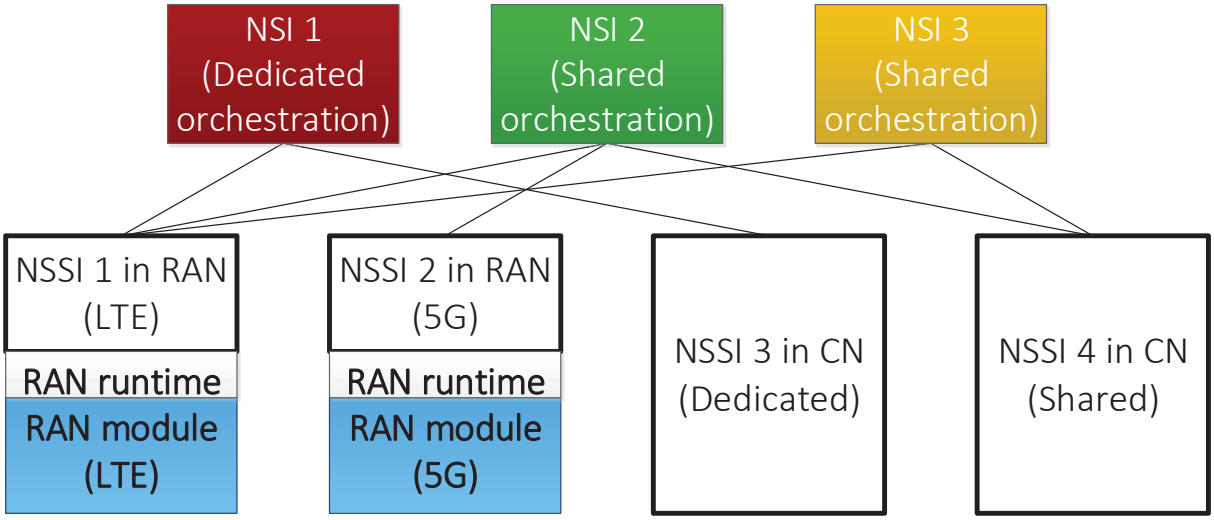


Figure 6.2: Relation between NSI, NSSI and RAN runtime

1. Efficient slice modeling to validate service requirements when deploying on a per-slice basis,
2. Multi-service chaining of customized and/or shared PNFs/VNFs, and
3. Multi-service chain placement in densely deployed infrastructures with auto-scaling action.

To tackle these challenges in this chapter, we first exploit the RAN runtime by a set of interfaces for the interaction with the management and orchestration system. Such proposal retains the compatibility with the E2E slice management and orchestration functions introduced by 3GPP in [206]. Moreover, we propose a multi-service chaining and placement algorithm for the disaggregated RAN. Finally, the proposed algorithm is evaluated in a specific UDN scenario showing the impacts of the auto-scaling actions to increase the service acceptance ratio.

6.2 Exploiting RAN runtime for slice management and orchestration

In this section, we first outline the slice-based orchestration architecture via exploiting the proposed RAN runtime slicing system with the network slice management system defined by 3GPP. Further, we elaborate on the functionalities of the required interfaces for the communication between the RAN runtime and the orchestration systems. Finally, the dynamic slice modeling approach is introduced exploiting the identified interfaces.

6.2.1 Overview of network slice management and orchestration

As mentioned beforehand, the management of network slices requires analyzing the service requirements to ensure the desired performance on a continuous basis. Such a process consists of the operations related to admission control, SLA monitoring, performance management, fault management and service lift-cycle management on the NSIs. In this sense, 3GPP defines the following entities in [206] regarding orchestration and management of NSIs:

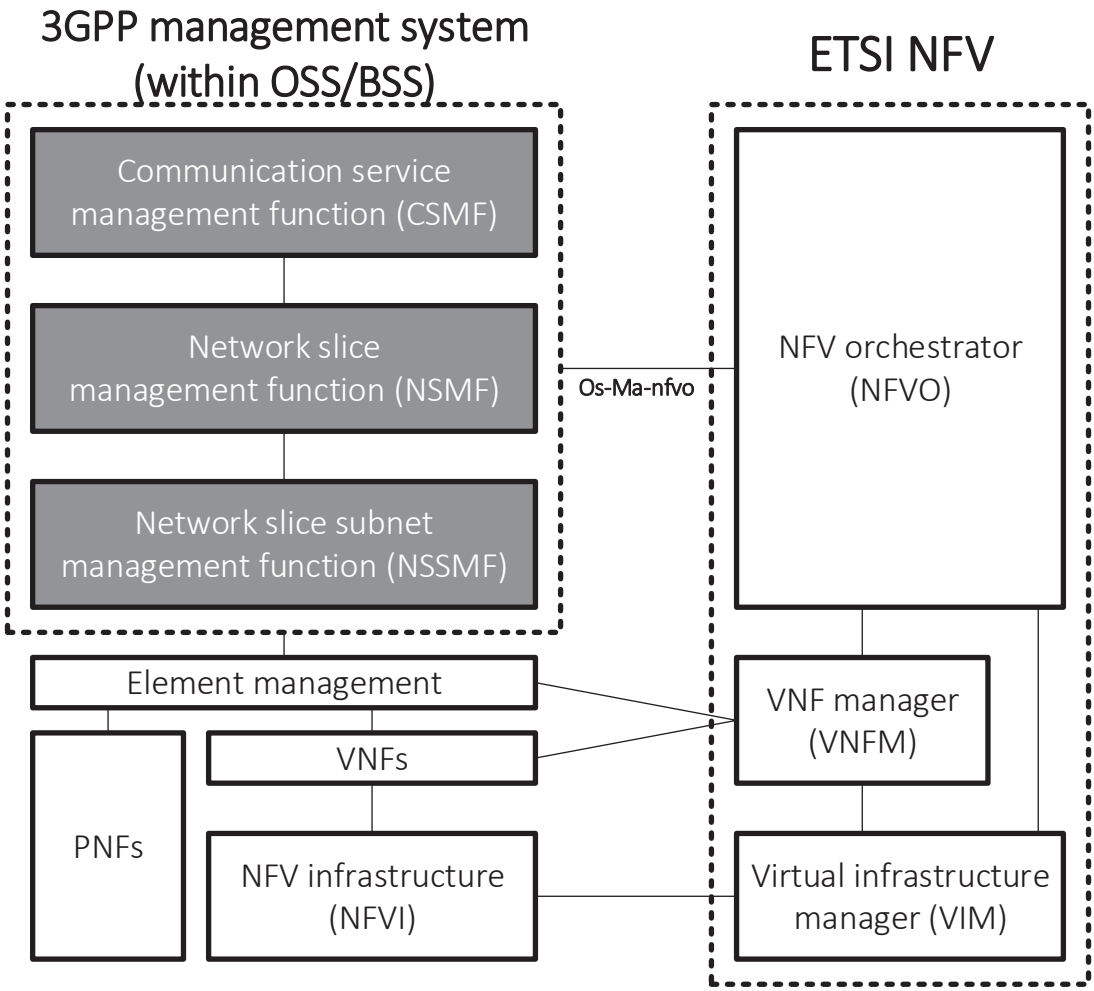


Figure 6.3: 3GPP network slice management in an ETSI NFV architecture

- *Communication service management function (CSMF)*: Responsible for translating the communication service requirements to network slice requirements
- *Network slice management function (NSMF)*: Responsible for the E2E management and orchestration of the NSI
- *Network slice subnet management function (NSSMF)*: Responsible for the management and orchestration of the sub-network slice instance in a specific domain (e.g., RAN-NSSMF and CN-NSSMF is responsible for the management of RAN and CN, respectively)

These entities can be interacted to the ETSI NFV architecture as shown in Figure 6.3, in which the Os-Ma-nfvo interface is used for the interaction between the 3GPP management system and ETSI NFV orchestrator. More specifically, to support the requirements for an NSI and NSSI, the NSMF and NSSMF need to determine the mapped network services, VNFs and PNFs in either dedicated or sharing manner.

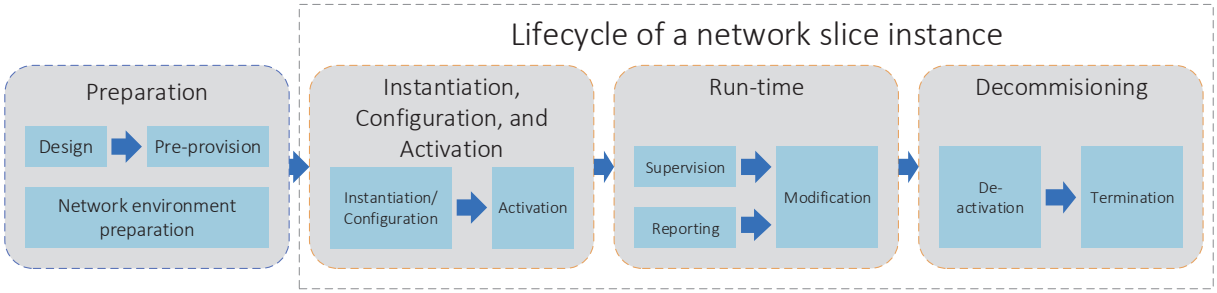


Figure 6.4: Life-cycle phases of an network slice instance

Moreover, 3GPP defined the life-cycle of an NSI in Figure 6.4, in which there are four phases: a) preparation; b) instantiation, configuration, and activation; c) run-time, and d) decommissioning, that are explained in more details in [206]. From the ETSI NFV perspective, the resource requirement for a network slice template (NST) (used for network management) in the preparation phase can be realized by existing or new NSDs (used for the VNF deployment). Specifically, the NST can describe the business applications, including the particular use case (e.g., public safety), deployed topology (e.g., geographical region), corresponding SLA, policies (e.g., resource isolation) and requirements (e.g., E2E latency). Such NST can be further translated into the NSD defined by the ETSI, including the related PNFs and VNFs with their dependencies, monitoring parameters, KPIs, and deployment attribute (e.g., life-cycle events). Then, in the instantiation, configuration and activation phase, the ETSI NFV functions will instantiate the underlying network services (cf. Figure 6.1), configure the virtualization-related parameters, and trigger the VNF activations. In the run-time phase, the ETSI NFV will operate for the performance management, fault management, and life-cycle management of virtualized resources. Finally, the decommissioning phase triggers ETSI NFV to terminate the underlying network service instances.

6.2.2 Architecture of the RAN runtime slicing system and 3GPP management system

Based on the aforementioned 3GPP management functions, our goal is to incorporate our designed RAN runtime slicing system with these management functions to facilitate the network slice orchestration and management over the RAN domain. Figure 6.5 illustrates the proposed architecture, which includes the NSSMF entity to carry out the RAN life-cycle management and the RAN runtime to provide a multi-service execution environment considering both dedicated and shared orchestration and management functions for the underlying RAN module. NSMF is responsible for preparing an integrated environment for the NSI, while also controlling the NSI life-cycle, interacting with the domain-specific orchestrator, i.e., RAN-NSSMF. Note that the RAN runtime allows a running NSSI to interact with the RAN modules via the NSSMF, which monitors the allocated resources and controls the behavior of the underlying network.

The isolation, abstraction, and customization properties provided by the RAN runtime enable the RAN-NSSMF to orchestrate the NSSI and the behavior of the underlying RAN modules considering the service requirements. The RAN runtime also allows the creation of NSSIs that exploit reusable RAN service chains, reducing instantiation costs and CP/UP processing.

Specifically, the following three RAN runtime services are leveraged to facilitate the network slices orchestration procedures:

- The *context manager* will do the CRUD operation on the input NSD. Moreover, it will register the slice-specific life-cycle primitives to the slice manager and the requested resources and/or performance to the virtualization manager.
- The *slice manager* determines the CP/UP processing chain for each NSSI and the relevant traffic flows based on the service requirements and its context. It also programs the forwarding engine through a set of rules allowing direction of input and output streams across shared processing operated by the underlying RAN module, while customizing the processing function required by each NSSI. When NSSI is modified, the RAN processing operated by the slice manager may be updated to support service continuity. This slice manager also resolves conflicts among different NSSIs based on predetermined policy rules.
- The *virtualization manager* provides the required level of isolation and sharing to each NSSI. Specifically, it abstracts physical stats and resources to virtualized ones and partitions them based on the NSSI requirements and modules context. Finally, it accommodates them back to the physical states and resources.

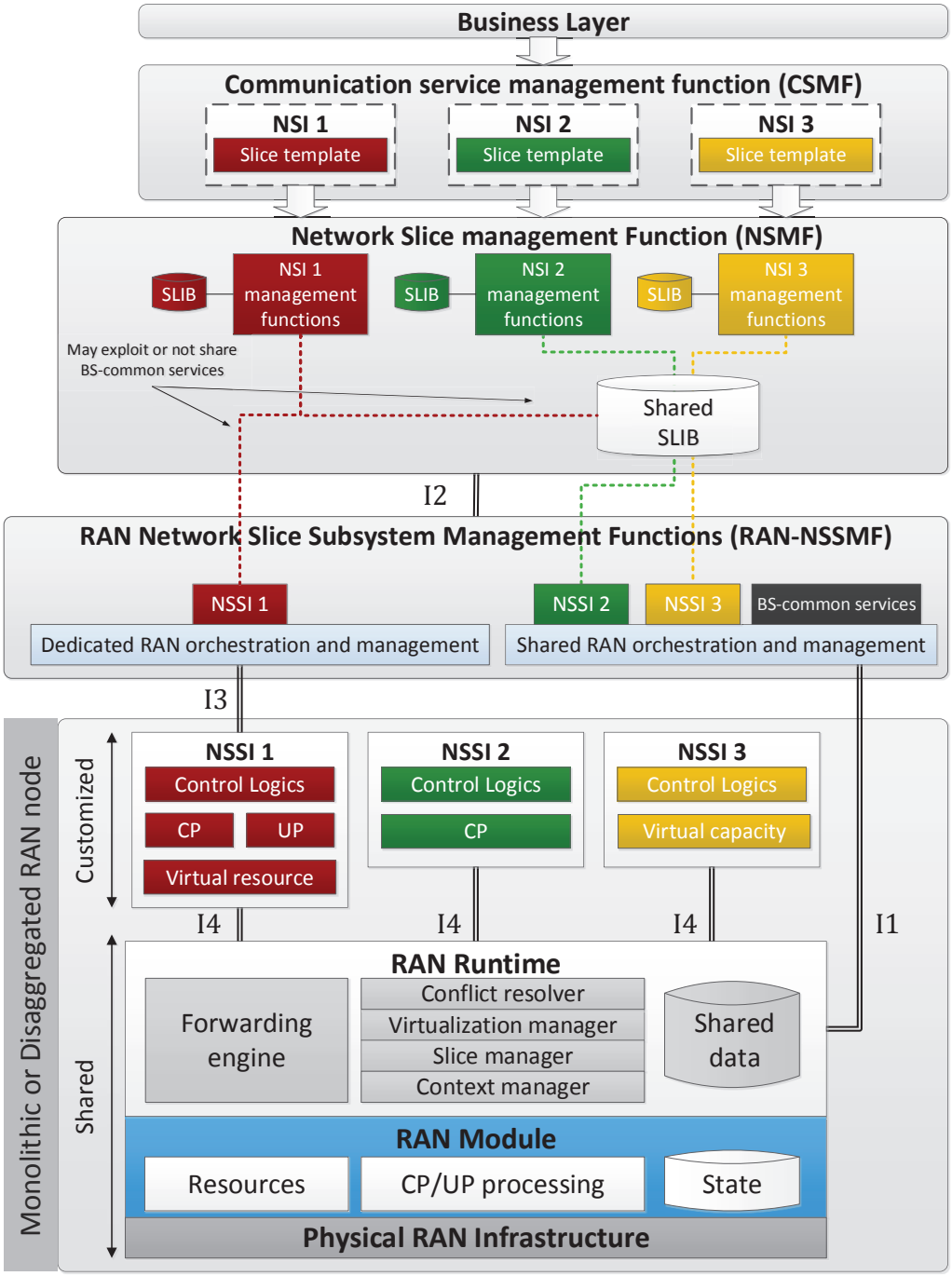
Both the *slice manager* and the *virtualization manager* exploit the data that is shared between different NSSIs. This data is related to the different NSSIs' context (e.g., basic information like its identity, registered RAN runtime services, user context, and NSSI association) and module context (e.g., CP and UP state information, module primitives) that is used to customize and manage a slice. Such data may be exploited by different NSSIs, especially once modified to reflect user and network dynamics.

Via using the provided architecture, each slice can be orchestrated through the 3GPP management functions and RAN runtime. The CSMF creates a slice template for each slice that communicates to NSMF, which contains tangible slice requirements. Using this template, the NSMF performs the NSI preparation. The NSMF maintains a shared slice information base (SLIB) that contains slice-specific information including SLA, user context, user-to-slice association and service inventory. Once instantiated, each slice is orchestrated in the RAN domain, which can be in either shared or dedicated mode from the RAN-NSSMF entity. For NSSI 1 in Figure 6.5, the RAN-NSSMF orchestrates a set of dedicated RAN network functions that are customized according to its SLA requirements. The RAN runtime is the intermediate entity responsible for fine tuning the necessary RAN modules. As NSSI 2 and NSSI 3 do not request such customization, their network functions are managed through a shared service from the RAN runtime, also exploiting the information stored in a shared SLIB at the NSMF level.

Finally, the RAN runtime can support a number of NSSIs depending on the amount of requested resources and the SLA of each slice. If all slices opt for a virtualized resource without any hard guarantee, a large number of slices can be accommodated subject to the overhead at RAN runtime to manage slices. Otherwise, the bottleneck will be the number of requested radio resources and traffic dynamics when a certain level of guarantee is required by each slice.

6.2.3 Interfaces functionality analysis

A set of interfaces (`I1` to `I4`) are introduced enabling communication between the various entities as depicted in Figure 6.5.



Interface	Description
I1	Expose active RAN runtime services and retrieve messages for monitoring and feedback.
I2	Subscribe to the slice-specific events and populate SLIB accordingly.
I3	Customize management and monitoring for each slice and indicate any changes in underlay RAN.
I4	Register a slice to the RAN runtime and consume RAN runtime service as a separate process.

Figure 6.5: Architecture of the RAN runtime slicing system and 3GPP management system

I1 serves several purposes. Before instantiating any NSSI or service, the RAN runtime and RAN modules are operational and expose BS services like broadcasting system information or neighboring nodes information. These are irrelevant to the number of instantiated NSSIs. Information about the active services and capabilities of the RAN runtime are exposed through the I1 interface. A slice registration to the slice manager by the context manager during the creation of an NSSI is required. Based on the service description, the slice manager performs several operations as detailed in Section 6.2.2. However, certain operations are only performed for slices orchestrated and managed in a shared manner. Monitoring and feedback messages are retrieved from the RAN runtime through the I1 to facilitate coordination among different slices regarding the available resources and service requirements.

I2 is the interface between the NSMF and the RAN-NSSMF and is currently standardized by 3GPP.

I3 is the interface between the dedicated orchestrator and the corresponding customized RAN network functions at the RAN node, through which a slice owner can customize slice management and monitoring, apply slice-specific life-cycle primitives, and maintain own service continuity and isolation. For instance, this interface can be used for the communications required to operate customized CP/UP processing and then program the forwarding engine at the RAN runtime accordingly (through the I4 interface). Moreover, the slice manager will indicate through I3 about any changes of the underlying RAN module toward the dedicated orchestration for agile changes on its service definition.

I4 provides a communication channel with the RAN runtime allowing an NSSI to register with the RAN runtime and consume its services, whether it is local or remote. When a slice is deployed locally, I4 exploits the inter-process communication mechanism to allow a slice to perform real-time operations (e.g., MAC network function) with hard guarantees. However, when considering non-time-critical operations (e.g., PDCP network function), communication is made through an asynchronous communication interface (e.g., message bus).

6.2.4 Slice modeling

Based on the aforementioned interfaces between the RAN runtime and the 3GPP management functions, we hereby use an example to elaborate on how the slice template is modeled and updated. While slice templates capture use-case specific requirements with the associated SLA and infrastructure provider policies [159], dynamic slice updates may be needed to maintain and optimize the performance. As depicted in Figure 6.6, a slice template contains different types of resources (e.g., clouds, nodes, links, network functions and applications, radio spectrum, and RAN modules) considering the infrastructure provider's libraries and catalogs, and requested SLA. During the run-time phase, the slice template may be periodically optimized (e.g., compare service KPIs against the SLA) based on the monitoring information provided by the RAN runtime through I1 and I3 interfaces. For instance, the life-cycle of each slice is triggered based on the life-cycle of underlying RAN module, e.g., the stop primitive of RAN module triggers the stop of all slices. The updated slice template is then actuated by NSMF, and applied to the RAN module through RAN-NSSMF and RAN runtime. Such optimizations process may require negotiations among different providers to fulfill the service requirements, and can be applied for resource partitioning/priority (by the RAN runtime), service placement and scaling (by the orchestrator), functional split (by the infrastructure provider), and processing customization (by the slice owner) relying on the predefined rules and/or the cognitive logic.

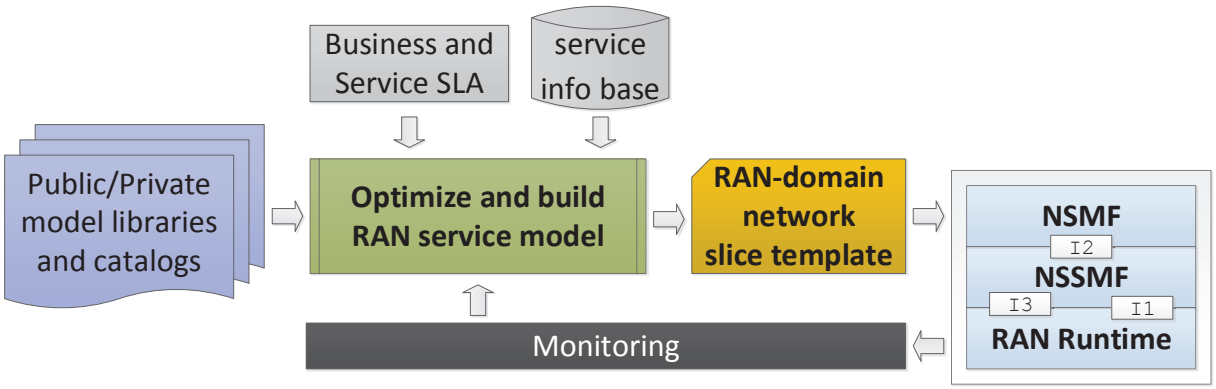


Figure 6.6: Process of RAN modeling and slice template optimization

6.3 Slice orchestration for disaggregated RAN

While the RAN runtime enables a multi-service execution environment, slice orchestration in a disaggregated RAN still remains an open question. In this section, two main challenges are investigated: (1) *multi-service chaining* to customize per-slice service chain, and (2) *multi-service placement* for shared/dedicated service chains.

6.3.1 Multi-service chaining

A RAN service chain can apply the functional split (cf. Figure 2.2) in a disaggregated RAN to serve the network slice in a customized manner. Practically, it can be composed *horizontally* based on the shared network functions provided by the underlying RAN module (Figure 6.7) and/or *vertically* through the customized ones chained by the slice owner to fulfill its service requirements. Note that the functional split between disaggregated RAN entities is generally determined by the infrastructure provider based on the aggregated BS KPIs and fronthaul/midhaul performance (e.g., capacity and latency). However, slice owners can change the functional split through the CP/UP customization with the associated input/output forwarding chain through RAN runtime. For instance, a service may request a hardware-based accelerator for channel decoding and a dedicated radio resource to enable low-latency communications. These network functions are managed by either the RAN runtime or the VNF manager, corresponding to PNF and VNF, respectively. The E2E RAN service chain is maintained by the forwarding engine at the RAN runtime, leveraging the SDN-based match-action abstractions to build slice-specific forwarding input and output paths across shared and dedicated network functions.

An example with different levels of slice customization is shown in Figure 6.7. The three slices are orchestrated according to their respective slice templates, which contains the requested level of dedication and sharing in terms of the RAN-domain management and orchestration function and the RAN processing. Slice 1 requires the dedicated slice management and orchestration function using the `I3` interface with the customized processing over SDAP, PDCP, RLC, MAC and High-PHY layers, while it only share the Low-PHY layer processing. In contrast, both Slice 2 and Slice 3 utilize the shared service orchestration and management function with `I1` interface toward the RAN runtime. Finally, Slice 2 only customizes its SDAP processing and Slice 3 is built on top of the shared processing.

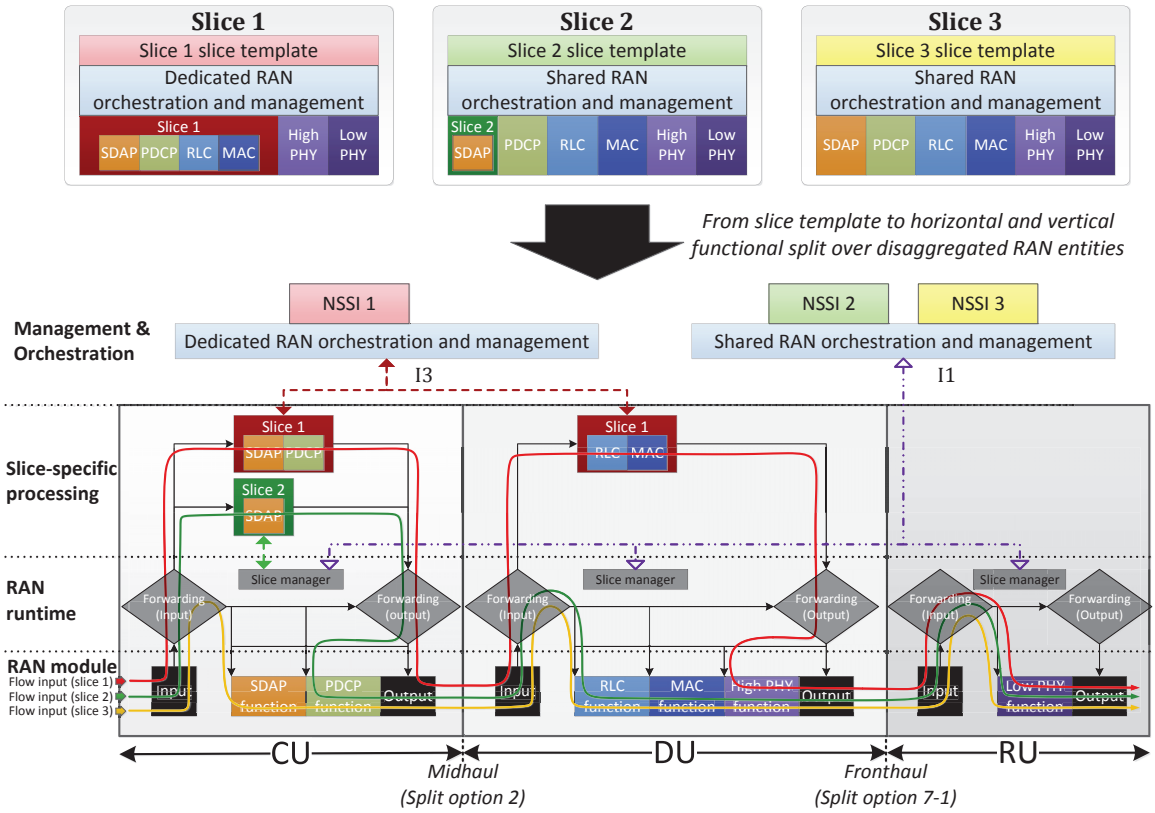


Figure 6.7: Example of multi-service chaining and forwarding in a disaggregated RAN

Note that the slice-specific processing customization is described in the slice template, and the customized forwarding path of each slice is maintained by the slice manager of the RAN runtime under the control of the corresponding RAN-NSSMF. However, the input/output endpoints of each disaggregated RAN entity perform infrastructure-dependent packet processing like encapsulation and switching/routing for fronthaul/midhaul transportation. As the RAN runtime maintains the state for both dedicated and shared network functions, it facilitates dynamic slice template update by handling state synchronization among the involved RAN entities. This is the case when the RAN functional split and/or placement are updated, for example according to the spatio-temporal traffic dynamics.

Moreover, different faults may happen at several levels when chaining a RAN service, such as broken infrastructure, lack of resources, missing SLAs, or RAN runtime overload. The fault can also be service-dependent: if it is customized, the simplest recovery approach would be reusing the shared processing with the default configurations. The consequence would be service unavailability and SLA violation for the slice users, but they may still remain connected to the network for basic BS services. Last but not least, since one NSSI can be either dedicated to one NSI or shared by more than one NSIs, the alarm notifications for the fault management shall be transported to the level of NSMF for the corresponding NSIs.

6.3.2 Multi-service placement

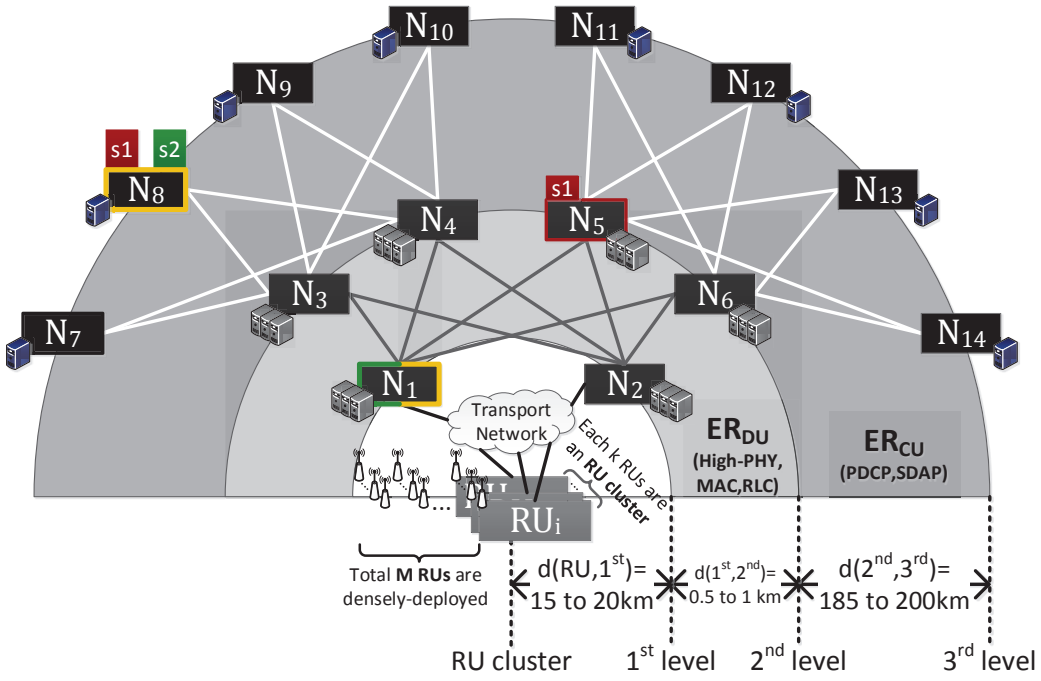
Once RAN service chains are composed, the associated network functions are placed accordingly while respecting the service requirements. Such requirements are described in terms of resources (e.g., compute, memory, networking) and performance (e.g., average throughput, maximum latency). The placement also considers objectives such as cost/power/resource optimization imposed by the infrastructure provider. Finally, according to the placement strategies, several actions can be taken when scaling the RAN service. For instance, more nodes (i.e., DU/CU) and links (i.e., fronthaul/midhaul) can be provisioned to increase the horizontal scalability (i.e., scale-in/out) or more resources are added on the available nodes to increase the vertical scalability (i.e., scale-up/down). These two operations as well as their applicable scenarios will be explained in more details in Section 6.4.

To properly place these shared and customized network functions, we propose a *two-stage placement algorithm* to place slice service chains composed in both horizontal and vertical manners. In principle, the shared network functions are placed first, followed by the customized ones of each slice while retaining the requirements to compose a complete RAN service chain. The algorithm is extended from the multi-objective placement (MOP) one described in [228] with further consideration on the placement of both customized and shared processing. More practically, four following steps are included:

- **Step 1**: For each shared network function with distinct latency constraint in the service chain, determine its eligible regions corresponding to the set of RAN nodes that satisfy the latency requirements.
- **Step 2:** Determine the candidate group that comprises the nodes from the eligible regions satisfying the remaining slice requirements.
- **Step 3:** Select the best node among candidate group based on the considered operator objective, e.g., load balancing.
- **Step 4:** Repeat the above three steps to place the customized network functions based on the results of the shared chains.

An function placement example is shown in Figure 6.8 when placing the service function chain of CU ($\{\text{SDAP}, \text{PDCP}\}$) and DU ($\{\text{RLC}, \text{MAC}, \text{High-PHY}\}$) using 14 nodes (i.e., $\{N_1, \cdots, N_{14}\}$) that are categorized into 3 different levels: $\{N_1, N_2\}$, $\{N_3, \cdots, N_6\}$ and $\{N_7, \cdots, N_{14}\}$. These 14 nodes are served as either DU and CU to serve a number of densely-deployed M RUs that are grouped into several RU clusters. In specific, each RU cluster contains k RUs that are associated with a pair of DU and CU for the centralized processing. To this end, there are M RAN service chains to be placed and every k chains shall be placed with a common DU and a common CU.

The algorithm first selects the eligible regions based on the latency constraints of service function chains to be placed at CU and DU (cf. Step 1 in Figure 6.8). Then, candidate groups are formed based on the processing requirement of the given function and the available processing resource (e.g., number of CPUs) at each node (cf. Step 2 in Figure 6.8). The next step is to select the best node to place the shared RAN service functions chain (cf. Step 3 in Figure 6.8). Afterwards, the customized one for each slice is placed based on the previous placement results of Step 3 using the same algorithm (cf. Step 4 in Figure 6.8). As the input and output endpoints are not slice-customized (cf. Figure 6.7), an extra RTT is included when



Step 1: Eligible region of DU = {N_1,N_2,N_3,N_4,N_5,N_6}
Eligible region of CU = {N_7,N_8,N_9,N_{10},N_{11},N_{12},N_{13},N_{14}}

Step 2: Candidate group of DU = {N_1,N_2}
Candidate group of CU = {N_7,N_8,N_9,N_{10},N_{11},N_{12},N_{13},N_{14}}

Step 3: Best node of shared DU = N_1
Best node of shared CU = N_8

Step 4: Best node of dedicated DU of slice 1 = N_5
Best node of dedicated CU of slice 1 = N_8
Best node of dedicated CU of slice 2 = N_8

Overall path for each slice in the downlink direction
Slice 1: N_8 → N_1 (via N_3 or N_4)→ N_5 → N_1→ RU
Slice 2: N_8 → N_1 (via N_3 or N_4)→ RU
Slice 3: N_8 → N_1 (via N_3 or N_4)→ RU

Figure 6.8: Example of a two-stage function placement for multi-service

computing the eligible region for the customized processing. Taking the DU in Figure 6.8 as an example, the placement of customized functions of slice 1 (e.g., RLC, MAC) shall preserve the latency constraint with respect to the remaining shared network functions in the service function chain (e.g., Input, High-PHY, Output). Thus, the RTT between N_1 and N_5 is considered when placing the customized function at N_5 (cf. the overall path in Figure 6.8).

Table 6.1: Slice context maintained by the RAN runtime

Resource heterogeneity index	Number of CPUs at each node													
	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}	N_{13}	N_{14}
1	32	32	32	32	32	32	2	2	2	2	2	2	2	2
2	48	16	32	32	32	32	1	1	1	1	3	3	3	3
3	32	32	48	48	16	16	1	1	1	1	3	3	3	3
4	48	16	16	16	48	48	2	2	2	2	2	2	2	2
5	48	16	16	16	48	48	3	3	3	3	1	1	1	1

6.4 Performance evaluation

In this section, we analyze the performance of the proposed multi-service chaining and placement algorithm using the performance utilization ratio and acceptance ratio as performance metrics. To highlight its capability, we consider a particular ultra-dense RAN deployment with a multitude of RAN service chains to be placed. Moreover, our simulation is done in the MATLAB platform based on the processing time measurements obtained from the OAI platform. Finally, several auto-scaling actions are highlighted in response to the network dynamics to increase the acceptance ratio.

6.4.1 Experiment scenario

We consider the same three-level infrastructure topology shown in Figure 6.8 with the following inter-level distances: 15 km from the densely-deployed RUs to the first-level nodes (cf. d(RU,1^{st})=15 km in Figure 6.8), 0.5 km from the first level nodes to the second level nodes (cf. d(1^{st},2^{nd})=0.5 km in Figure 6.8), and 185 km from the second level nodes to the third level nodes (cf. d(2^{nd},3^{rd})=185 km in Figure 6.8). A subset of M RUs are grouped together into RU cluster, where each cluster forms the minimum placement granularity for a given chain as mentioned in Section 6.3.2. For instance, if 6 RUs are grouped together, then 6 service function chains are placed simultaneously across DU and CU such that they are physically co-located within the same node. These co-located chains belonging to the same RU cluster can facilitate the real-time control and coordination such as joint processing to enable the coordination and cooperation feature (e.g., CoMP), which is important for the ultra-dense RAN deployment to improve the network performance. In contrast, service function chains belonging to different clusters can only opportunistically cooperate in a larger time-scale. Moreover, we apply the same service function chain (i.e., functional split, customized network functions) for each slice as depicted in Figure 6.7. Furthermore, to better examine the impacts of resource heterogeneity, i.e., heterogeneous number of CPUs at each node, five different conditions are examined as shown in Table 6.1, labeled as heterogeneity index 1 to 5. Last but not least, the traffic workload of each slice is considered as following: slice 1 and slice 2 takes 15% and 10% of the overall load supported by each RU cluster respectively, while the remaining load (i.e., 75%) is utilized by the shared service function chain.

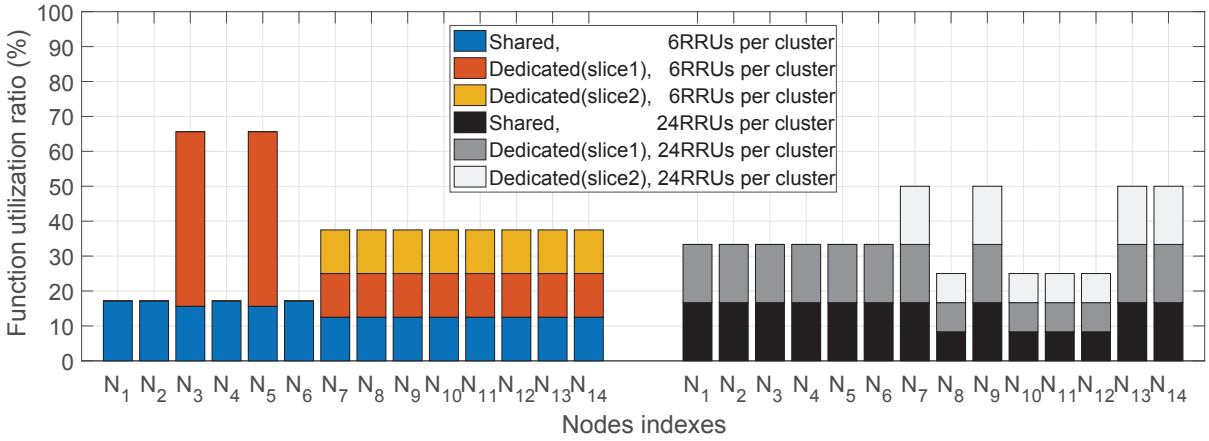
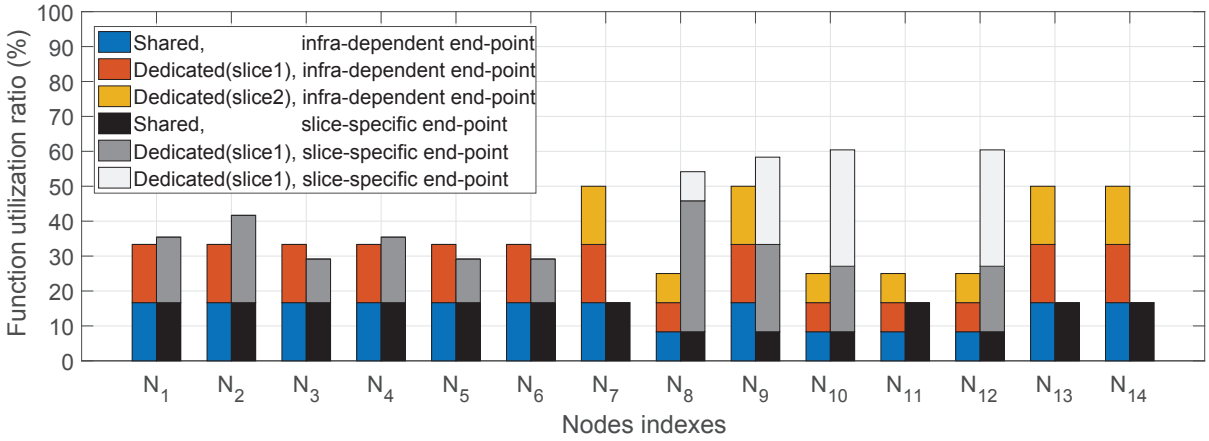


Figure 6.9: Function utilization ratio of shared and dedicated chain for 384 RUs to be grouped in a size of 6 (left) or 24 (right) RUs

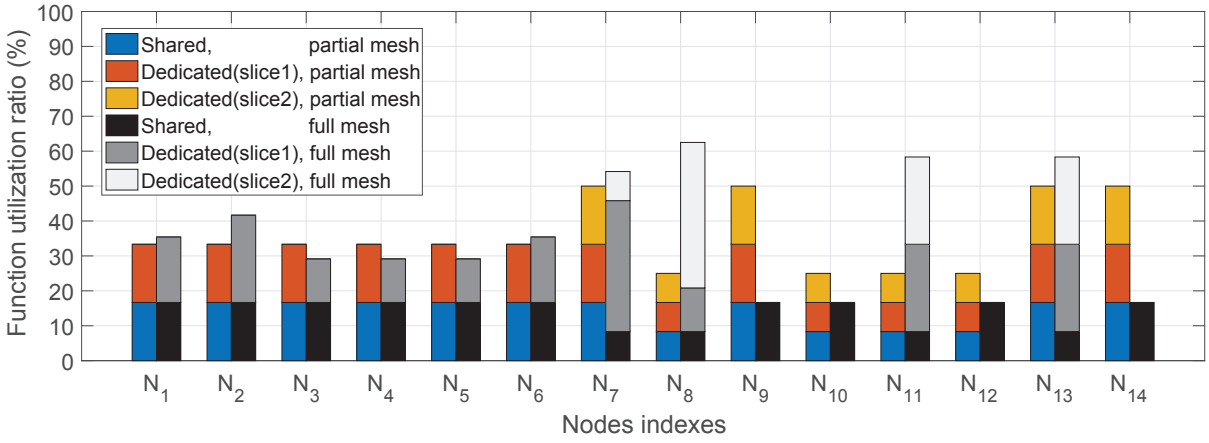
6.4.2 Function utilization ratio

First of all, the function utilization ratio of all 14 nodes (i.e., N_1 to N_{14}) using resource heterogeneity index 1 is shown in Figure 6.9 with 384 densely-deployed RUs that are grouped into two different sizes: i) 6 RUs forming 64 RU clusters and ii) 24 RUs forming 16 RU clusters. Within the results, the shared service chains are evenly distributed among all 14 nodes, with slightly better allocation when grouping 6 RUs as it has a lower level of granularity. In contrast, the dedicated service chain for slice 1 at the second level nodes (i.e., N_1 to N_6) show different trends for two RU cluster sizes. When grouping 24 RUs, the requested resources for DU will restrict the possible placement locations and cause a higher probability to collocate the dedicated service function chain with the shared one, unlike when grouping 6 RUs.

Moreover, the dedicated service chains for both slices 1 and slice 2 at the third level nodes (i.e., N_7 to N_{14}) are uniformly distributed for both RU cluster sizes and are collocated with the shared service chain., i.e., all these seven nodes serve shared and dedicated chain simultaneously. The reason is that the input/output endpoints are infrastructure-dependent, and the RTT between any two nodes of the third level will break the latency constraint of the overall RAN service function chain. This issue can be solved by either enabling the slice-customized input/output endpoints, or provisioning additional links between nodes of the third level. These two possible approaches are examined in Figure 6.10 in comparison with the results of grouping 24 RRUs. The first approach shown in Figure 6.10 reveals that the dedicated service functions have a higher chance to be not co-located with the shared ones, e.g., N_7, N_{11}, N_{13}, and N_{14} do not contain any dedicated ones. Concerning the second approach, we compare the results of partial mesh (the one used in Figure 6.8) with the fully mesh (i.e., all layer three nodes are interconnected) and the result shows that some nodes will only contain shared service chain, such as N_9, N_{10}, N_{12} and N_{14} only contain the shared function in both cases, while N10 shows differently in both cases. To sum up, these two approaches can not only increase the resource efficiency via utilizing the unused resource but also introduce placement flexibility.



(a) Compare infrastructure-dependent and slice-specific end-points

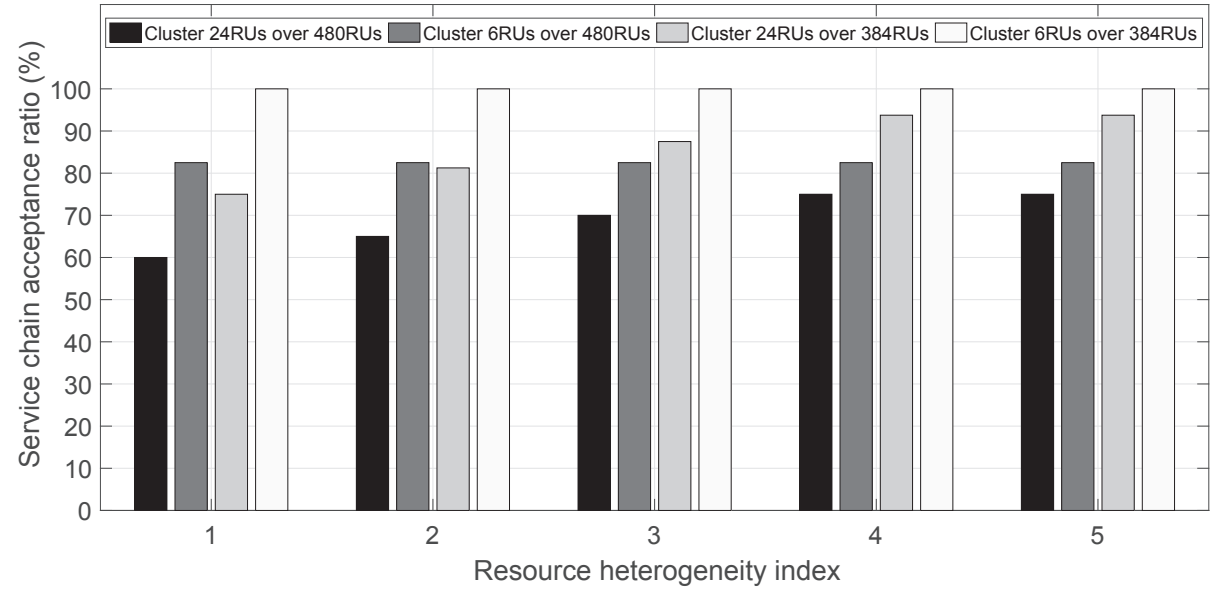


(b) Compare partial mesh and full mesh

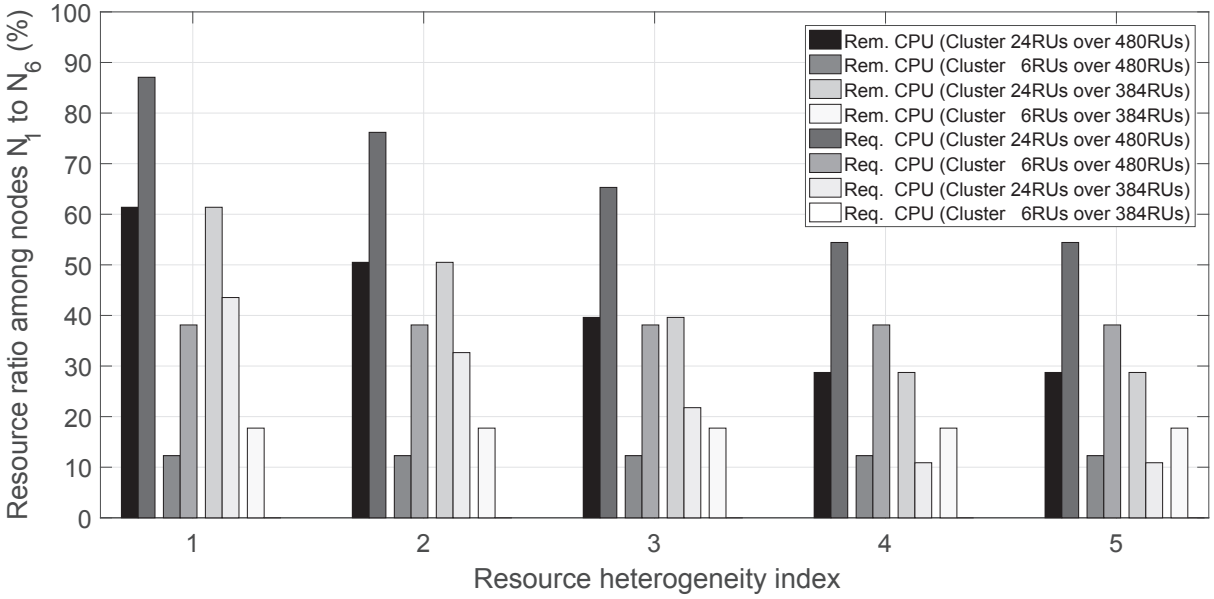
Figure 6.10: Function utilization ratio for grouping 384 RUs in a size of 24 RUs

6.4.3 Acceptance ratio

In Figure 6.11a, the acceptance ratios to place 384 chains are 100% when grouping 6 RUs and 75% when grouping 24 RUs for the resource heterogeneous index 1. Such results justify the efficiency of our proposed algorithm and the higher acceptance ratio benefit when using a smaller RU cluster, due to its smaller granularity as explained before. When the node resources become heterogeneous, the orchestrator shall incorporate auto-scaling action to efficiently manage both infrastructure and slice workload dynamics. Hence, Figure 6.11a illustrates the acceptance ration of all five different resource heterogeneity indexes (cf. Table 6.1) to place 384 or 480 service chains for two different RU cluster sizes. When grouping 24 RUs, heterogeneity indices 4 and 5 provide the largest enhancement as the DU functions located at the second level nodes (i.e., N_1 to N_6) consume more resources and better exploit the resource heterogeneity, while fewer enhancements are observed when RUs are grouped by 6.



(a) Acceptance ratio



(b) Remaining and required CPU ratio

Figure 6.11: Acceptance ratio and remaining/required CPU ratio for different resource heterogeneity indexes

To determine which action shall be taken to further increase the acceptance ratio, we compare the number of remaining resources at all second level nodes after placement (i.e., unused resources) and the number of unsatisfied requested resources (i.e., resources of rejected chains). We can observe that the remaining CPUs is more than the requested ones in the case of 384

RUs clustered in 24 for all resource heterogeneity indexes, which shall trigger a scale-up action[4] to reallocate the unused resources to a subset of nodes to increase the acceptance ratio. In contrast, a scale-out action[5] shall be triggered to provision more nodes in the case of 480 RUs clustered in both 6 and 24, as the remaining resources are less than the requested ones. Note that these aforementioned auto-scaling operations shall also provision the networking resource such as the interconnection between nodes if the infrastructure-dependent end-point is applied, as we discussed in Section 6.4.2.

To sum up, two strategies can be applied to enhance the acceptance ratio: (a) utilize a smaller cluster size of RUs as the minimum placement granularity, or (b) provision heterogeneous resources based on the service requirements to preserve scalability. The former requires adaptation when generating the slice template, while the latter needs an agreement between the slice owner and the infrastructure provider for the pricing model when different scaling actions are taken. However, the infrastructure provider can update the shared service chain (e.g., change the functional splits) to optimize the resource utilization across different nodes, but it will impact every slice and thus shall be planned on a larger timescale.

6.5 Discussions

In this chapter, our aim is to further extend the proposed RAN runtime slicing system in order to cooperate with the defined network orchestration and management function by other standardization bodies, e.g., 3GPP CSMF, NSMF and NSSMF. To further extend our work in this chapter, following directions are highlighted.

The first direction is to identify the necessary messages for all `I1` to `I4` interfaces. These defined messages can be used together with the management system (e.g., 3GPP) specific messages not only to compose both shared and dedicated RAN-domain service orchestration and management but also to expose the available RAN runtime services introduced in Chapter 5 toward different network slices.

Another direction is to quantitatively investigate the performance impacts of sharing and dedication of network functions on the composition of the service function chain. On one hand, the sharing of a network function may degrade the network function performance when there are a number of tenants. On the other hand, the performance of dedicated network functions may differ from the shared ones, which is not discovered in this section. To this end, an in-depth investigation on the function sharing and dedication can provide a more sophisticated management scheme.

Finally, the last direction is to consider the interactions between the other domains, e.g., CN, in order to orchestrate an E2E network slice. As mentioned in Chapter 2, several states-of-the-arts are already initiated; nevertheless, there are few studies aiming to consider the RAN evolution in terms of disaggregation (i.e., CU, DU, RU) and heterogeneity (i.e., Wi-Fi, LTE, 5G). These RAN evolution directions will need more complex network management operation and a better fine-tuning capability.

[4] Conversely, the scale-down action does the inverse operation.
[5] Conversely, the scale-in action does the inverse operation.

6.6 Conclusions

In this chapter, we leverage the proposed RAN runtime slicing system to support 3GPP-aligned slice orchestration and management. We identify a number of new interfaces to enable the communication between orchestration and management systems with the RAN modules through the RAN runtime and to facilitate the customized dedicated orchestration logic for each slice. Moreover, we present the slice modeling approach and introduce the process of splitting and placing RAN network functions to form the shared or dedicated function chain. Finally, we evaluate our approach in a disaggregated UDN deployment scenario for different levels of resource heterogeneity and show the benefits of the auto-scaling mechanism to increase the service acceptance ratio.

Chapter 7

Enabling Control Applications for Multi-Service RAN Programmability

7.1 Introduction

Through the previous two chapters, we provide the RAN runtime slicing system and highlight how it can interact with the standardized orchestration and management system (e.g., 3GPP). Hence, the RAN runtime is shown to be the core component that not only exposes different levels of isolation and sharing to each network slice but also facilitates the chaining and placement operations for multiple services. As the next step, one critical challenge is related to customize programmable logics that monitors and control the operation and performance of each network slice. This logic programmability is the ultimate characteristic to extend a slice into a true self-contained and self-controlled virtual network so as to fully unleash the potentials of network slicing. To this end, RAN runtime is extended to provide an environment to facilitate the development and deployment of programmable control logics.

One common raised issue to apply the slice-customized CLs is how these logics interact through the RAN runtime toward underlying RAN module. To properly address this issue, the "plug-and-play" (PnP) scheme can be applied, in which these slice-customized CLs can be plugged as control applications utilizing a programming interface to interact with the provided RAN runtime slicing system with little or no intervention by either slice owner or infrastructure provider. Further, the RAN runtime slicing system shall possess the capability of recognizing and adapting itself to play the plugged control applications. Furthermore, a conflict resolver within the RAN runtime is necessary to resolve conflicts from multiple control applications. Toward these ends, one effort of this chapter is to provide the *runtime SDK* primitive over the sliced RANs to enable flexible, customized, and PnP control applications. Such SDK provisioning notion is also highlighted by some related works. The network store concept introduced in [158] features the developed network functions and applications for each slice utilizing the provided SDK. In [235], a Python-based SDK empowers network applications development exploiting several programming primitives through representational state transfer (REST) or Python API. The SONATA SDK [236] supports the programming model and software tools for service developers to define complex services. In comparison, the provided *runtime SDK* in this chapter further explores the RAN runtime services as previously mentioned in Chapter 5 for the control application development and deployment.

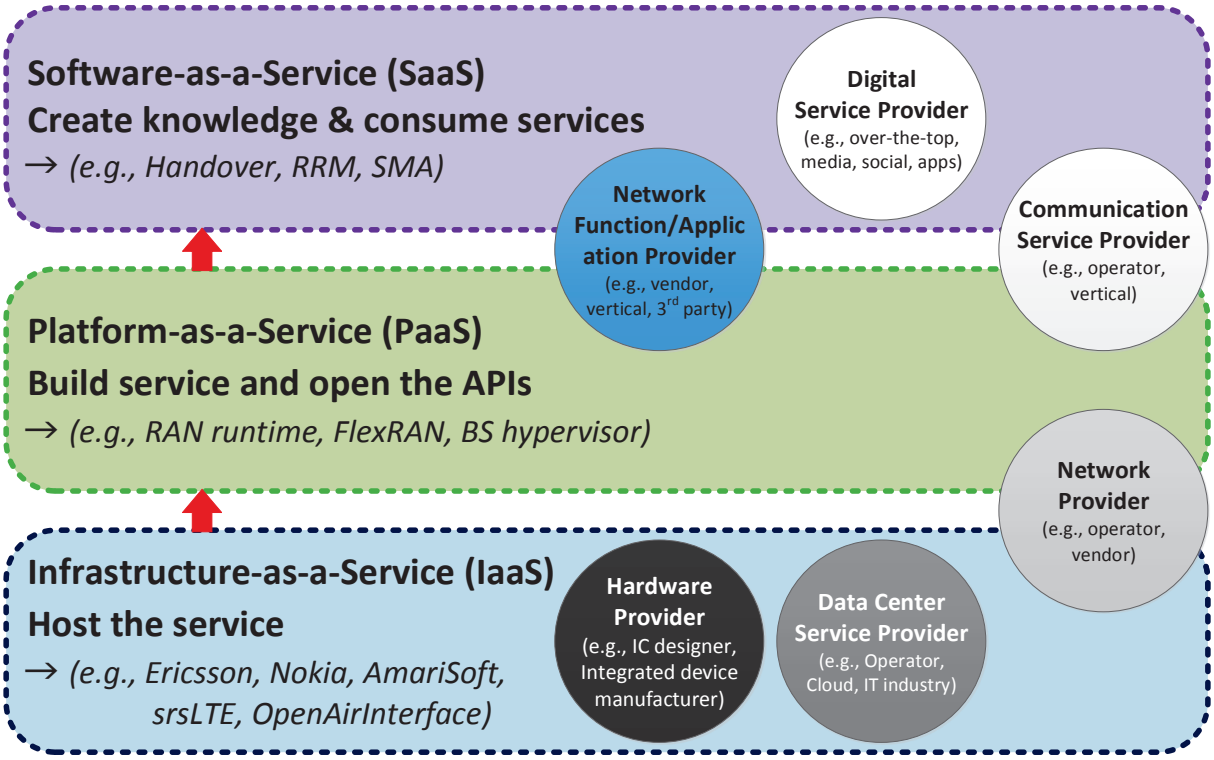


Figure 7.1: Three as-a-service levels among different providers in the value chain

Beyond the aforementioned potentials of control application, another point is to leverage these developed control applications to generate sophisticated control logics across domains for the service providers. For a better understanding, we take the different providers in the value chain of the telecommunication industry from Figure 5.1 and depict the three *as-a-service* levels illustrated in Figure 7.1. These three levels are observed among different providers. First, the Infrastructure-as-a-Service (IaaS) can provide the programmable physical and/or virtual infrastructures (e.g., software-defined radio and x86-based infrastructure) to host the RAN services, ranging from commercial (e.g., Nokia, Ericsson, AmariSoft) to open source (e.g., srsLTE [237], OpenAirInterface [164]). The Platform-as-a-Service (PaaS) extends the IaaS approach in support of monitoring, control, orchestration, and network function virtualization and provides open APIs and the slice-friendly development environment. The aforementioned FlexRAN [154], BS hypervisor [213] and RAN runtime belong to this category. The Software-as-a-Service (SaaS) consumes the programmable control applications, such as the RRM and SMA, to provide the CLs. For instance, the programmability of spectrum management and RRM can allocate available spectrum and specific radio resources to deliver the mobile broadband with the certain QoE levels to the OTT digital service. Finally, a service can be built by means of composing all three as-a-service levels, e.g., programming the CLs through a number of control applications as SaaS, integrating the multi-vendor PNF/VNF for CP/UP processing as PaaS, and exploiting underlying common and/or specialized cloud infrastructures as IaaS.

In correspondence with the above points, a control application execution environment is necessary leveraging the runtime SDK and the application plane over the RAN runtime.

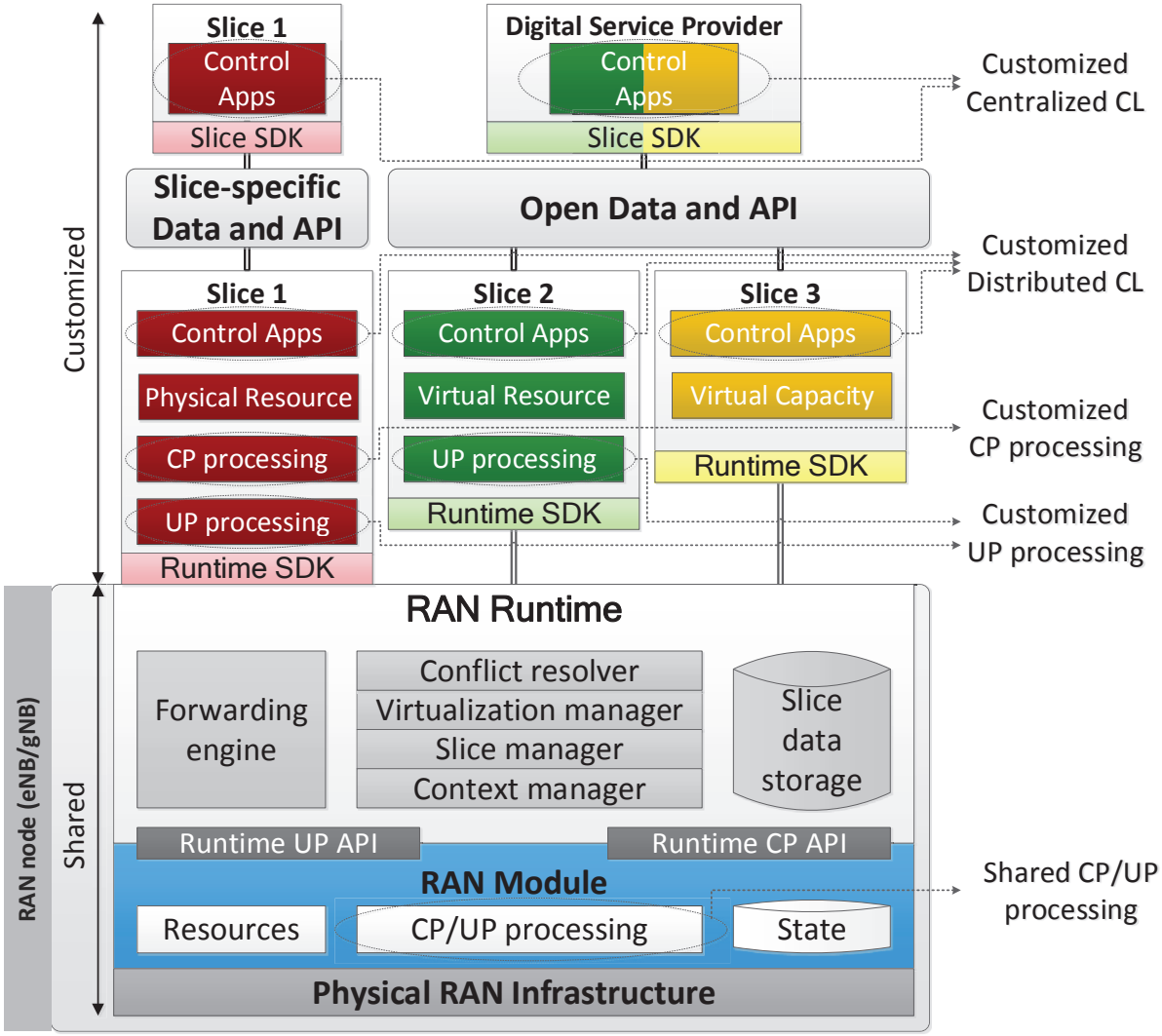


Figure 7.2: Enabling control applications over RAN runtime

7.2 Exploiting RAN runtime for multi-service programmability

As introduced in Chapter 5, the RAN runtime can provide a multi-service execution environment targeting flexible customization and sharing at the RAN node, i.e., LTE eNB or new radio next generation Node B (gNB). Then, each running slice can develop, and plug a bundle of control applications on top of the runtime SDK, interact with the RAN modules to access its resource/state, and control the underlying behaviors, as shown in Figure 7.2.

7.2.1 Overview of RAN runtime slicing system for control logic customization

As mentioned in Chapter 5, the RAN runtime can allow the slice owners to (a) orchestrate and manage their slices, (b) perform customized CLs (e.g., handover decisions) and/or CP/UP processing, (c) operate on a set of virtual resources (e.g., resource block), performance indicator

(e.g., data rate or latency), and (d) access their CP/UP state (e.g., user identity) revealed by the RAN runtime. Also, the RAN runtime enables the infrastructure provider to manage the underlying RAN module, enforce slice-specific policies, perform access control, and provide the BS-common (i.e., slice-independent) services to users. To reach these goals, five different RAN runtime services are enabled as shown in Figure 7.2: Context manger, Slice manager, Virtualization manager, Conflict resolver and forwarding engine.

To enable the CL customization, we highlight the following functionalities provided by two RAN runtime services:

- The *virtualization manager* can reveal the virtual resources and states through partitioning, abstraction and accommodation, as the slice-specific view. Hence, customized control logics can be executed over its virtualized network view. For instance, the slice-specific intra-slice resource scheduling mentioned in Chapter 5 will schedule the virtualized resource provided by the virtualization manager.

- The *conflict resolver* can accommodate the customized control logics from different slice-specific control applications, resolve their conflicts, and enforce a feasible policy to underlying RAN module. For instance, when both slice 1 and slice 2 have their customized handover and power control applications that make different control decisions toward the same user equipment, which belongs to both slices. The conflict resolver can either base on the pre-defined policy to directly apply any control decision (e.g., handover the user equipment according to the control decision made by slice 1 control application) or base on the operator's method to generate the solution as a compromise in between (e.g., weighed average the power control decision from both control applications).

Moreover, the customized network slice can be initiated leveraging the exposed runtime SDK and runtime CP/UP API. The former is in the north-bound toward each instantiated slice to allow each slice span its own execution environment, either at the host or guest level, leveraging operating system and virtualization technologies, such as container or virtual machine. A more detailed explanation on the functionalities provided by the runtime SDK is provided in Section 7.3. The latter is in the south-bound toward the underlying RAN module to enable each slice to control and manage its service by requesting radio resources, applying control decisions, and accessing states. As for the runtime UP API, the PDUs of the corresponding network layer are exchanged between the shared and customized processing, possibly including extra headers for flow-based match-action processing done by the RAN runtime.

By exploiting the above two RAN runtime services, the slice-specific control applications can be deployed either in the distributed manner relying on the *runtime SDK* to access the slice-specific resources, states, and processing, or in the centralized manner relying on the data and API that can be slice-specific or open through the *slice SDK*. These data and API can further open the opportunities for service providers to flexibly compose their control logics. For instance, in Figure 7.2, the slice-specific data and API of slice 1 can enable the the centralized control application built on top of distributed control applications. Moreover, the open data and API of slice 2 and slice 3 can expose their (abstracted) RAN network view to the digital service provider for content optimization control application to dynamically adjust the video bit rate. To this end, these centralized applications can be developed and deployed by the digital service provider that consumes the data and API from one or more network slices to further enable a flexible service composition (cf. nested service composition defined by ETSI NFV in Fig 6.1).

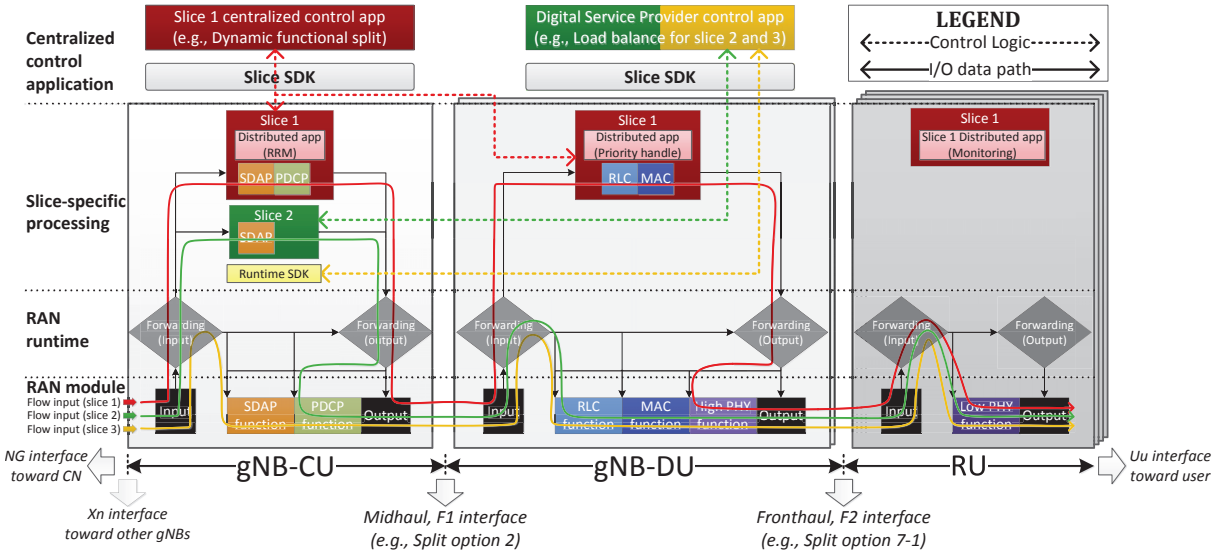


Figure 7.3: An example of multi-service chaining, forwarding and programmability in a disaggregated RAN

7.2.2 Multi-service programmability in a disaggregated RAN

Following the RAN evolution, a disaggregated 5G RAN example is presented in Figure 7.3 with the three-tier RAN nodes (gNB-CU, gNB-DU, RU) following the 1:m:n relationship. Hence, the slice service chain can be split between RU and gNB-DU (i.e., fronthaul with F2 interface [238]) and between gNB-DU and gNB-CU (i.e., midhaul with F1 interface[1] identified by 3GPP in TS38.470 [240]). Note that several other interfaces are identified by 3GPP in TS38.401 [177]: NG, Xn, and Uu interfaces are between the gNB and CN, another gNB, and the user equipment, respectively. The overall service function chain of each slice can be composed horizontally between RAN nodes (gNB-CU/gNB-DU/RU) and/or vertically when customized CP/UP processing is required tailored to service requirements.

Based on the virtualized slice-specific view exposed from disaggregated RAN nodes, the control applications can monitor the underlying RAN information and apply their CLs, subject to the conflict resolver. Take slice 1 as an example. The RRM application at gNB-CU can manage the radio resource, the priority handling application at gNB-DU can dynamically prioritize the channels and users over its customized MAC processing, and the monitoring application at RU can provide real-time RAN information (e.g., channel quality indicator [CQI]). Moreover, the centralized application of slice 1 can update the functional split between the customized processing at gNB-CU and gNB-DU via transporting the slice 1 data among multiple RAN runtime instances. Additionally, the application of a digital service provider can plug and play its CLs to the customized processing (slices 2) and shared processing (slice 3) leveraging the slice SDK on the open data and API from both slices.

[1]It is surveyed as V1 interface for the LTE system by 3GPP in TR37.876 [239].

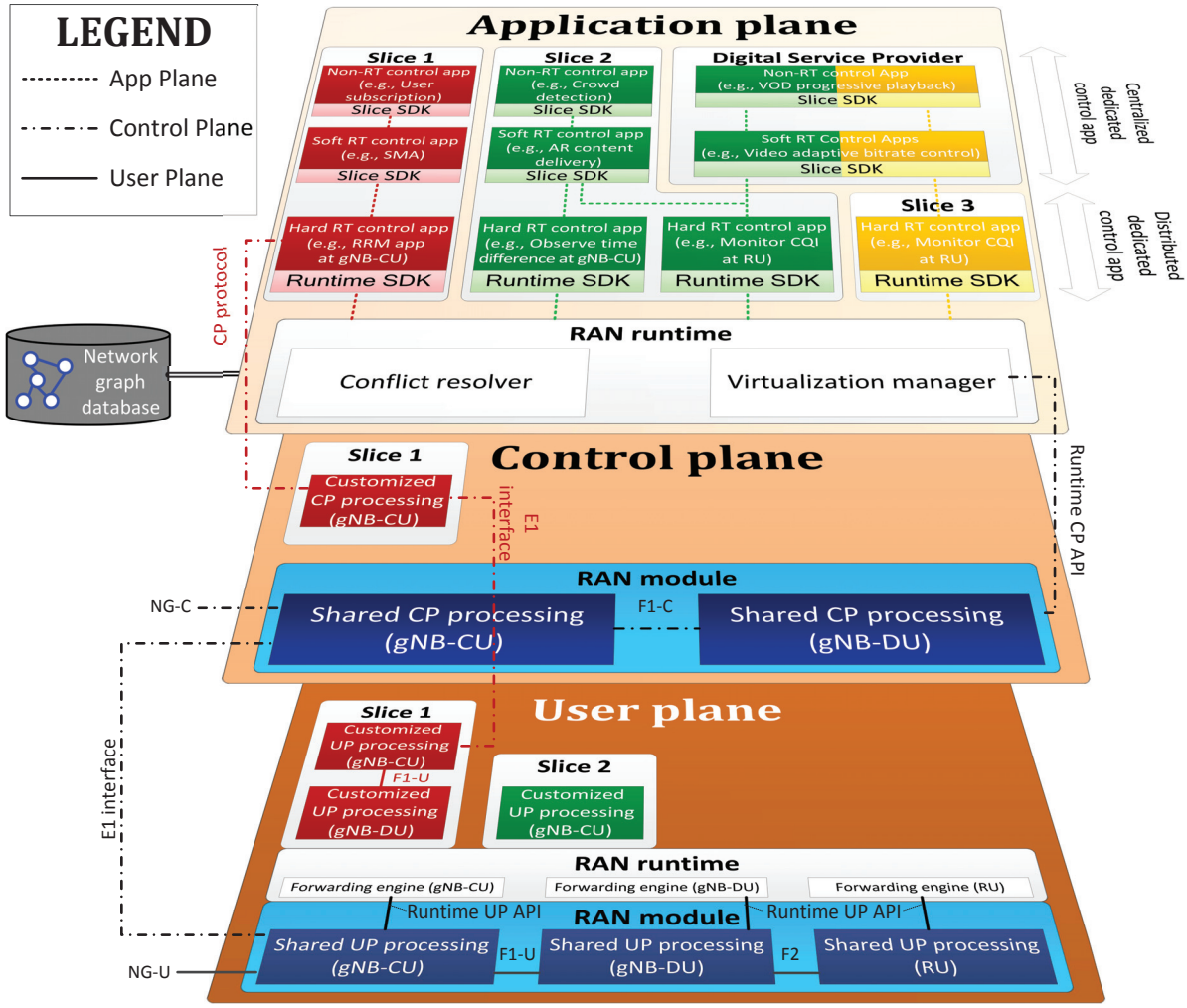


Figure 7.4: SDK and application plane for flexible and programmable RAN control

7.3 Flexible and programmable RAN control

The RAN monitoring and control are performed by chaining dedicated and/or shared control applications, allowing a slice to flexibly (a) process CP/UP states to create and share knowledge and (b) apply actions to the underlying RAN modules. Specifically, in Figure 7.4, two components are critical to support slice-specific control application developments and deployments: (1) Runtime SDK and slice SDK and (2) cross-domain control application chaining.

7.3.1 Software development kits

Generally, SDK provides a software development environment to simplify the design, development, testing and update of applications. It abstracts the underlying network by means of technology-agnostic and technology-dependent APIs and includes a group of libraries to provide specific functions and methods to be accessed through one or more API calls. More specifically,

it can reveal necessary network information over the underlying heterogeneous RAN with multiple RATs and multiple deployment scenarios. Within the architecture depicted in Figure 7.4, a two-level abstraction view of the underlying RAN entities is provided through both runtime SDK and slice SDK. As mentioned beforehand in Section 7.2.1, such two-level scheme can enable the flexible and extensible service composition.

First, the runtime SDK can expose both high- and low-level APIs. The high-level APIs rely on the RAN runtime services to (1) manage the life-cycle of a virtual BS instance on top of RAN modules, (2) collect monitoring metrics and KPIs, and (3) retrieve/allocate the virtualized state and resource corresponding to a slice-specific network view. In contrast, the low-level APIs utilize the aforementioned runtime CP/UP APIs to access the instantaneous network state information for a specific slice (e.g., the BS configuration and relevant user information), and to modify the CP/UP processing of the underlying RAN modules (e.g., user measurement configuration). Moreover, the second-level abstraction is enabled by the slice SDK to facilitate the extensibility and coordination among control applications spanning different technology (e.g., RAN and CN) and administrative domains (e.g., communication and digital services providers), corresponding to open or slice-specific data and APIs, as shown in Figure 7.2. Such slice SDK can also expose context-aware semantic information [241] to facilitate the reasoning of actionable knowledge from heterogeneous information sources and foster interoperability among a variety of applications.

More specifically, we elaborate the following capabilities provided by the runtime SDK to enrich the advanced functionalities of control applications:

7.3.1.1 Authentication, authorization and management

Before utilizing any API calls, the application needs to be authenticated. Moreover, the slice-customized application must be authorized by the RAN runtime when accessing the slice-specific state information. Hence, it must provide its credential (or granted access token) and the identities of the target users and BSs for authorization. However, no authorization is needed when accessing the public information, such as a globally unique cell identity. Besides, a slice management operation can handle the life-cycle management of a network slice. It can also dynamically upgrade or downgrade the overall slice to a new profile, involving the related adaptations in terms of QoS control, PNF/VNF configuration, and so forth.

7.3.1.2 Registry and discovery

The SDKs provide the functionality for control applications to register and to discover other control applications to compose sophisticated control logic. Such functionality enable the new control application to flexibly exploit the status from existing control applications to establish their relationship on-demand and to enable the PnP characteristic. To this end, these control applications can be loosely-coupled and can achieve the benefits of scalability, reusability and agility. More explanation will be given in Section 7.3.2.

7.3.1.3 States monitoring

This provides APIs for monitoring slice-related states over multiple granularities according to the exposure level (e.g., resource, network function and application, slice, and service levels) and enables PnP applications for runtime control and adaptation. Furthermore, monitoring metrics of the corresponding cell/slice/user can be retrieved after authorization.

7.3.1.4 Network graph database

This provides the graph-based primitives (e.g., split or merge) to operate on the network information, which can efficiently model, traverse and correlate more complex and dynamic relations between densely deployed RAN nodes. Moreover, it can naturally support the multi-tenant application through graph partitioning into subgraphs for multiple substrates (e.g., multi-domain or multi-service). Hence, each application can perform its graph-based operations (e.g., shortest path) that take node relationships in time series into account in its abstracted network view.

For instance, the network graph shown in Figure 7.5 depicts three slices in a disaggregated RAN deployment. The disaggregated RU (i.e., RU_1 to RU_3), DU (i.e., DU_1, DU_2), and CU (i.e., CU_1) are virtualized for three different services to serve five users (i.e., u_1 to u_5). For instance, RU_1 is virtualized for slice 1 and slice 2 as $RU_{1,1}$ and $RU_{1,2}$, respectively. The edges between the vertices can represent their relations and be used for different control applications. For example, the edges between a user and a virtualized RU, as in $u_1 \leftrightarrow RU_{1,2}$, can represent the measured CQI or traffic metrics. Such a subgraph can be utilized as input for handover or traffic-steering applications. Additionally, the edges between disaggregated RAN entities, as in $CU_{1,1} \leftrightarrow DU_{1,1} \leftrightarrow RU_{1,1}$, capture their association and the applied functional splits for multi-service chaining and placement application. Furthermore, the edges between virtualized instances within the same physical RAN node, as in $RU_{2,2} \leftrightarrow RU_{2,3}$ and $DU_{2,2} \leftrightarrow DU_{2,3}$, show their relations in terms of sharing (e.g., multiplexing) and priority (e.g., preemption) that can be used for resource management, while the edges between different physical entities, as in $RU_2 \leftrightarrow RU_3$, depict the policy of cooperation (e.g., spectrum sharing) or constraint (e.g., exclusive coexistence) that can be combined with other graphs (topology graphs, for example) for dynamic radio spectrum management. In summary, the graph database can naturally represent complex relations in a monolithic/disaggregated RAN deployment and be partitioned or combined for control applications among multiple substrates.

7.3.2 Single- and cross-domain application chaining

The communication between control applications can produce several benefits: it can (1) structure an application as a collection of loosely-coupled[2] micro-services [242], (2) synchronize applications' status when cooperation is needed, and (3) allow the implementation of a common interface for different applications. An example of inter-application communication for three control applications is shown in Figure 7.6, i.e., App1, App2, App3. Both App2 and App3 first register for the communication channel toward App1 and request the capability list from App1. Then, App3 asks for the latest results from the list, and thus App1 responds with the results and notifies App2 of the new status. Also, we provide some example commands using the JSON-RPC 2.0[3] protocol below Figure 7.6.

Leveraging the communication between control applications, various dedicated/shared control applications can be chained together as the application plane shown in Figure 7.4. Such chaining enables the automation and extendibility of the control logics and improves the decision-making process across different slices. Specifically, three categories of applications can be chained: (a) non-real-time applications that enforce CLs when possible or when being instructed by higher layers, (b) soft-real-time applications that require an average delay guarantee within

[2]Note that the tightly-coupled case may still be needed when delay and throughput is required.

[3]`https://www.jsonrpc.org/specification`

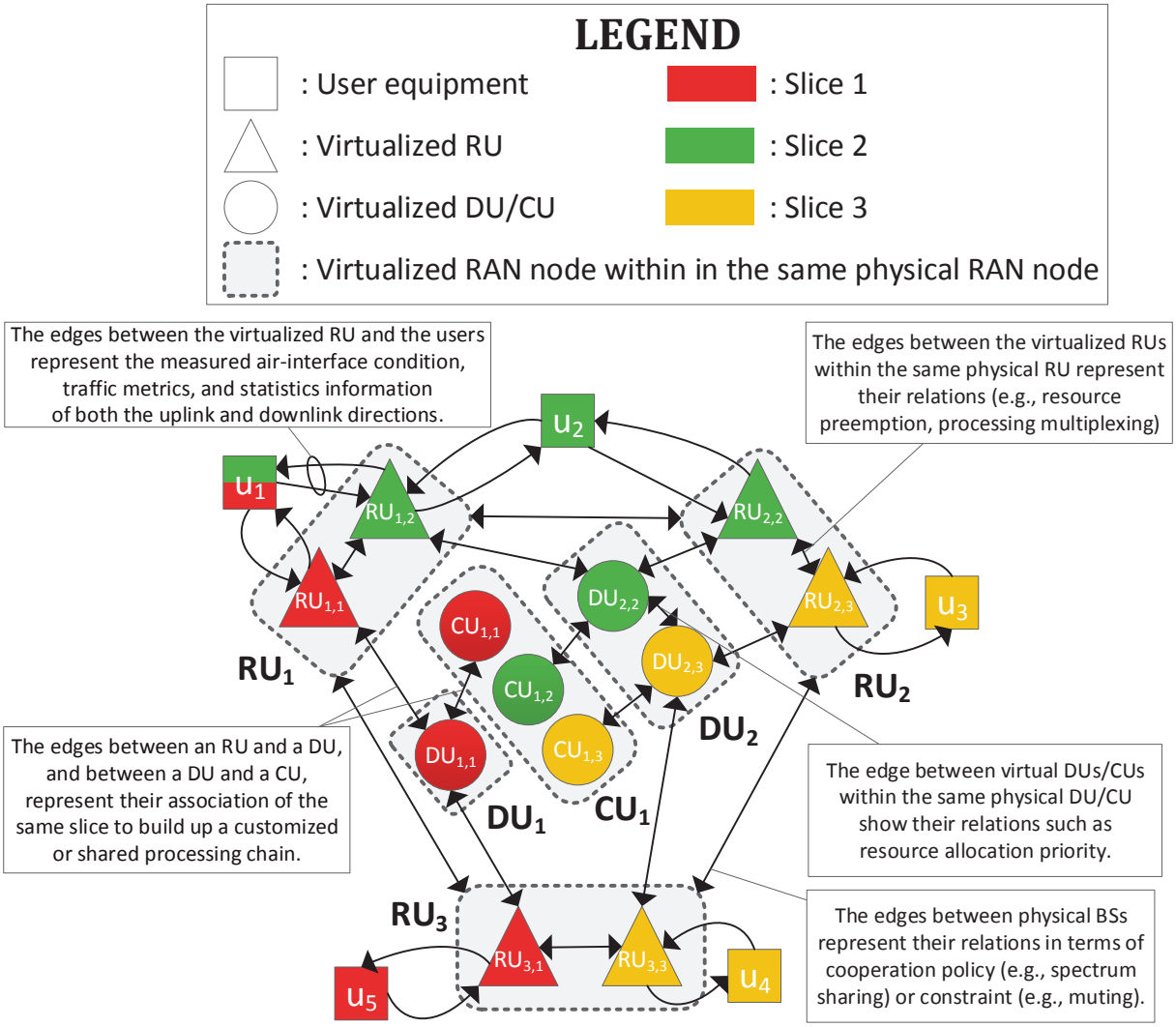


Figure 7.5: An example of a network graph in a disaggregated RAN deployment

a tolerance when performing CLs, and (c) hard-real-time applications that require a delay guarantee when applying CLs, which otherwise cause a performance degradation. An example with three slices is provided in Figure 7.4. For slice 1, the user's subscription information (e.g., user classes) can be utilized to allocate its carrier frequency and radio bandwidth through the spectrum management application (SMA) and to control its radio resource through the RRM application in a single administrative domain (i.e., the same communication service provider). A cross-domain example is also shown in which the digital service provider can offer both a customized video-on-demand (VOD) playback application and a soft-real-time video adaptive bit rate (ABR) control based on the monitored CQI from one or more slices (e.g., slices 2 and 3). Note that such an application plane can span disaggregated RAN nodes, e.g., slice 2 can monitor the CQI at RU and the observed time difference at gNB-CU for augmented reality content delivery and crowd detection applications.

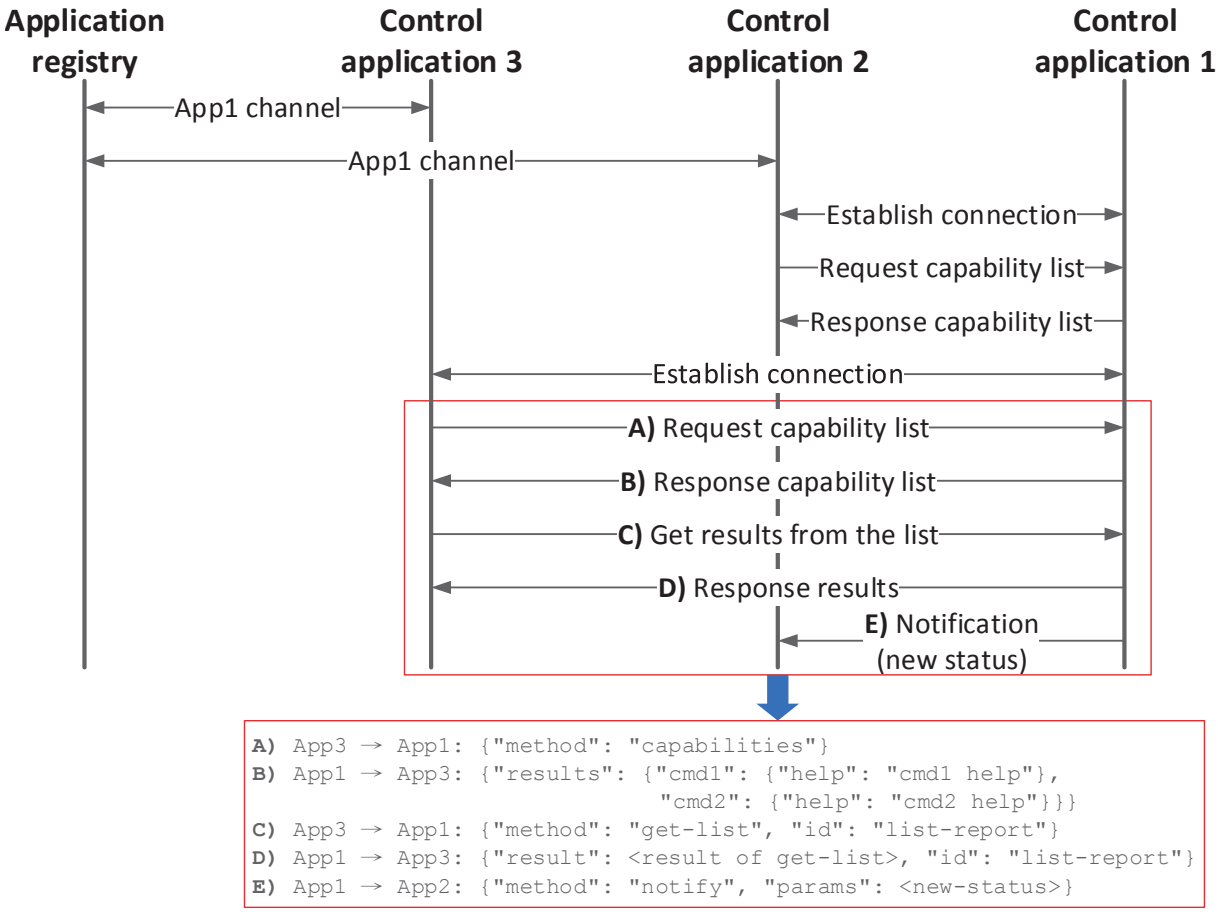


Figure 7.6: Protocol flow for communications between control applications

7.4 Proof-of-concepts

To explore the potentials of control applications over the application plane, we create remote slices using an asynchronous communication channel toward the RAN runtime, embed its control applications, and operate on the virtualized resources and states based on the runtime SDK. In the following, we present the results for three use cases: (a) Spectrum management application, (b) RAN-aware video optimization, and (c) Subscription-aware RAN resource provisioning.

7.4.1 Spectrum management application

The SMA is an efficient tool to manage and process different policies and rules defined by various stakeholders such as national regulatory authorities, operators and licensed shared access. It is a clean-slate replacement for legacy platform-dependent spectrum management solutions and can provide customized control programmability and agile resource utilization. Specifically, it can process several types of input information (i.e., short-term and long-term policies, sensing data) aggregated through well-defined interfaces, and derive spectrum management control decisions, and enforce the applied policies. In Figure 7.7, the processing flow of SMA is shown leveraging the

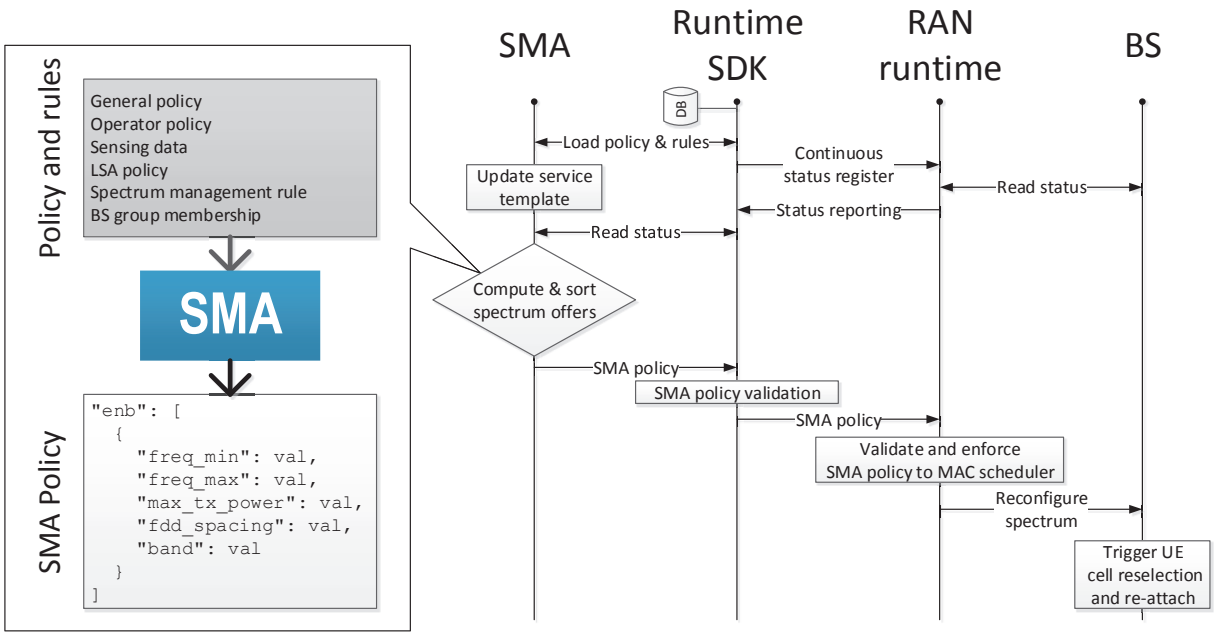
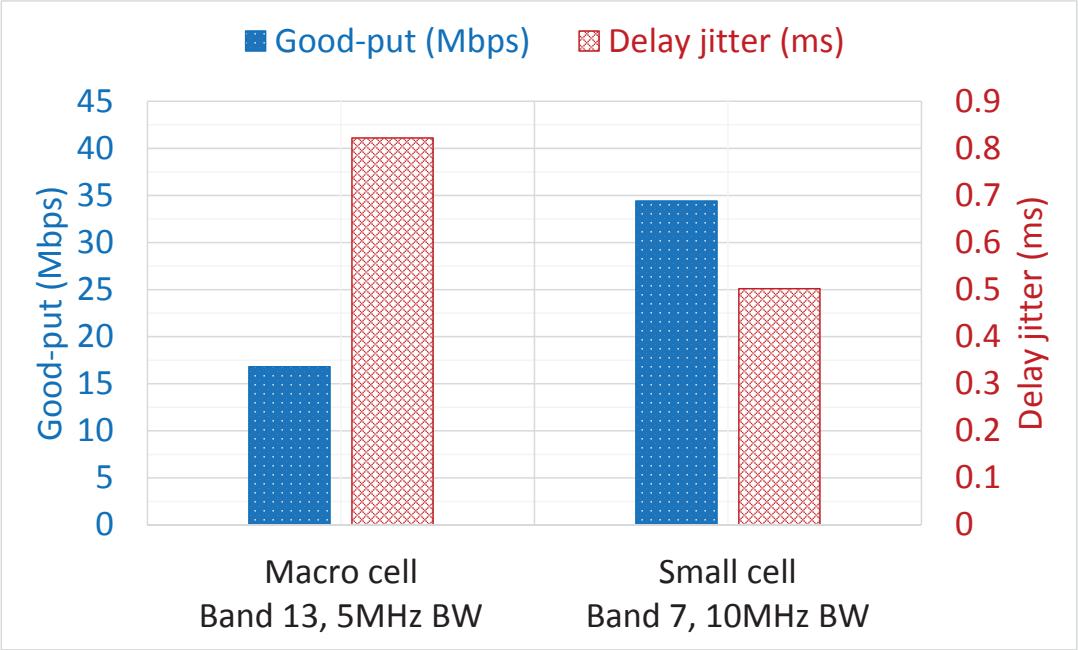


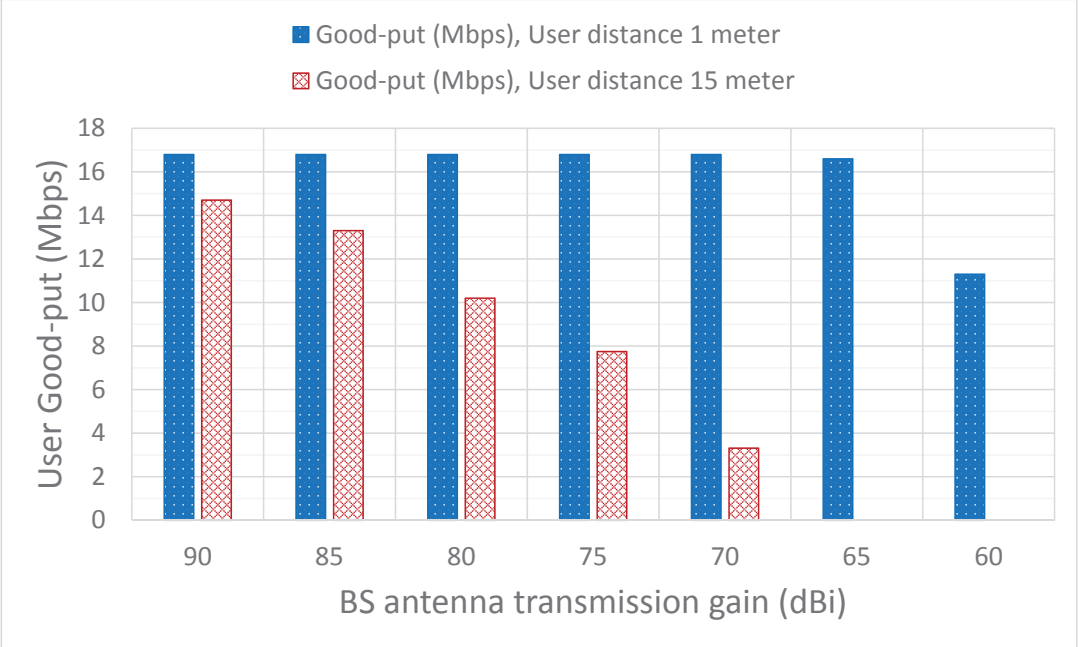
Figure 7.7: Design of spectrum management application

underlying RAN runtime and runtime SDK to build and apply its SMA policy. Specifically, this SMA policy includes (1) minimum frequency (`freq_min`), (2) maximum frequency (`freq_max`), (3) maximum transmission power (`max_tx_power`), (4) spacing between UL and DL direction in FDD mode (`fdd_spacing`), and (5) band identity (`band`). After validation, it will be enforced via reconfiguring the affected BS to soft-restart, and thus the attached UE will do the operations of cell re-selection and attachment.

The implemented SMA can manage the spectrum usage in a real-time manner and we show two representative scenarios: (a) phantom cell and (b) cell zooming. In Figure 7.8a, we examine the phantom cell scenario [243] that deploys small cells for UP transportation at higher frequency band with a larger bandwidth to boost the user data rate. We can observe a better user experience when deploying the phantom cell at a higher band (band 7, 2.6 GHz) with larger bandwidth (10 MHz), compared with the macro cell at a lower band (band 13, 750 MHz) with smaller bandwidth (5 MHz). Note that the SMA can provide the cell reconfigurability in terms of the applied carrier frequency and bandwidth to enable a flexible phantom cell deployment. The second cell zooming scenario [244] aims to adaptively adjust the cell size (i.e., zoom in/out) according to the traffic and user status. Here, a simple cell-zooming mechanism is applied to adjust the transmission power according to the user distance toward the base station shown in Figure 7.8b. We can see few impacts when the user is close to the cell site (i.e., 1 meter); however, distant user (i.e., 15 meter) will suffer significantly and will even loss the connection when we largely reduce the transmission power. To sum up, the SMA can serve as an efficient tool to expose the control logic programmability over the underlying RANs and to fulfill heterogeneous RAN deployment.



(a) Phantom cell scenario



(b) Cell zooming scenario

Figure 7.8: Two scenarios for spectrum management application

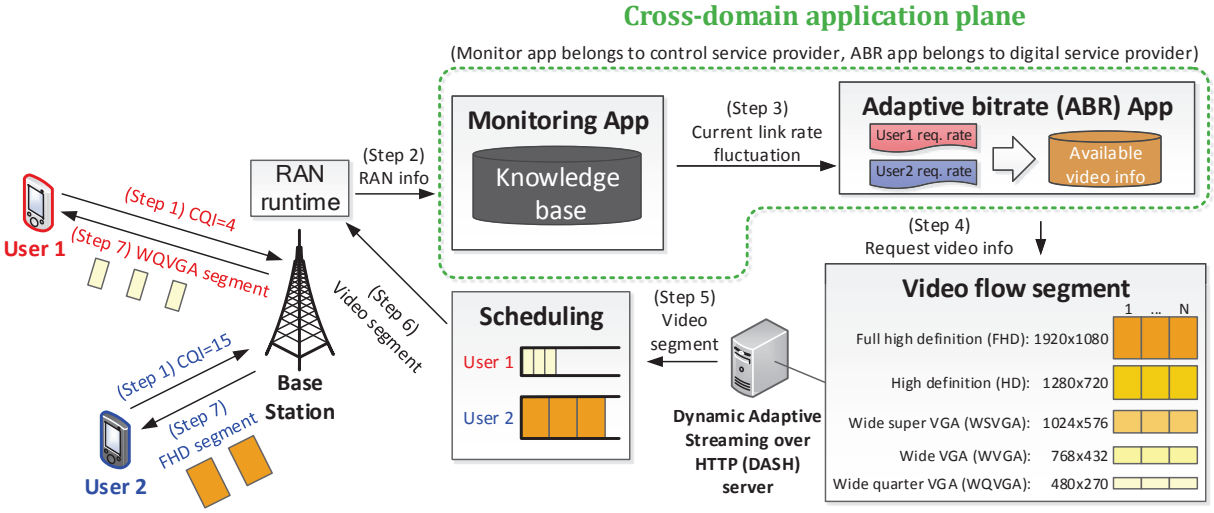


Figure 7.9: Process flow of RAN-aware video optimization use case

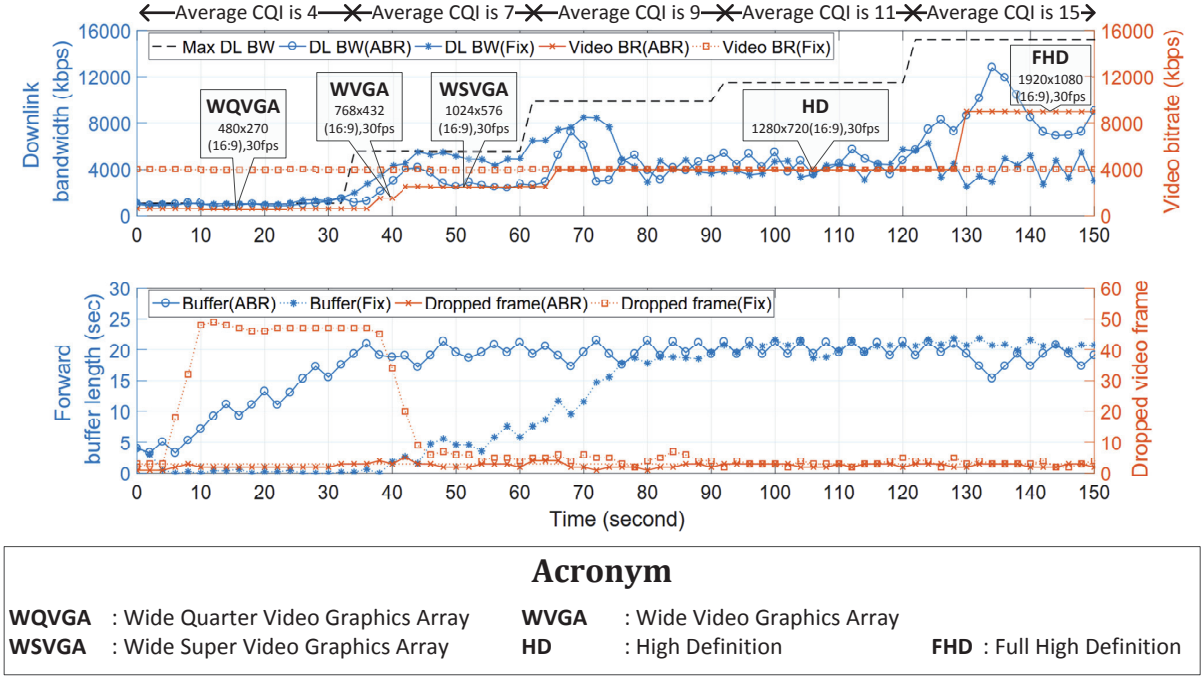
7.4.2 RAN-aware video optimization

In the second use case, we adapt the video bitrate based on the instantaneous RAN information (i.e., user CQI), relying on the chaining of monitoring and ABR applications. A detailed step-by-step description is shown in Figure 7.9. We can see that the RAN information will be mapped to the current link rate (cf. Step 3 of Figure 7.9) as the input to the ABR application. Afterwards, video segments of different qualities will be provided by the dynamic adaptive streaming over HTTP (DASH) server (cf. Step 5 of Figure 7.9) according to the requested rate to maintain the video QoE [245] . Finally, video segments of different qualities will be transported to the corresponding users.

We first measure the maximum user good-put corresponding to different CQI values in both DL and UL directions as shown on the top of Figure 7.10. Afterwards, two experiments are conducted to compare the cases with and without RAN-aware video optimization, i.e., labeled as ABR and Fix respectively, when the user CQI value increases from 4 to 15. The former case shows that the video quality can be adapted from wide quarter video graphics array (WQVGA), wide VGA (WVGA), wide super VGA (WSVGA), high definition (HD) to full HD (FHD)[4], when the user channel quality is improved with fewer dropped video frames. However, the latter case uses a fixed HD video quality irrelevant to the CQI fluctuation. We can observe that a large amount of dropped frames and zero buffer length when the average CQI is low (i.e., when CQI is 4 or 7) and an inferior video quality when the CQI is high (i.e., when CQI is 15). These results reveal that the QoE can be significantly improved when chaining monitoring and ABR applications that belong to two different administrative domains.

[4]Disfferent graphics display resolutions can be found in `https://en.wikipedia.org/wiki/Graphics_display_resolution`

CQI value	Downlink (Mbps)	Uplink (Mbps)
4	1.08	0.69
7	5.59	2.49
9	9.88	4.47
11	11.47	6.04
15	15.22	8.08



Acronym

WQVGA : Wide Quarter Video Graphics Array
WVGA : Wide Video Graphics Array
WSVGA : Wide Super Video Graphics Array
HD : High Definition
FHD : Full High Definition

Figure 7.10: Measured maximum good-put of each CQI value and the experiment results

7.4.3 Subscription-aware RAN resource provisioning

In the next use case, we chain the slice subscription and RRM applications within a single administrative domain to provision different numbers of radio resource for different user classes, following the steps depicted in Figure 7.11. We can observe that the user subscription information can be categorized into five classes that is used explicitly to indicate the maximum consumed radio resources to be applied at the RRM application (cf. Step 3 of Figure 7.11). Specifically, these different user classes have different pre-defined policy profiles that determine the amount of radio resources a user can consume in the DL and UL directions.

To quantify the overall QoE of different user classes, the neprf application [246] is installed on the COTS UE connecting to the OpenAirInterface BS, as shown in Figure 7.12. The super user gets the best score, considering bit rates, delay and jitter, web browsing and video streaming performance rate. The platinum user's score is close to the super users', while there are performance drops for the gold and silver users in web browsing and bit rate. The bronze users suffer from a significant drop in the bit rate, web browsing, and video streaming. Summing up, chaining both user subscription and RRM applications can provision RAN radio resource in a real-time manner.

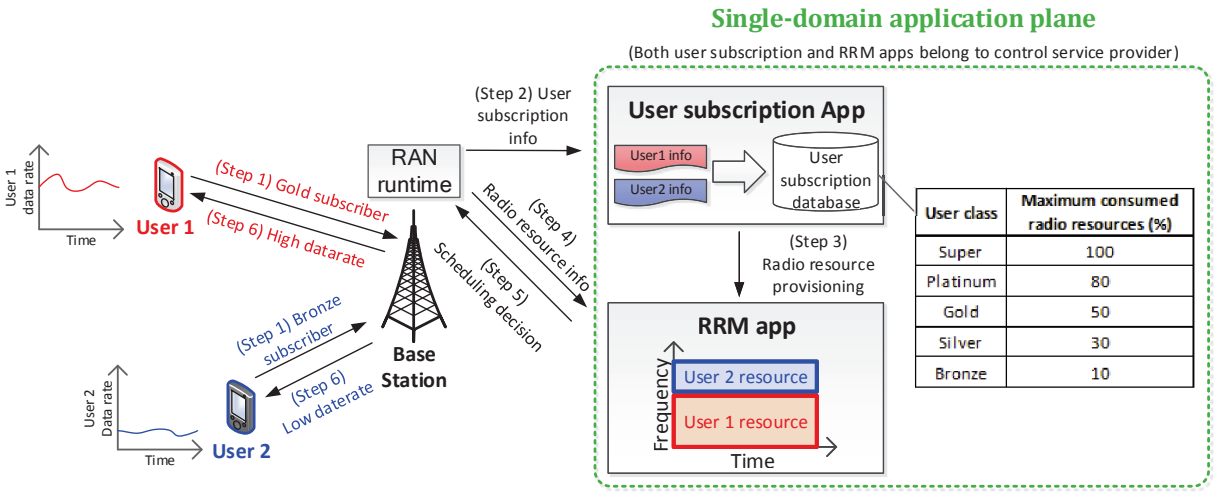


Figure 7.11: Process flow of a subscription-aware RAN resource provisioning use case

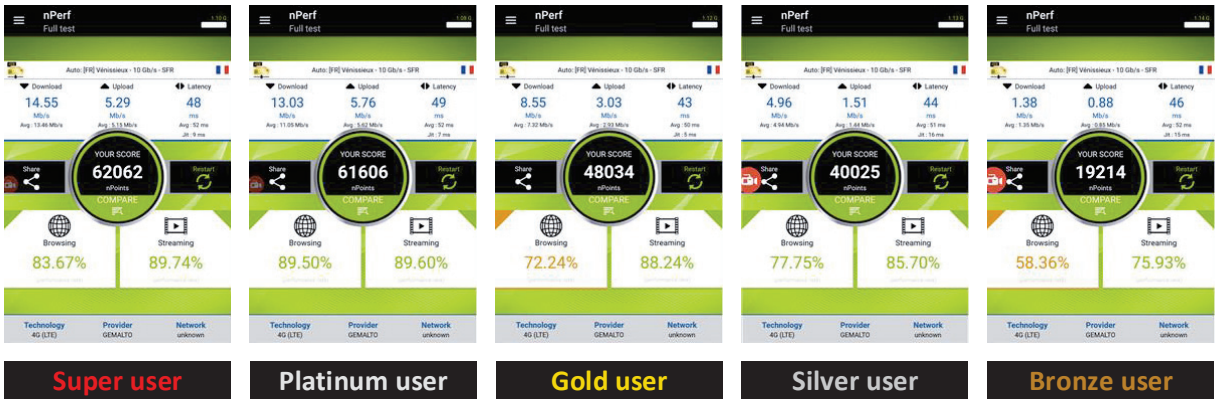


Figure 7.12: Measured QoE for five user classes.

7.5 Discussions

In this chapter, we extend the previously-proposed RAN runtime slicing system to further facilitate the control applications developments/deployments via two key enablers: (1) runtime SDK, and (2) application plane. As the next steps, several possible directions are highlighted and elaborated in the following.

First of all, even with the aforementioned capabilities provided by the runtime SDK; however, there is still no clear view on how the control applications shall be developed to facilitate the deployment on the cloud environments. In legacy, the cloud-based design only deploy on-premise software on the cloud infrastructure without exploiting any advantages of cloud computing. To this end, several cloud-native design principles, e.g., micro-service architecture, shall be further considered when developing the control applications.

Another possible direction is to explore how the generated data and knowledge of one control application can be utilized by others. Currently, only abstraction approach is applied to

expose the necessary network information through the slice SDK without considering any further processing to generate meaningful semantic information. Through generating this semantic information, control applications can be more straightforwardly chained to serve the service performance.

Last but not least, one missing point in this chapter is in terms of the management of these control applications. Like the VNF management scheme provided by the ETSI NFV, the management of these control applications can apply the similar approach but with a more heterogeneous composition. Furthermore, another question show up in terms of how to interact between these different orchestration and management functions to deploying network service.

7.6 Conclusions

In this chapter, we provide the runtime SDK for the development of control applications. To further enable flexible CL programmability, the two-level abstraction concept is proposed, relying on several SDK capabilities and the application plane to chain shared/dedicated control applications. Finally, several use cases are presented over the proposed runtime SDK and RAN runtime to provide sophisticated and customized CLs.

Chapter 8

Conclusions and Future Perspectives

8.1 Conclusions

Nowadays, 5G is right around the corner, and a broad variety of capabilities are anticipated for the intended usage scenarios. In order to facilitate this vision, 5G will provide a paradigm shift beyond just several new radio access technologies for establishing an agile and sophisticated communication system. In essence, the 5G architecture shall be designed with a certain level of flexibility via incorporating the principles of softwarization and virtualization, e.g., SDN and NFV, to compose multiple end-to-end logical or virtual networks that are customized to meet particular multi-service requirements.

In correspondence, the network can be used efficiently and independently via crafting several customized and logically-separated networks for multiple services. In addition, each logical network can be based on a combination of different physical deployments (monolithic and disaggregated) leveraging the cloud environment to deploy relevant network functions, while still satisfying specific service requirements. To this end, this thesis investigates the RAN cloudification in the first part (Chapter 2, Chapter 3, and Chapter 4) and RAN slicing in the second part (Chapter 5, Chapter 6, and Chapter 7).

In the first part, our focus is on the C-RAN, in which the centralized RAN processing in a cloud can support joint multi-cell processing and boost multiplexing gains. In this sense, the legacy monolithic BSs are replaced with (1) the distributed passive or active radio elements, and (2) the centralized pools of baseband processing units where baseband and protocol processing takes place. Despite its appealing, the C-RAN concept still face challenges in terms of the severe capacity and latency requirements of the FH interface that connects distributed RRU toward the centralized BBU.

- The work in Chapter 2 investigates the impacts among several factors, such as functional split, packet processing and packet scheduling, on the Ethernet-based solution for FH transportation. Specifically, the proposed packetization algorithm and packet scheduling/discard scheme can significantly boost the multiplexing benefits.

- In Chapter 3, two implemented functional splits with the A-law compression scheme over the OAI platform are presented and compared in terms of several KPIs. Further, the improvements of legacy A-law compression scheme are proposed especially for high-order modulations.

- We also present a model and an analytical framework in Chapter 4 for the E2E RAN service delivery to maximize the network spectral efficiency via interplaying different design factors, i.e., user association, RRU clustering, functional split, BBU anchoring, and FH routing.

In the second part, we focus on the RAN slicing to not only provide different levels of isolation and sharing to each slice but also enable the customization across CP, UP and CL.

- We propose a flexible execution environment at the RAN domain as the RAN runtime slicing system in Chapter 5 to host slice service instances over the underlying RAN modules. Moreover, we present a new set of radio resource abstractions in Chapter 5 to customizedly provide physical or virtualized radio resource and to exploit potential multiplexing benefits.
- Additionally, a number of new interfaces are identified in Chapter 6 for the communications between the RAN runtime and the orchestration system. To support slice-based multi-service chain creation and chain placement for disaggregated UDN, we propose a two-stage placement algorithm with an auto-scaling mechanism to increase the performance.
- Finally, the runtime SDK is introduced in Chapter 7 on top of the RAN runtime to simplify the design and development of control applications. These control applications can compose sophisticated and customized control logics by the two-level abstraction approach to be chained across different domains.

8.2 Toward a 5G future

Toward the future 5G, several emerging technology enablers, such as virtualization, softwarization and cloud computing, are highlighted to bring the network flexibility and service deliverability into the picture. Such big leap is far beyond the legacy 3G/4G communication system and it brings the opportunities for both academy and industry to study the whole 5G system in various aspects. Via synthesizing different aspects, 5G can bring its versatility through a number of novel services. In line to provide such versatility, this thesis brings the specific attentions to the RAN domain and explores both C-RAN and RAN slicing topics. To further extend our work done in this thesis, we provide a number of discussion sections at the end of each chapter to individually highlight how they can be technically expanded in the respective focus. More extensively, our work can be served as the foundation for a number of novel applications in the 5G system.

Firstly, our works can fabric the deployment synergy among centralized RAN units (CU and DU), control and user planes CN functionalities, MEC entity, and management and orchestration functions in an edge cloud environment. A deployment example is depicted in Figure 7.1 with the RU prototype that can do the low-PHY processing. Via leveraging the edge cloud deployment together with network functions across different domains, the E2E network slicing can be enabled flexibly between the required level of isolation and sharing. For instance, a slices can utilize the dedicated functions ranging from CU/DU, CN, and management and orchestration function but with shared RUs to guarantee its isolation characteristic.

Secondly, the concept of *as-a-service* for mobile network can be extended toward an *everything-as-a-service* (XaaS) vision, in which everything in the 5G system can be treated as the service following, for instance the pay-as-you-go or zero-rating models. In this sense, this thesis can

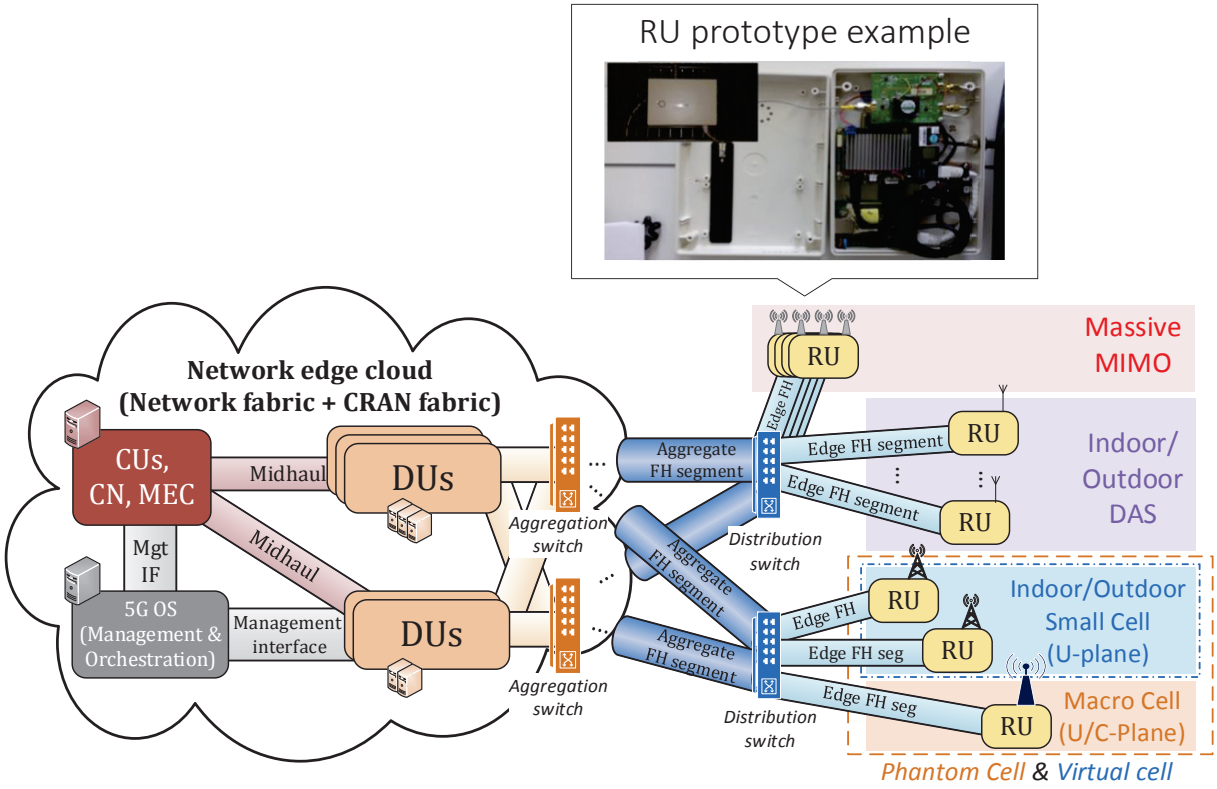


Figure 8.1: An example of 5G edge cloud deployments

shed some lights on a viable RAN-as-a-service model with the associated SLA by considering different levels of isolation and sharing in either monolithic or disaggregated deployment. Moreover, the notion of control applications developed in Chapter 7 can be self-contained and open itself as the shared control applications, e.g., one can provide its spectrum management rule as the inputs to the shared SMA as the SMA-as-a-service.

Last but not least, our work can be seen as the building blocks to facilitate the network intelligence over the RAN domain for enhancing network experience. In this sense, an network intelligence engine [247] can lay in between the high-level applications and the underlying RAN in order to (1) translate the intent from high-level application into network actions/configurations for underlying RAN, and to (2) interpret network states/events into the understandable topology/measurements, via applying big data analysis technologies. Take the intelligent FH management and orchestration in C-RAN as an example, the aforementioned network intelligence engine needs to provide a real-time optimization framework in correspondence to spatio-temporal traffic dynamics, via enabling flexible and dynamic resource slicing and functional split.

Appendix A

Résumé en français

A.1 Evolution vers les réseaux mobiles 5G

Au cours des dernières décennies, les réseaux sans fils 2G, 3G et 4G ont connu une progression du réseau sans fil mobile. Néanmoins, la croissance continue des statistiques de réseau exige un technologique en constante évolution. Par exemple, selon les prévisions de Cisco dans [2], le trafic global de données mobiles atteindra 49 exaoctets par mois d'ici 2021, soit sept fois plus qu'en 2016 en raison de la montée en puissance des smartphones et de la croissance constante de la connexion machines-à-machines (M2M). Par conséquent, une question naturelle surgit dans nos esprits : que sera la 5G? Pour répondre à cette question, les capacités 5G requises fournies par l'UIT-R sont représentées dans la figure A.1 ainsi que la vision IMT-2020 [1], en demandant des améliorations significatives dans le système 3G (IMT-2000) et 4G (IMT-Advanced). Cette large variété de capacités est fortement couplée aux scénarios d'utilisation et aux applications de la 5G. En pratique, il existe trois scénarios d'utilisation clés connues à ce jour [6, 7]: (a) réseau mobile à haut débit amélioré (en anglais enhanced mobile broadband, eMBB), (b) une communication ultra fiable et en faible latence (en anglais ultra-reliable low-latency communication, uRLLC) et (c) communication massive de machines-à-machines (en anglais massive machine type communication, mMTC).

Tout d'abord, le scénario eMBB exige une plus grande importance parmi les huit capacités mises en évidence comme montrer dans la figure A.1. Cependant, toutes les métriques n'ont pas la même importance simultanément dans tous les cas d'usage. Par exemple, un taux de transfert de données utilisateurs plus élevé avec une mobilité du type piétons représente les zones des hot-spots, comparé à une zone de couverture plus large telle que les régions rurales. Comme pour le scénario uRLLC, la latence est de la plus haute importance pour permettre aux applications (quasi-) temps-réels, telles que les systèmes de transport intelligents, les réseaux industriels intelligents, et réseau tactiles commande à distance. De même, une telle capacité est requise même dans un cas de forte mobilité, par exemple jusqu'à 500 km/h, tandis que les débits de données sont comparativement moins importants. Enfin, le scénario mMTC prévoit une densité de connexion massive pour prendre en charge un nombre considérable de capteurs susceptibles de transmettre occasionnellement un débit relativement faible et élastique en latence avec ou sans mobilité. En même temps, l'efficacité énergétique du réseau est essentielle pour le scénario mMTC afin de permettre les déploiements à grande échelle de capteurs très peu coûteux avec une longue durée de vie opérationnelle.

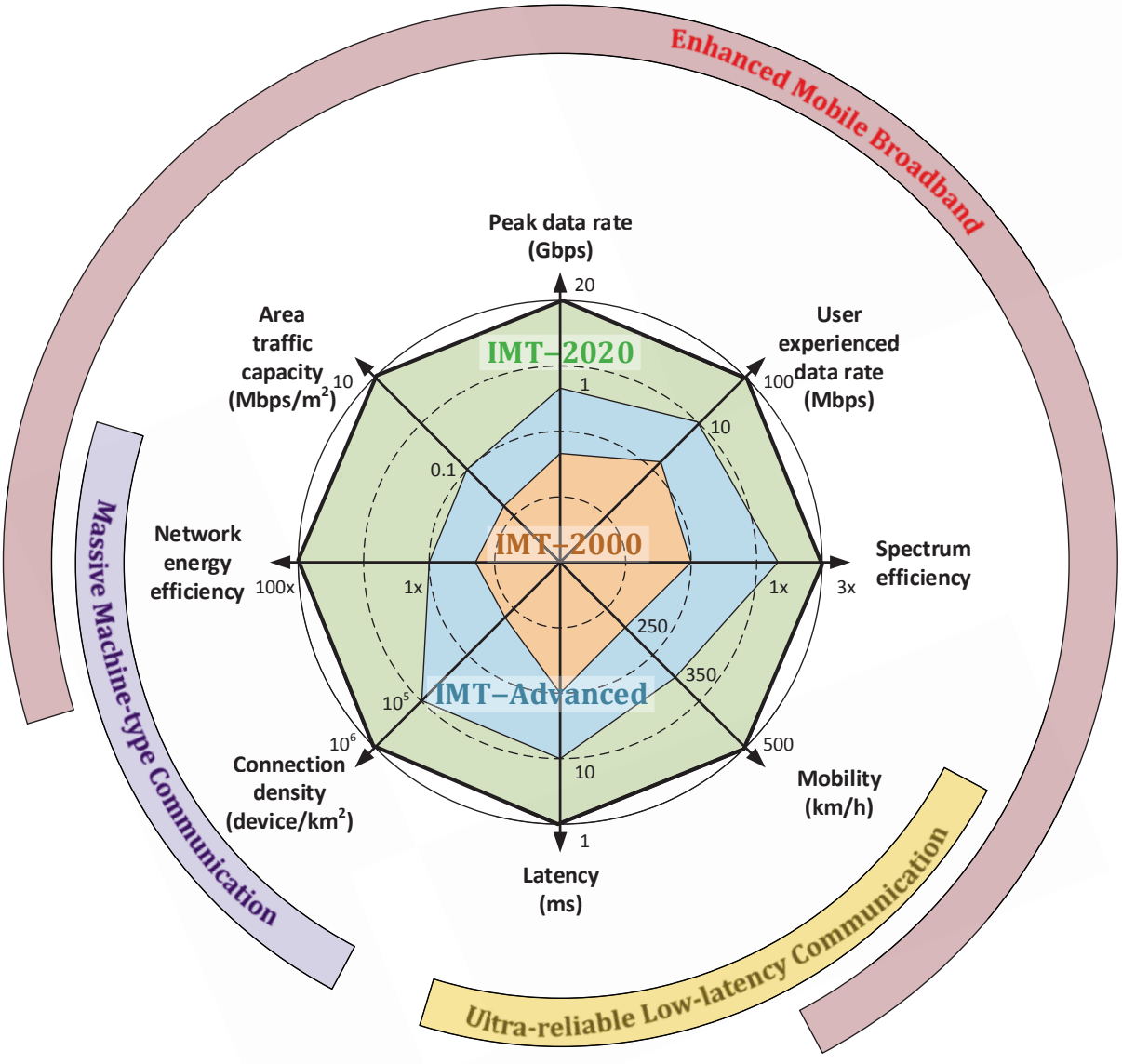


Figure A.1: Vision IMT sur 5G

Pour rendre possible cette vision, l'architecture 5G doit être conçue avec un certain niveau de flexibilité via l'intégration des principes de softwarization de la virtualisation. Des exemples de ces deux principes sont largement connus sous le nom de techniques SDN [19] et NFV [20]. Par conséquent, nous pouvons observer que la 5G fournira un changement de paradigme au-delà de la technologie d'accès radio afin d'établir un système de communication agile et sophistiquée. Par exemple, la synergie des technologies cloud, SDN, et NFV permutent de mieux gérer des spectres disponibles (Ondes millimétriques et sub-6 GHz, par exemple) et des technologies d'accès hétérogène (Wi-Fi, 4G et 5G, par exemple) [26, 27]. Par conséquent, plusieurs services réseau peuvent être crée au-dessus d'infrastructure 5G en composant chaque réseau logique de bout en bout (BEB) personnalisé de manière souple et flexible, y compris les points d'entrée et sortie,

l'infrastructures, les fonctions physiques et virtuelles (en anglais physical and virtual network function, PNF/VNF) à travers plusieurs domaines technologiques, et les fonctionnalités de gestion et d'orchestration de réseau physique et logique. En résumé, l'importance primordiale du système 5G est de fournir un cadre BEB adapté aux exigences d'une multitude de services réseau innovants, tels que les exigences de qualité de service (QoS).

A.2 Motivation et contribution de la thèse

Par rapport à l'évolution vers le réseau mobile 5G, les réseaux logiques multiservices peuvent être établis via le réseau virtuel au-dessus des infrastructures physiques sous-jacentes, comme illustré dans la figure A.2, en comparant à la solution monolithique fournie par la 4G. En outre, le réseau peut être utilisé de manière efficace et indépendante via la création de plusieurs espaces séparées logiquement, un par service, appelés tranches de réseau (en anglais network slice), de manière BEB en incluant des points entrés et sortis, des infrastructures réseau et clouds associés. De plus, chaque réseau logique peut utiliser la technique de virtualisation pour déployer ses fonctions de réseau dans un environnement de nuage souple, soit aux centres de données situés dans des régions périphériques ou locales à la proximité de service de la figure A.2, tout en satisfaisant aux exigences de service spécifiques. Les techniques de virtualisation et de découpage des fonctions présentent toutes deux des difficultés dans le domaine des réseaux d'accès radio (RAN) en raison des caractéristiques suivantes: (i) le domaine RAN fournit les opérations les plus critiques en termes de latence qui auront un impact direct sur les exigences temps réel des VNF et/ou PNF, (ii) la plupart des opérations de calcul intensif existent dans le domaine RAN par rapport à d'autres domaines (par exemple, le cœur de réseau mobile), et (iii) une amélioration substantielle des performances peut être obtenue en appliquant le traitement coordonné et/ou centralisé au RAN (par exemple, traitement multipoint coordonné). À cette fin, l'objectif de cette thèse est d'étudier ces deux techniques et le manuscrit actuel est divisé en deux parties: (i) Cloud-RAN (Chapitre 2, Chapitre 3 et Chapitre 4), et (ii) découpage fonctionnel de RAN (Chapitre 5, Chapitre 6 et Chapitre 7).

Dans la première partie, nous étudions la cloudification du RAN, à savoir le cloud-RAN (C-RAN), dans lequel les traitements de RAN sont centralisé dans un cloud pour une gestion efficace des ressources radio et un traitement multi-cellules commun. Par conséquent, les stations de base monolithiques traditionnelles (BS) sont remplacées par (1) les éléments radio passifs ou actifs distribués et (2) les pools centralisés pour des unités de traitement en bande de base (en anglais base band unit, BBU) où le traitement de signal ainsi que celui de protocole sont effectué. En dépit de ses avantages, le concept C-RAN est toujours confronté à des exigences sévères en matière de capacité et de latence de l'interface fronthaul (FH) qui connecte l'unité de radio distante distribuée à l'unité de traitement en bande de base centralisée. Les travaux du chapitre 2 ont pour objectif de etudier l'impact de plusieurs facteurs, tels que le split fonctionnel et le traitement des paquets, en considérant la solution radio sur Ethernet (RoE) pour le transport FH. De plus, deux divisions fonctionnelles implémentées sur la plate-forme OpenAirInterface (OAI) [164] sont présentées et comparées au chapitre 3 pour justifier l'applicabilité du C-RAN dans un environnement réel. Nous présentons également un modèle et un cadre analytique au chapitre 4 pour la prestation de services BEB RAN afin d'interagir avec différents facteurs pour maximiser l'efficacité spectrale du réseau.

Dans la deuxième partie, nous nous concentrons sur le découpage RAN non seulement pour

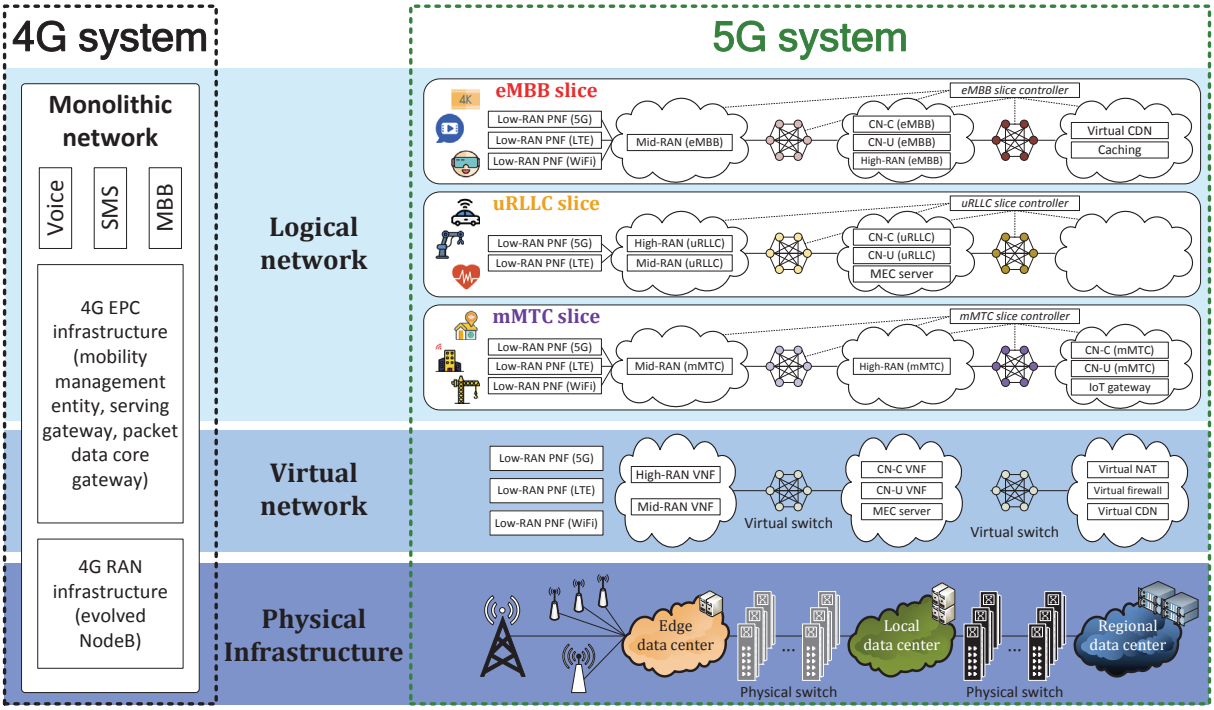


Figure A.2: Correspondance entre une infrastructure physique et un réseau virtuel et des réseaux logiques

permettre des différents niveaux d'isolation et de partage à chaque tranche de réseau (en anglais, network slice), mais également pour customiser le plan de contrôle (en anglais control plane, CP), le plan utilisateur (en anglais user plane, UP) et la logique de contrôle (en anglais control logique, CL). Par conséquent, nous proposons un environnement d'exécution flexible pour le système de slicing de «RAN Runtime »dans le chapitre 5 pour héberger les instances de service sur chacun des modules RAN sous-jacents. De plus, nous présentons au chapitre 5 un nouvel ensemble d'abstractions des ressources radio pour un partage efficace des ressources radio en exploitant les avantages du multiplexage. En outre, un certain nombre de nouvelles interfaces sont identifiées dans le chapitre 6 pour les communications entre le RAN Runtime et le système d'orchestration afin d'augmenter le taux d'acceptation du déploiement de services via le schéma de mise à' l'échelle automatique (en anglais auto-scaling). Enfin, le kit de développement de logiciels (en anglais, software development kit, SDK) est présenté au chapitre 7 en plus du RAN Runtime afin de simplifier la conception et le développement des applications de contrôle. Ces applications de contrôle peuvent composer des logiques de contrôle sophistiquées grâce à l'approche d'abstraction à deux niveaux qui doit être chaînée sur différents domaines.

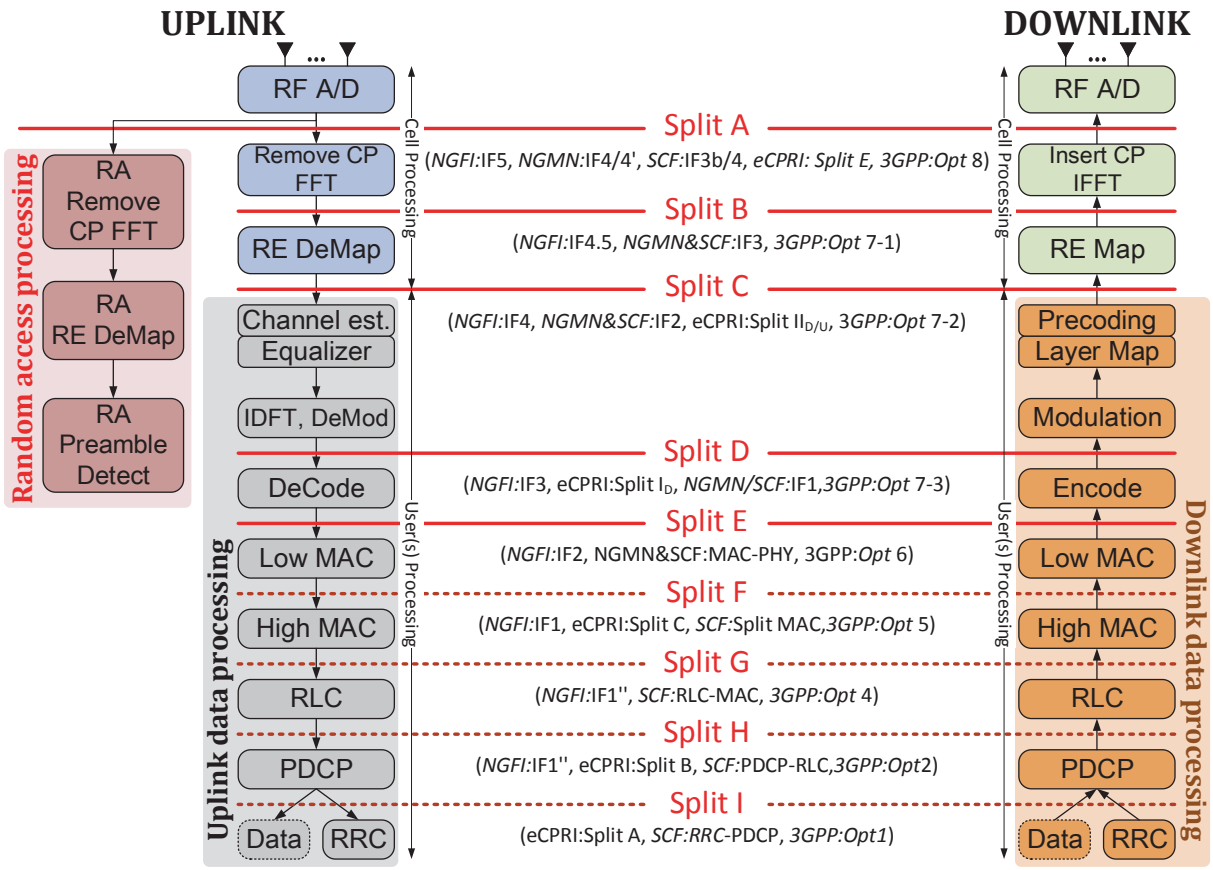


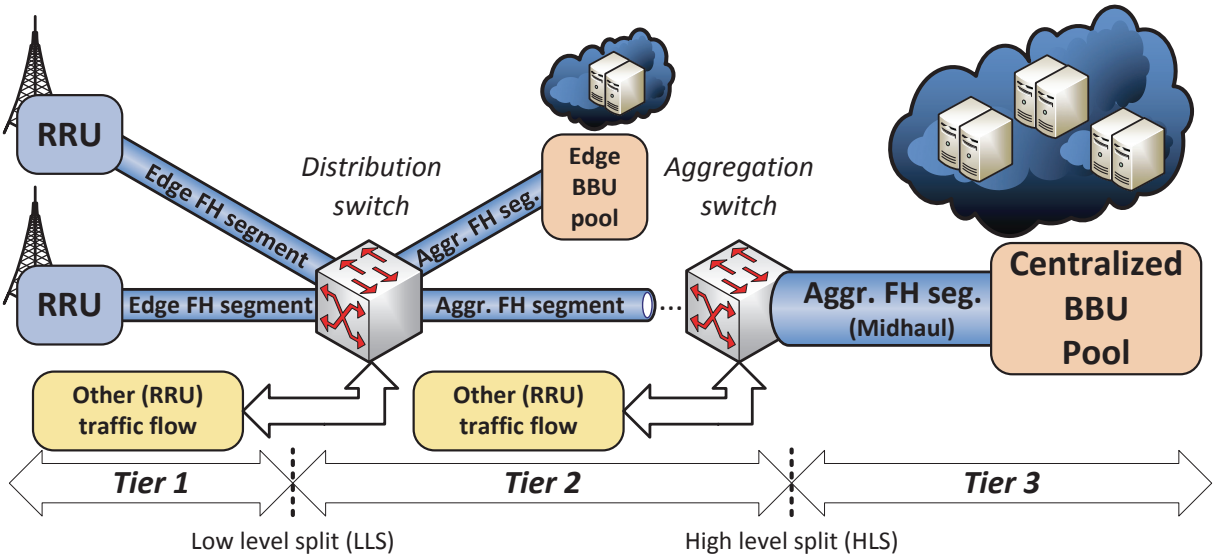
Figure A.3: Plusieurs split fonctionnels des liaisons montante et descendante

A.3 Chapitre 2 — "Impact de split fonctionnel, Packetization, et Ordonnancement de paquet sur la performance de C-RAN"

A.3.1 Motivation

Un obstacle majeur pour le C-RAN est la capacité excessive requit sur des liens FH qui fournit les interconnexions entre le BBU et RRU. Pour réduire l'exigence de FH, le concept de C-RAN a été revu et une distribution plus flexible des fonctionnalités de bande de base et de protocole entre le RRU et le BBU est envisagée [31]. Par conséquent, plusieurs splits fonctionnels sont définis pour répartir le traitement de BS entre les RRU et BBU, à savoir le split A à split I comme démontrer dans la figure A.3, qui également révèle le mappage aux interfaces identifiées par NGFI (en anglais next generation fronthaul interface) [172, 174], SCF (en anglais small-cell forum) [30], et NGMN (en anglais next generation mobile network alliance) [175], et CPRI (en anglais common public radio interface) [167], et 3GPP (en anglais third generation partnership project) [32].

Une autre question clé est de savoir comment les informations entre RRU et BBU sont transportées sur la liaison FH. À la lumiére du concept de split fonctionnel, des types d'informations



Functionalities		Tier 1	Tier 2	Tier 3
3GPP	TS 38.401	gNB-DU		gNB-CU
	TR 36.756	eNB-DU		eNB-CU
	TR 38.816	lls-DU	lls-CU	
NGMN alliance		RU	DU	CU
CPRI forum	CPRI	Radio equipment (RE)	Readio equipment control (REC)	
	eCPRI	eCPRI RE (eRE)	eCPRI REC (eREC)	
NGFI		RRU	Radio aggretaion unit (RAU)	Radio cloud centre (RCC)
IEEE 1914		RU	DU	CU
xRAN forum		RU	lls-CU	
TIP		RRU	vBBU	

Figure A.4: Les topologies de réseau C-RAN considéré

différents sont transportés sur la liaison FH. Étant donné l'adoption généralisée d'Ethernet dans les infrastructures cloud et les réseaux de transport, l'approche RoE [171] est une alternative générique, économique et prête à l'emploi pour le transport de trafic sur la liaison FH. De plus, il est prévu que le réseau FH évolue vers une topologie de réseau maillé multi-sauts plus complexes, et nous nous concentrons donc sur une topologie de réseau FH multi-segment comme illustré dans la figure A.4. Dans ce cas, le commutateur de distribution et/ou agrégation peuvent être utilisé pour transporter non seulement le trafic C-RAN, mais également d'autres trafics, ce qui est conforme au concept selon lequel le réseau C-RAN pourrait réutiliser des réseaux Ethernet déjà déployés. En outre, la topologie C-RAN considérée peut-être associée à l'architecture à trois niveaux proposée par l'alliance 3GPP [177–179], NGMN, Forum CPRI, NGFI, IEEE 1914, forum xRAN [173], projet de télécommunications (TIA) [180], et SCF, dans lesquels il existe deux split fonctionnels entre la division de bas niveau et la division de haut niveau. À cette fin, nous proposons l'utilisation d'un réseau FH base sur le paquet et considérons un réseau FH à deux segments dans les analyses suivantes.

Table A.1: Nombre de RRU supporté en tenant compte la packetization et le split fonctionnelle

Split fonctionnelles		**Nombre de RRU supporté**				
		Taux de pointe	**LPB**	**CLB**	**ACB**	**ASB**
Split A ($P_{opt} = 5011$)		5	5			
Split B ($P_{opt} = 2400$)		9	9			
Split C ($P_{opt} = 4363$)		9	10	10	10	11
Split D	($P_{opt} = 3741$, LAD=10, no pre-fetch)	7	12	11	12	13
	($P_{opt} = 8960$, LAD=10, pre-fetch)		18	17	18	19
	($P_{opt} = 8960$, LAD=3, pre-fetch)		29	29	26	29
Split E ($P_{opt} = 8960$)		66	153	154	141	142

A.3.2 Les résultats obtenues

Pour représenter l'impact conjoint de la mise en paquets et du split fonctionnel, le tableau A.1 présente les avantages du multiplexage en termes de nombre de RRU pouvant être pris en charge sur un réseau FH Ethernet à deux segments pour une capacité FH de 4 et 20 Gbit/s sur chacun des segments FH, respective. Tout d'abord, l'analyse du taux de pointe est effectuée dans la direction de la liaison montante en tant que base de comparaison. Ensuite, nous considérons les impacts de la mise en paquets en termes de plusieurs paramétres: (1) taille optimale de la charge utile (P_{opt}), en octets qui minimise le délais sur le FH, (2) la profondeur anticipée (en anglais look-ahead depth, LAD) définit par la durée non-causal des symboles future pour le traitement actuel, et (3) le schéma de pré-extraction (en anglais pre-fetch) qui a pour but de traiter et de mettre en paquets immédiatement les échantillons radio, le cas échéant.

Le premier paramétre est utile pour les splits de bas niveau (par exemple, Split C), car leur débit de trafic FH est plus important. Alors que les deux derniers paramétres sont bénéfiques pour les splits de niveau plus élevé (par exemple, Split D) en raison de trafic bursty sur la liaison FH. Ces résultats sont calculés en fonction du 95e percentile du délai FH sous quatre limites de multiplexage UE, à savoir, limite d'approvisionnement inférieure (en anglais lower provisioning Bound , LPB), limite inférieure conservatrice (en anglais conservative lower bound , CLB), limite commune agrégée (en anglais aggregated common bound , ACB) et limite stricte agrégée (en anglais aggregated strict bound , ASB). Les premiers résultats ne prennent en compte que l'ordonnément des paquets du premier arrivé-premier servi (en anglais first-come-first-serve, FCFS) au niveau du commutateur agrégé. Par conséquent, nous étudions d'autres schémas de l'ordonnément de paquets, tels que le temps de traitement le plus court (en anglais shortest processing time, SPT), le moindre bit restant (en anglais least remaining bit, LRB), la premiére date d'échéance (en anglais earliest due date, EDD) et le moindre temps de relâchement (en anglais least slack time, LST). Les nombres maximums de RRU supportés pour le split E sont indiqués dans le tableau A.2, et la politique LRB montre la plus grande amélioration puisqu'elle proratise les RRU avec le moins de bits restants, c'est-à-dire les RRU faiblement chargées. De tels avantages se font au détriment des inéluctabilitès parmi les RRU, et nous pouvons observer sur la figure A.5 que le paquet rejeté pour le LRB est principalement constitué de gros paquets provenant de RRU fortement chargées.

Les travaux liés à ce chapitre sont les suivants:

- C.-Y. Chang, R. Schiavi, N. Nikaein, T. Spyropoulos, and C. Bonnet, "Impact of pack-

Table A.2: Nombre de RRU supporté en tenant compte l'ordonnancement des paquets

Split fonctionnelles	Nombre de RRU supporté				
	FCFS	**SPT**	**LRB**	**EDD**	**LST**
Split E	153	156	162	153	140

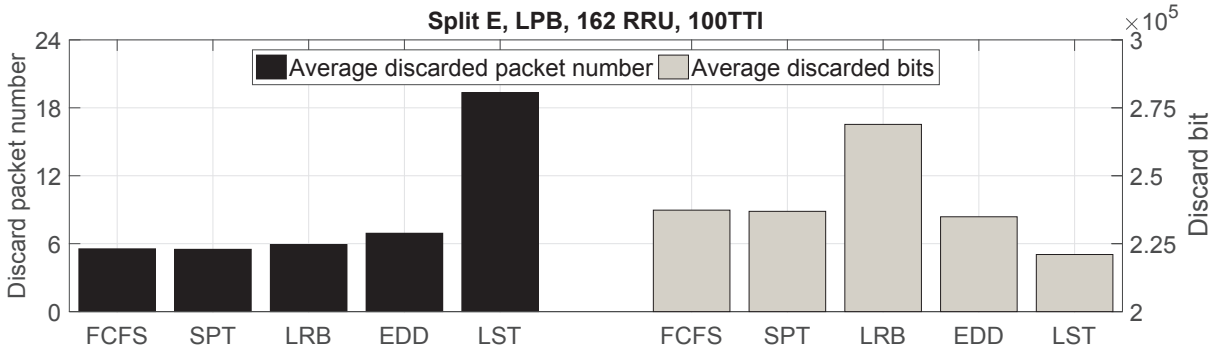


Figure A.5: Statistiques des paquets rejetés en tenant compte de l'ordonnément des paquets

etization and functional split on C-RAN fronthaul performance," in *Proceedings of 2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1-7.

- C.-Y. Chang, N. Nikaein and T. Spyropoulos, "Impact of packetization and scheduling on C-RAN fronthaul performance," in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016, pp. 1-7.

A.4 Chapitre 3 — "Split fonctionnelle flexible sur la liaison Fronthaul base sur Ethernet dans C-RAN"

A.4.1 Motivation

Selon les splits fonctionnelles identifiées dans la figure A.3, le compromis est constaté entre le niveau de centralisation et les exigences de la FH. Par exemple, le Split A peut bénéficier de tous les avantages de la centralisation au détriment des exigences de capacité FH extrêmement élevées, tandis que les split de niveau supérieur peuvent réduire la capacité FH mais avec limite des possibilités pour le traitement centralisé et conjointe. À cette fin, la demande de split fonctionnelle flexible est identifiée dans la figure A.6, dans laquelle le RAN évolue de manière distribuée (c'est-à-dire D-RAN) vers un déploiement centralisé prenant en charge la séparation fonctionnelle flexible.

A.4.2 La solution proposé et les résultats obtenues

Dans ce chapitre, nous présentons d'abord une solution architectural RRU/BBU unifié dans la figure A.7 avec sept composants principaux. Parmi ceux-ci, trois composants principaux sont mis en évidence: (1) l'unité de fonction de l'interface split effectue le traitement spécifique adapté

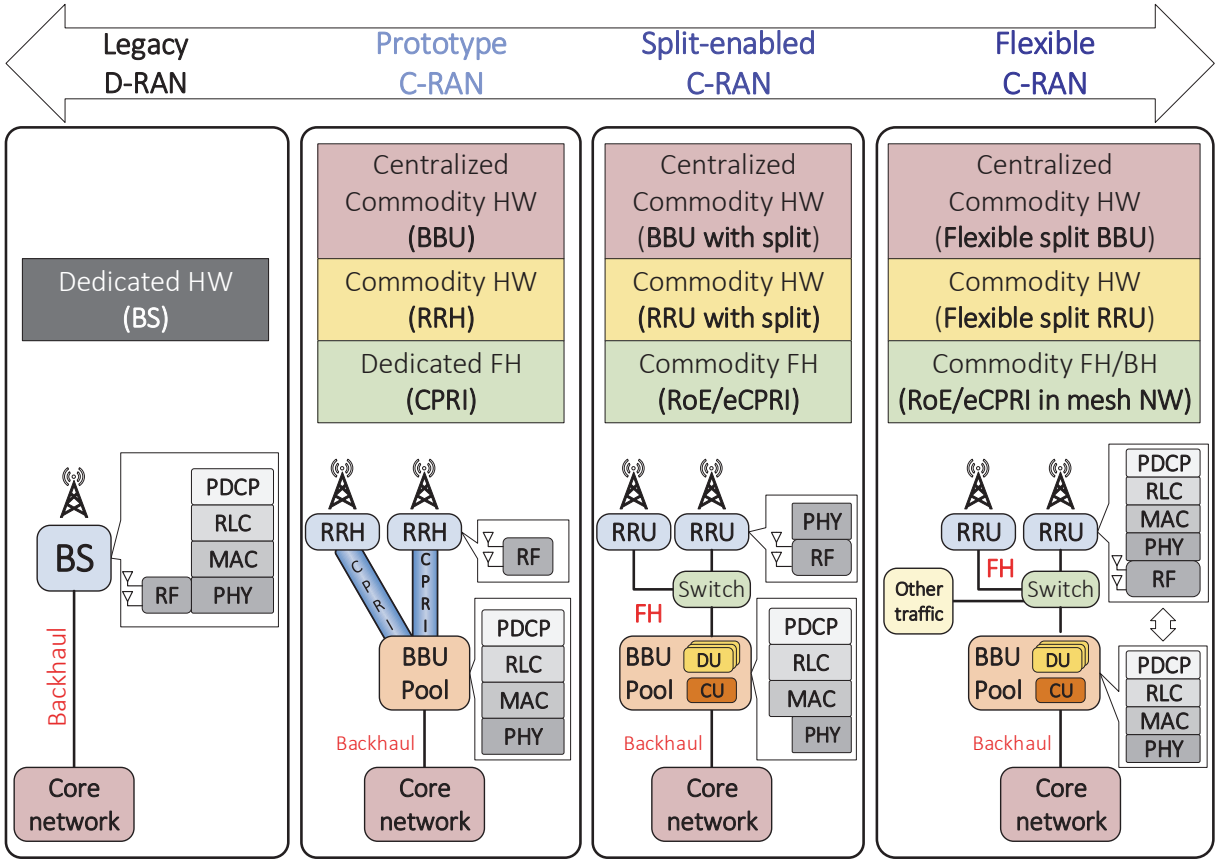


Figure A.6: Évolution du RAN du D-RAN au C-RAN flexible

aux splits fonctionnelles appliquées, (2) unité de (dé-)compression appliquant le schéma de (dé-)compression aux données pour réduire le débit FH, et (3) unité de configuration de transport qui applique le traitement de mise en paquets et ajuste l'horodatage entre RRU et BBU. De plus, le schéma de transport FH basé sur Ethernet est capable de transporter à la fois les flux logiques de contrôle et de données avec les en-têtes séparés en fonction du protocole UDP (User Datagram Protocol) ou Ethernet RAW. Le mode RAW utilise l'Ethernet brut pour réduire le délai de traitement.

Nous évaluons l'implémentation C-RAN en utilisant la plate-forme OAI pour les splits A et B sous deux déploiements C-RAN: 1 saut et 2 sauts entre RRU et BBU. Premièrement, le débit de liaison FH ainsi que le débit théorique pour les cas de largeur de bande radio de 5 MHz et 10 MHz sont indiqués dans la figure A.8. Nous pouvons voir que le mode RAW a peu de surcharge, alors que le mode UDP affiche plus de temps supplémentaire. De plus, la compression A-Law appliquée peut atteindre une réduction de presque 50% du débit FH en utilisant le format 8 bits pour représenter chaque échantillon de données 16 bits pour les parties réelles et imaginaires. De plus, le plan d'utilisation, UP, de plusieurs scénarios de déploiement C-RAN au niveau de l'application est mesuré à la figure A.9. Nous pouvons observer que ces scénarios de déploiement de C-RAN montrent presque la même variation de qualité que l'ancien D-RAN.

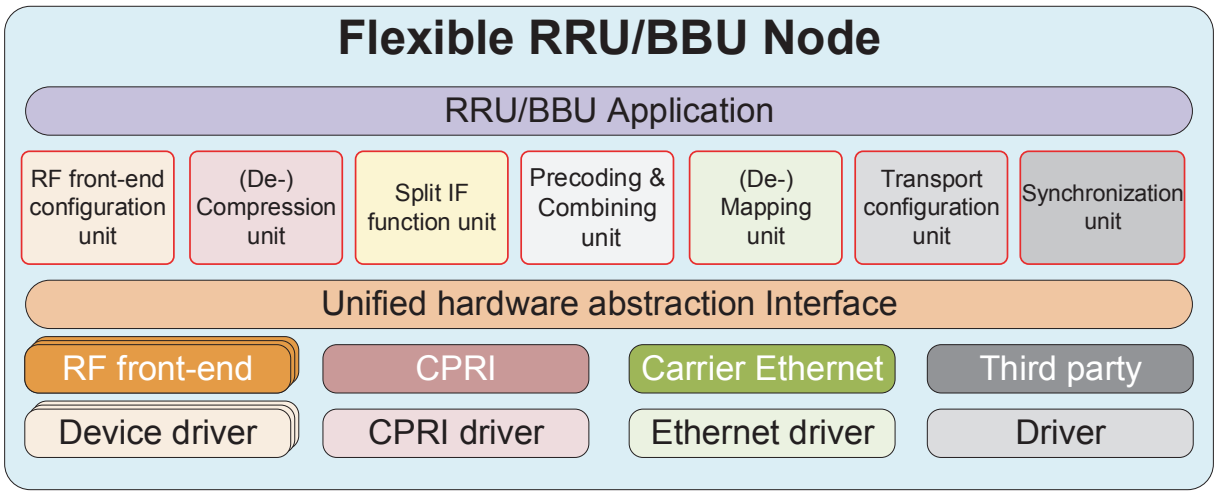
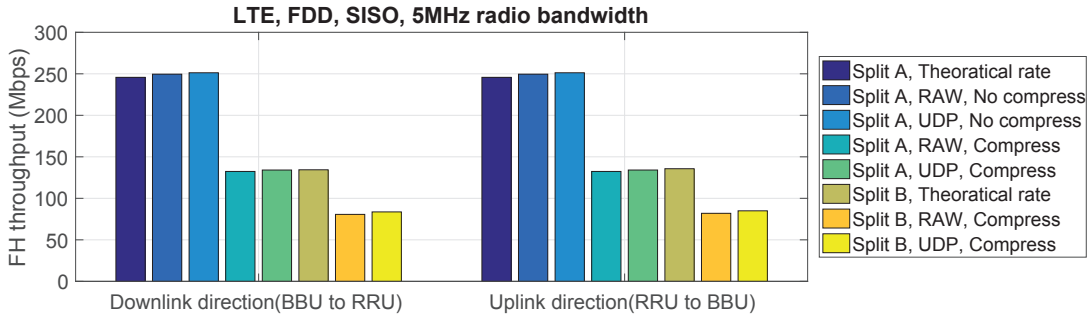
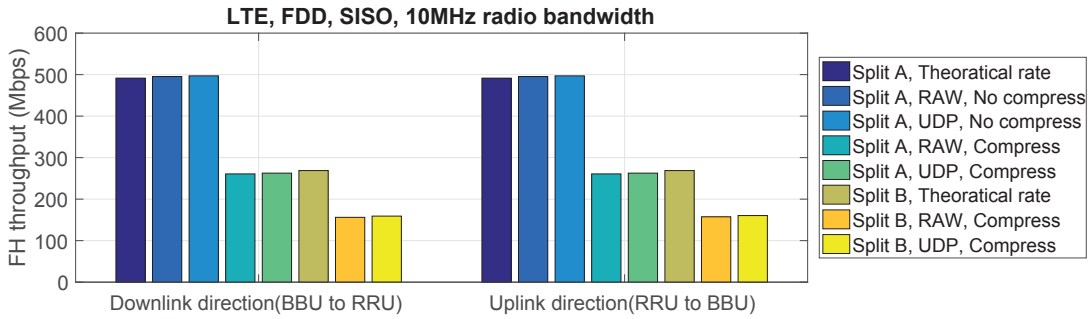


Figure A.7: Solution proposée pour les unités RRU et BBU



(a) Débit FH requise pour des bandes passantes de 5 MHz

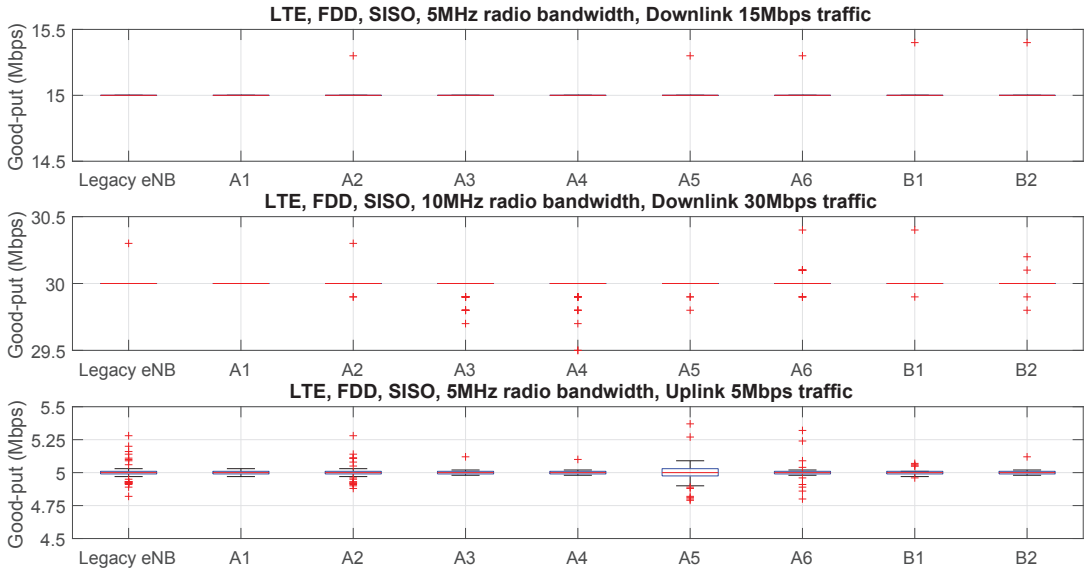


(b) Débit FH requise pour des bandes passantes de 10 MHz.

Figure A.8: Débit FH requise pour des bandes passantes de 5 MHz et 10 MHz

Les travaux liés à ce chapitre sont les suivants:

- C.-Y. Chang, N. Nikaein, R. Knopp, T. Spyropoulos, and S. S. Kumar, "FlexCRAN: A flexible functional split framework over Ethernet fronthaul in Cloud-RAN," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1-7.



Mode	Split	Protocole	Compression	Saut
A1	A	RAW	Non	1
A2	A	RAW	Non	2
A3	A	UDP	Non	2
A4	A	RAW	Oui	2
A5	A	UDP	Oui	2
B1	B	RAW	Oui	2
B2	B	UDP	Oui	2

Figure A.9: Mesures de débit en liaison descendante et montante de plusieurs scénarios de déploiement C-RAN

A.5 Chapitre 4 — "Centralisation flexible du C-RAN pour le service RAN de bout en bout"

A.5.1 Motivation

Le C-RAN flexible peut permettre un schéma de coopération adaptative de regrouper les RRU en plusieurs clusters RRU pour répondre aux cas d'usage divers, tels que le réseau MIMO et la cellule virtuelle. Prenons l'exemple de la figure A.10, chaque grappe RRU peut être formée en utilisant le fractionnement fonctionnel différent et le BBU ancré où le traitement coopératif a eu lieu. De plus, le routage de réseau FH et l'association d'utilisateur doivent être pris en compte pour montrer complétement comment le service RAN est fourni entre deux points d'extrémité, c'est-à-dire le pool BBU et les utilisateurs finaux. En résumé, ces facteurs peuvent être considérés conjointement comme représentant un service RAN BEB.

A.5.2 Les résultats obtenues

Sur la base des facteurs mentionnés ci-dessus, nous pouvons les classer en trois catégories: (i) regroupement des RRU et association des utilisateurs, (ii) split fonctionnelle, et (iii) ancrage des BBU et routage des réseaux FH. Tous ces facteurs sont pertinents et auront un impact mutuel.

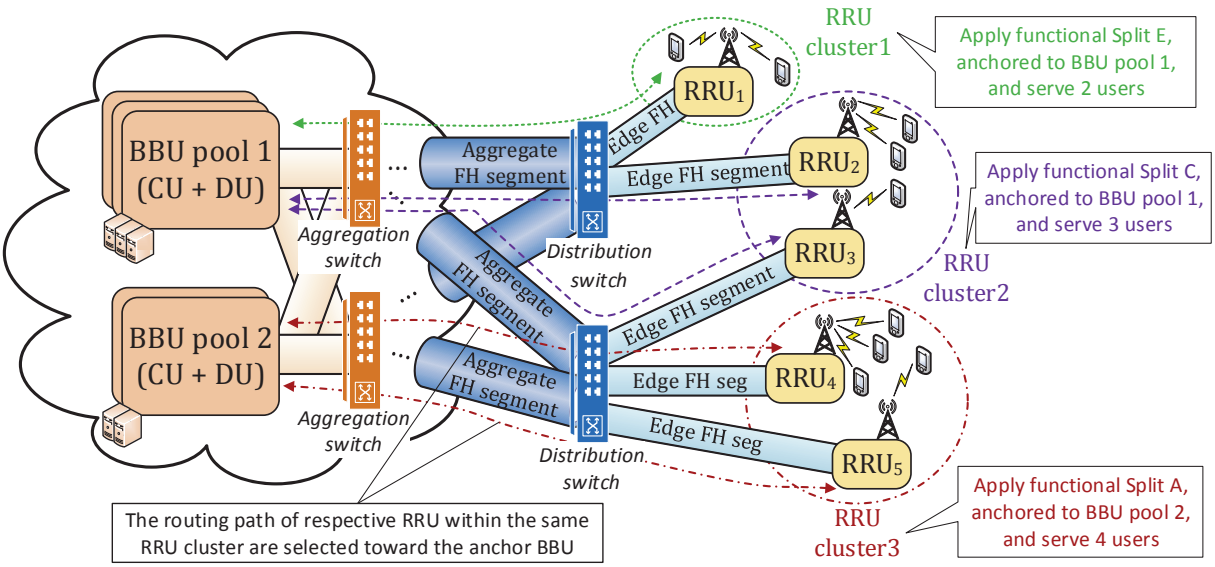


Figure A.10: Topologie C-RAN pour prendre en charge le service RAN BEB

Par exemple, former un plus grand cluster RRU peut aider à mieux coordonner leur traitement, mais aussi introduire un besoin de débit FH plus élevé vers le BBU d'ancrage centralisé. Pour remédier à ce débit FH plus élevé, nous pouvons appliquer la technique de split fonctionnelle pour déplacer certaines fonctions du BBU d'ancrage aux RRU associes en réduisant le gain de la centralisation. En ce sens, nous formulons le problème global d'optimisation afin d'optimiser l'efficacité spectrale du réseau en tenant compte des contraintes entre l'association d'utilisateurs, le regroupement RRU, le split fonctionnelle, l'ancrage des BBU et le routage réseau FH. Pour distinguer quantitativement les avantages de la centralisation, nous fournissons des scémas de CoMP de liaison montante différents pour des splits fonctionnelles différentes et formulons leurs valeurs respectives leur SINR (en anglais signal-to-interference-plus-noise ratio).

Néanmoins, le problème formulé peut être réduit polynomiquement en un problème de Knap-Sac à choix multiples (MMKP) multidimensionnel avec une complexité NP-hard [203]. À cette fin, nous revoyons le problème formulé et proposons une solution pratique pour traiter le problème de manière systématique et hiérarchisée. Enfin, nous fournissons l'algorithme efficace correspondant et montrons les résultats numériques pour justifier son efficacité.

Le travail lié à ce chapitre est le suivant:

- C.-Y. Chang, N. Nikaein and T. Spyropoulos, "Flexible centralization level of Cloud-RAN," to be submitted to ACM MOBIHOC 2019.

A.6 Chapitre 5 — "Système de Slicing RAN Runtime pour une exécution de service flexible dynamique"

A.6.1 Défis de motivation et de conception

Dans la seconde partie de cette thèse, nous nous concentrons sur le RAN Slicing, qui représente une évolution naturelle du concept de RAN partagé(en anglais RAN sharing) introduit depuis

l'ère 3G/4G. En outre, à l'origine de la découpe du RAN, l'idée de slicing vise à considérer collectivement les réseaux logiques associés un réseau physique [248]. Pratiquement, les actions prise par un slice n'affecteront pas négativement les autres slices [109], même si elles partagent la même infrastructure physique. De plus, pour desservir plusieurs services fournis par la 5G, plusieurs organisations soulignent l'importance de la conception orientée services d'une manière BEB, comme ITU-T [110], NGMN [111] et 5GPPP (en anglais 5G public private partnership) [21]. En outre, le 3GPP mentionne les principes de réalisation du découpage en slice dans [32, 113], qui peuvent être activés par le biais du concept RAN logiciel (en anglais software-defined RAN, SD-RAN) qui dissocie le traitement du plan de contrôle du traitement du plan utilisateur. En conclusion, quatre catalyseurs technologiques sont essentiels à la réalisation du concept de tranchage RAN décrit à la figure A.11.

La conception d'un système de découpage RAN révèle plusieurs défis :

- Permettre à chaque slice d'interagir avec le RAN sous-jacent et de modifier le comportement de CP et UP qui sont déterminés dynamiquement pendant son exécution,
- Fournir différents niveaux d'isolation et de partage entre différents slices pour permettre à chaque slice de composer de manière flexible les ressources RAN spécifiques aux service associé et de les traiter à partir des ressources multiplexées ou dédiées, et les fonctions CP/UP respectives,
- Considérer le gain de multiplexage pour les ressources radio sous-jacentes et les modules RAN,
- Fournir les interfaces de programmation d'applicatives (API) pour permettre la prise en compte des décisions CP, UP et de contrôle spécifiques de chaque slice.

A.6.2 Architecture proposée et résultats obtenues

Dans ce chapitre, nous concevons l'architecture de RAN Runtime comme présenté dan la figure A.12 qui peut résoudre les problèmes mentionnés. Le système de slicing de RAN Runtime peut fournir un environnement d'exécution flexible pour exécuter plusieurs instances de RAN virtualisées avec les niveaux d'isolation et de partage pour des ressources ainsi que des modules RAN et ses fonctionnes sous-jacents. Par conséquent, chaque slice en cours d'exécution peut interagir avec les modules RAN pour accéder aux ressources et à l'état, et contrôler le comportement RAN sous-jacent. Plus précisément, cinq services de RAN Runtime sont fournis pour chaque slice: (1) gestionnaire de contexte, (2) gestionnaire de slice, (3) gestionnaire de virtualisation, (4) application de contrôle commun et (5) commutateur des donnés utilisateurs.

De plus, un ensemble d'approche d'abstraction de ressources est fourni dans le tableau A.3, dans lequel les ressources RAN demandées peuvent être abstraites de différentes manières correspondant àla granularité de ressource demandée, par exemple une granularité fixe, contiguë, non contiguë et minimale. En outre, le schéma de virtualisation de ressources est fourni par le biais duquel le fournisseur d'infrastructure peut mapper les ressources allouées par chaque slice d'une manière différente pour augmenter le gain de multiplexage. Un résumé du gain de multiplexage potentiel est présenté à la figure A.13 en termes de nombre de slice supplémentaires que le RAN Runtime pourrait satisfaire (G_s), et le nombre de ressources supplémentaires non utilisées (G_r). Nous pouvons voir que l'algorithme granulaire et greedy proposé peut atteindre la performance proche en tant que solution optimale des cas moyens et des pires.

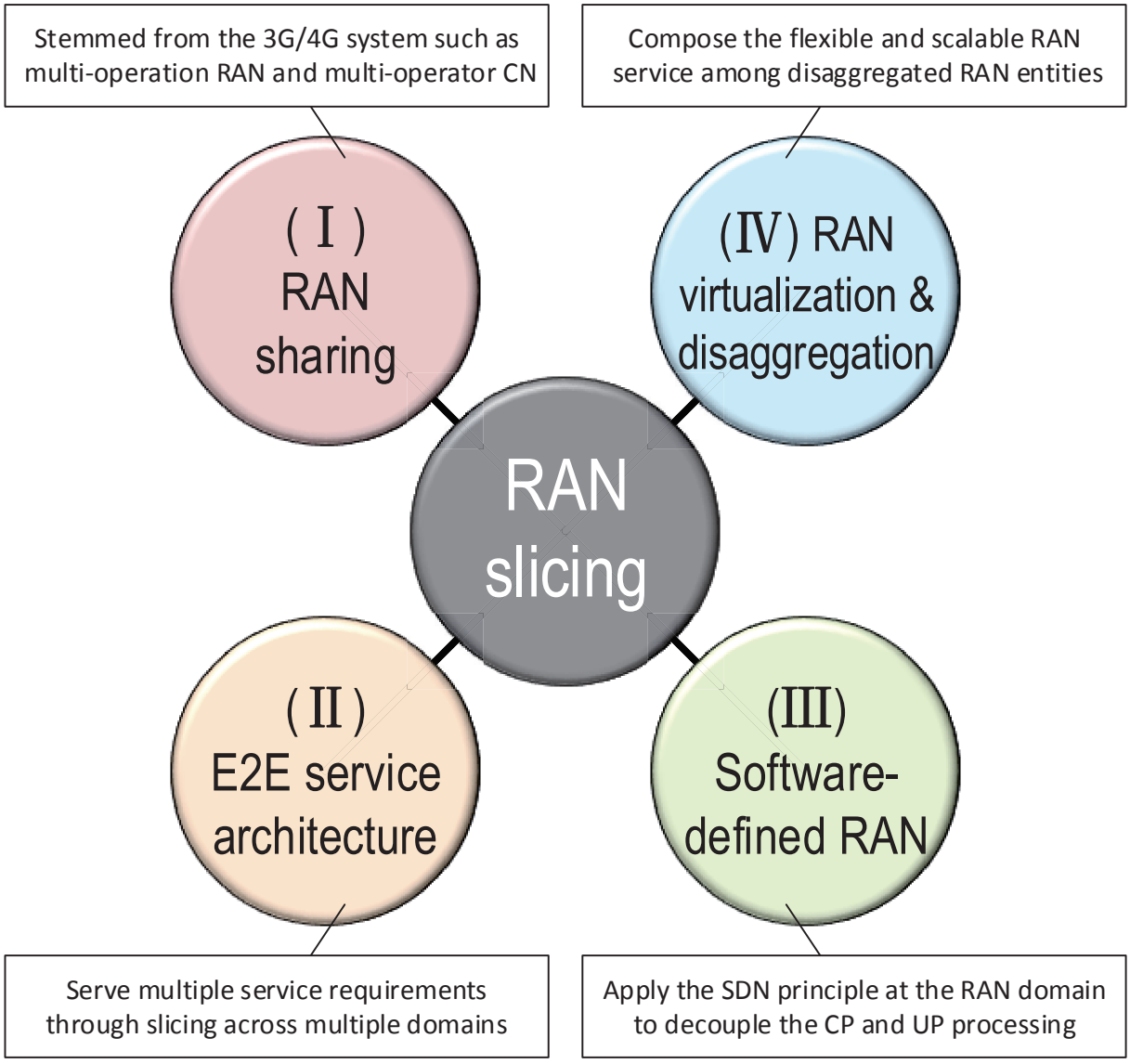


Figure A.11: Facilitateurs technologiques pour le RAN slicing

Pour valider le concept du système de découpage RAN Runtime, nous mettons également en œuvre un prototype 4G basé sur RAN Runtime. Le RAN Runtime est développé sur la base de FlexRAN [154] à partir de la plate-forme OAI et chaque slice instanciée est construite sur le contrôleur FlexRAN avec le CP, UP et CL customisé. Dans ce qui suit, nous montrons l'impact du multiplexage et de la préemption des ressources avec trois slice instanciées dans la igureA.14. Spécifiquement, le slice 1 peut préempter les ressources de toutes les autres tranches lorsque le débit réel (agrégé) dépasse le débit demandé, le slice 2 ne peut qu'accroître son gain de multiplexage en utilisant les ressources non allouées, tandis que le slice 3 ne peut ni préempter ni multiplexer les ressources, mais est soumis 'a la préemption du slice de priorité élevée (c.-à-d. le slice 1). On peut voir que le slice 1 peut adapter de manière agile son débit lorsque sa charge

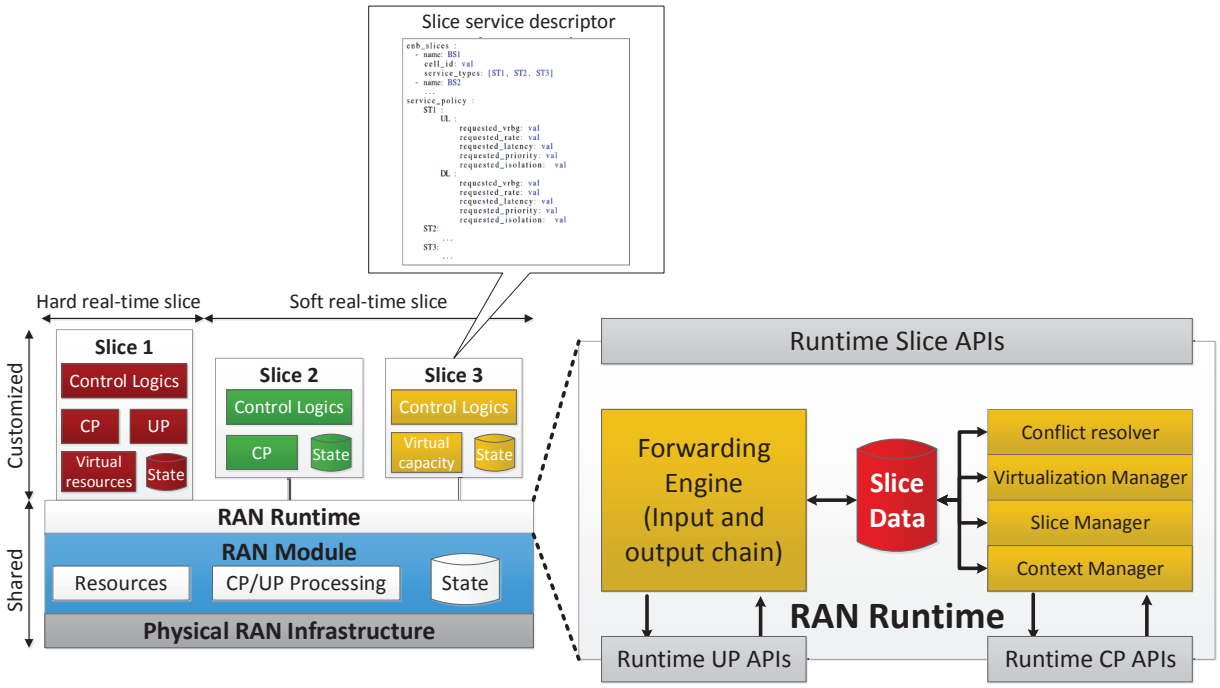


Figure A.12: Architecture de RAN Runtime.

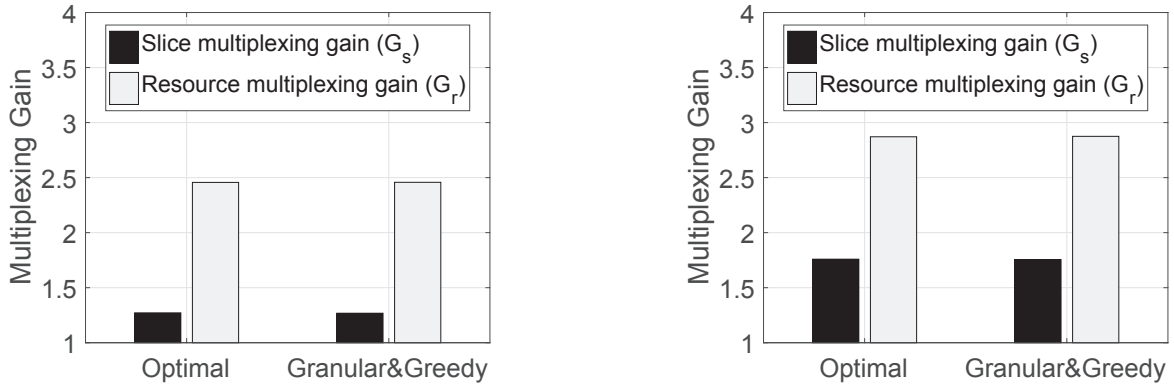
Table A.3: Correspondance entre le type d'abstraction de ressources et le type d'allocation.

Requested resources	Types d'abstraction (Granularité)	Type d'allocation de ressource	
		Liaison descendante	Liaison montante
Bloc de ressources	vRBG Type 0 (Non contiguë)	Type 0, Type 1, Type 2 distribué	Type 1
	vRBG Type 1 (Contiguës)	Type 0, Type 2 localized	Type 0
	vRBG Type 2 (Granularité fixe)	Type 2 localisé	Type 0
Capacité	vTBS Type 0 (Minimale)	Tous les types	Tous les types

utile varie en préemptant les ressources d'autres slices, c'est-à-dire de 3 Mbps à 6 Mbps, tandis que le slice 2 subit une chute de débit de 10 Mbps à 8 Mbps. La même tendance est observée dans la mesure de la gigue de retard, dans laquelle le slice 1 subit la gigue minimale car il a la plus haute priorité et le slice 3 souffre de la plus grande gigue de retard en raison de sa plus faible priorité.

Les travaux liés à ce chapitre sont les suivants:

- C.-Y. Chang and N. Nikaein, "RAN runtime slicing system for flexible and dynamic service execution environment," *IEEE Access*, vol. 6, pp. 34018-34042, 2018.



(a) Cas pire **(b)** Cas moyen

Figure A.13: Gain de multiplexage par slice et par ressource radio.

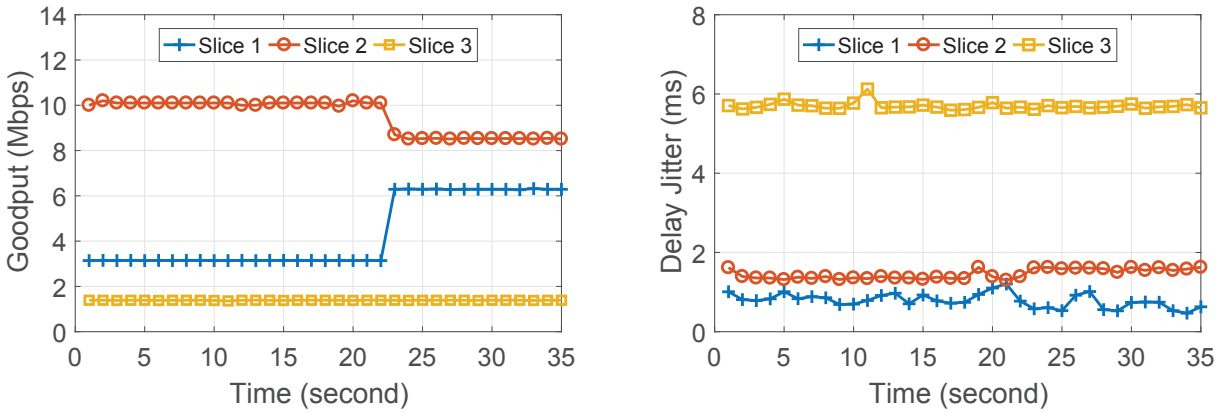


Figure A.14: Impact de la préemption et du multiplexage sur la gigue de débit utile bon et de retard.

- C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Radio access network resource slicing for flexible service execution," in *Proceedings of IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Apr. 2018, pp. 1-6.

A.7 Chapitre 6 — "Orchestration des Slice pour un RAN Multi-Service et Désagrégés"

A.7.1 Motivation

Basé sur le système de slicing RAN Runtime mentionné ci-dessus, le défi à' venir est de savoir comment prendre en charge les procédures d'orchestration des slices adaptées aux besoins du chaque locataire. Ce défi est particulièrement important dans le cas de RAN multiservices désagrégés pour un déploiement de réseau ultra-dense (en anglais ultra-dense networking, UDN),

dans lequel chaque station de base peut être située sur chaque réverbère, à chaque arrêt de bus ou dans des environnements intérieurs. Notez que lors de l'orchestration de plusieurs services sur le RAN, l'utilisation efficace des ressources et la mise à l'échelle des services sont obligatoires, tout en tenant compte des exigences spécifiques en termes de performances et de customisation de service. Dans ce chapitre, nous nous concentrons sur la définition d'un ensemble d'interfaces pour les interactions entre le RAN Runtime et le système d'orchestration pouvant prendre en charge la création de chaînes multiservices et le placement de chaînes.

A.7.2 Architecture proposée et résultats d'exécution

Dans la figure A.15, plusieurs interfaces sont mises en évidence entre le RAN Runtime et les entités d'orchestration définies par 3GPP dans [206], à savoir la fonction de gestion des services de communication (en anglais communication service management function, CSMF), la fonction de gestion des slices de réseau (en anglais network slice management function, NSMF) et celle des sous-réseaux (en anglais network slice subnet management function, NSSMF). Plus, un certain nombre de nouvelles interfaces est identifié: (1) I1 vise à exposer les services RAN Runtime actifs et à récupérer les messages pour le monitoring et le feedback, (2) I2 utilisé pour les événements spécifiques et remplir le SIB (en anglais slice information base) en conséquence, (3) I3 customise la gestion et la surveillance de chaque slice et indique les modifications éventuelles du RAN sous-jacent, (4) I4 enregistre un slice dans le RAN Runtime et utilise le service de RAN Runtime en tant que processus distinct.

En outre, nous présentons une utilisation pratique comme placement multi-services pour les chaînes de services partagées et dédiées dans une architecture RAN à trois niveaux: unité centrale, CU, avec protocole d'adaptation de données de service (en anglais service data adaptation protocol, SDAP) et traitement de protocole de convergence de données par paquets (en anglais packet data convergence protocol, PDCP), DU avec liaison radio contrôle (en anglais radio link control, RLC), contrôle d'accès (en anglais medium access control, MAC), et la couche physique avec les traitements de haut niveau, et RU avec a couche physique avec les traitements de bas niveau.

Ensuite, un algorithme de placement en deux étapes est fourni, dans lequel la fonction partagée est placée en premier, puis les fonctions customisées sont placées en conséquence. Nous pouvons d'abord voir que le ratio d'acceptation illustré par la figure A.16a sous différentes ressources CPU hétérogènes repartis parmi les 16 nœuds. Notez que nous considérons un déploiement ultra-dense (384 ou 480 RU) avec différentes tailles de chaque cluster RU (6 ou 24 RU). Lors du regroupement de 24 RU, les index d'hétérogénéité 4 et 5 fournissent la plus grande amélioration, car les fonctions DU dans les nœud N1 à N6 consomment davantage de ressources et exploitent mieux l'hétérogénéité des ressources. Dans la figure A.16b, nous comparons le nombre de ressources restantes à tous les DU après leur placement (c'est-à-dire les ressources inutiliséées) et le nombre de ressources demandées non satisfaites (c'est-à-dire les ressources des chaînes rejetées). Nous pouvons voir que les CPU restantes sont plus nombreuses que celles demandées dans le cas de 384 RU regroupées en 24, ce qui déclenchera une action de scale-up pour une redistribution des ressources inutilisées vers un sous-ensemble de nœuds pour augmenter le taux d'acceptation. En revanche, une action de scale-out doit être déclenchée pour fournir plus de nœuds dans le cas de 480 RU regroupées en 6 ou 24, les ressources restantes étant inférieures aux ressources demandées.

Le travail lié à ce chapitre est le suivant:

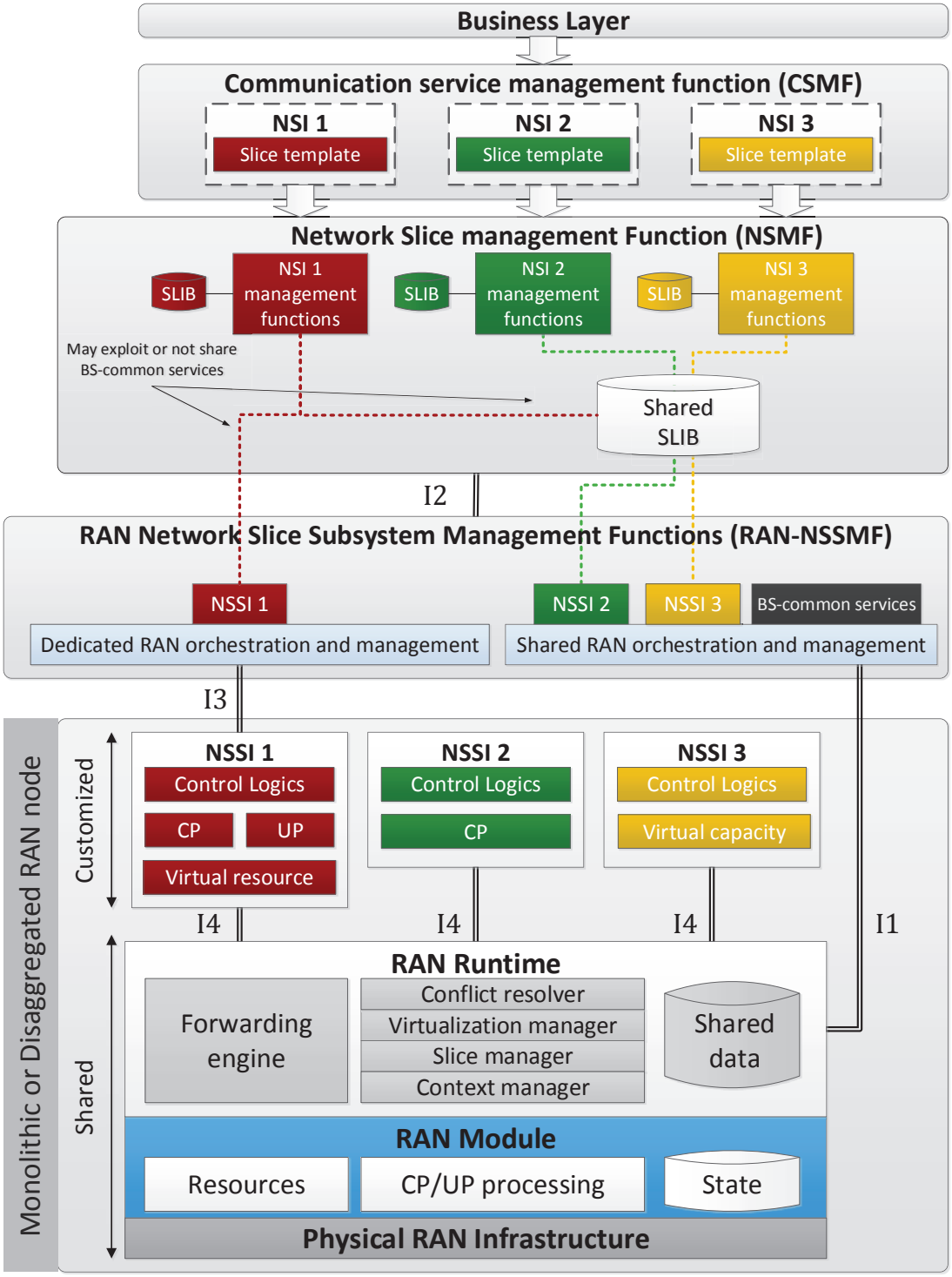
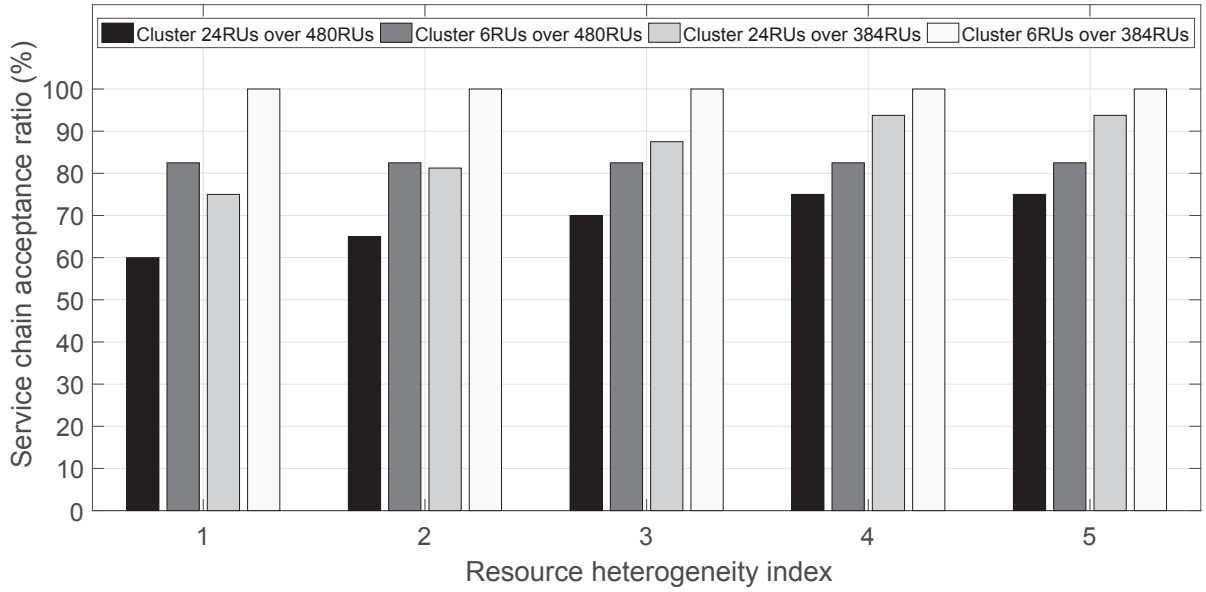
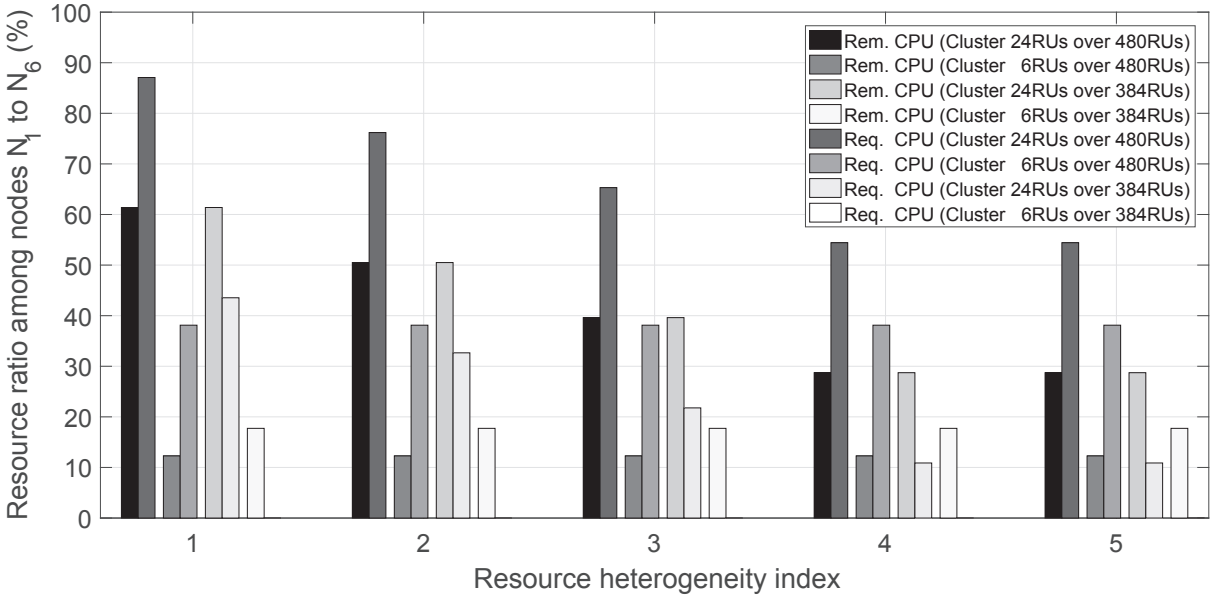


Figure A.15: Architecture du système de slicing RAN Runtime et de l'orchestration de services BEB.



Indice d'hétérogénéité des ressources	Ressource CPU													
	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}	N_{13}	N_{14}
1	32	32	32	32	32	32	2	2	2	2	2	2	2	2
2	48	16	32	32	32	32	1	1	1	1	3	3	3	3
3	32	32	48	48	16	16	1	1	1	1	3	3	3	3
4	48	16	16	16	48	48	2	2	2	2	2	2	2	2
5	48	16	16	16	48	48	3	3	3	3	1	1	1	1

(a) Ratio d'acceptation



(b) Ratio de CPU restant/requis

Figure A.16: Ratio d'acceptation et ratio de CPU restant/requis avec une ressource hétérogène.

- C.-Y. Chang, N. Nikaein, O. Arouk, K. Katsalis, A. Ksentini, T. Turletti, and K. Samdanis, "Slice orchestration for multi-service disaggregated ultra dense RANs," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 70-77, Aug. 2018.

A.8 Chapitre 7 — "Enabling Control Applications for Multi-Service RAN Programmability"

A.8.1 Motivation

Dans les deux chapitres précédents, nous mettons en évidence les services RAN Runtime et sa capacité à fournir une orchestration multiservice pour la plate-forme en tant que service (en anglais platform-as-a-service, PaaS). Étendu à partir du niveau PaaS, le logiciel en tant que service (en anglais, software-as-a-service, SaaS) peut consommer les applications de contrôle programmables telles que la gestion des ressources radio (RRM) et l'application de gestion du spectre (SMA) pour fournir les CLs pour chaque slice. De plus, pour déployer les applications de contrôle plug-and-play (P&P) pour des cas d'utilisation spécifiques, la plate-forme et les kits de développement logiciel (SDK) sont nécessaires. À cette fin, nous fournissons l'environnement d'exécution de l'application de contrôle en exploitant le SDK ainsi que le plan d'application pour l'exécution des CLs.

A.8.2 Résultats obtenues

Dans la figure A.17, nous réexaminons la solution RAN Runtime en mettant l'accent sur les applications de contrôle communes et dédiées. Les applications de contrôle communes peuvent fournir une CL partagée pour plusieurs tranches. Spécifiquement, il peut prendre en charge les décisions de contrôle customisés issues de différentes applications de contrôle spécifiques à un slice, résoudre leurs conflits et appliquer une stratégie réalisable au nœud RAN sous-jacent. D'autre part, l'application de contrôle dédiée peut être distribuée ou centralisée, en s'appuyant sur (1) le SDK pour accéder aux ressources, aux états et au traitement de la tranche, et (2) les données et à' l'API soit spécifique à chaque slice ou accessible via le slice. Notez que les applications dédiées centralisées peuvent être développées et/ou déployées par le fournisseur de services numériques (par exemple, over-the-top) qui utilise les données et l'API d'une ou de plusieurs tranches de réseau.

Pour prendre en charge les développements et les déploiements d'applications de contrôle spécifiques aux tranches, le SDK et le chaînage d'application de contrôle inter-domaine sont mis en évidence. Plus précisément, le SDK fournit plusieurs fonctionnalités, telles que (1) l'authentification, l'autorisation et la gestion, (2) la surveillance des métriques, (3) le contrôle et la délégation, et (4) la base de données de réseau sous forme d'un graphe. De plus, le plan d'application peut être formé en chaînant diverses applications de contrôle dédiées et/ou communes pour permettre l'automatisation et l'extensibilité des opérations de contrôle du réseau et améliorer la procédure de prise de décision sur différentes slice. Dans la figure A.18, nous fournissons le cas d'utilisation de l'optimisation du contenu sur le segment RAN, par exemple, dans lequel le chaînage de l'application de surveillance RAN et le débit adaptatif (an anglais, adaptive bitrate ABR) sont appliqués. Par conséquent, des segments vidéo de qualités différentes sont fournis sur la base du taux demandé mappé à partir de la valeur de l'indice de qualité de canal (en anglais channel quality indicator, CQI) pour maintenir la QoE vidéo. Deux expériences

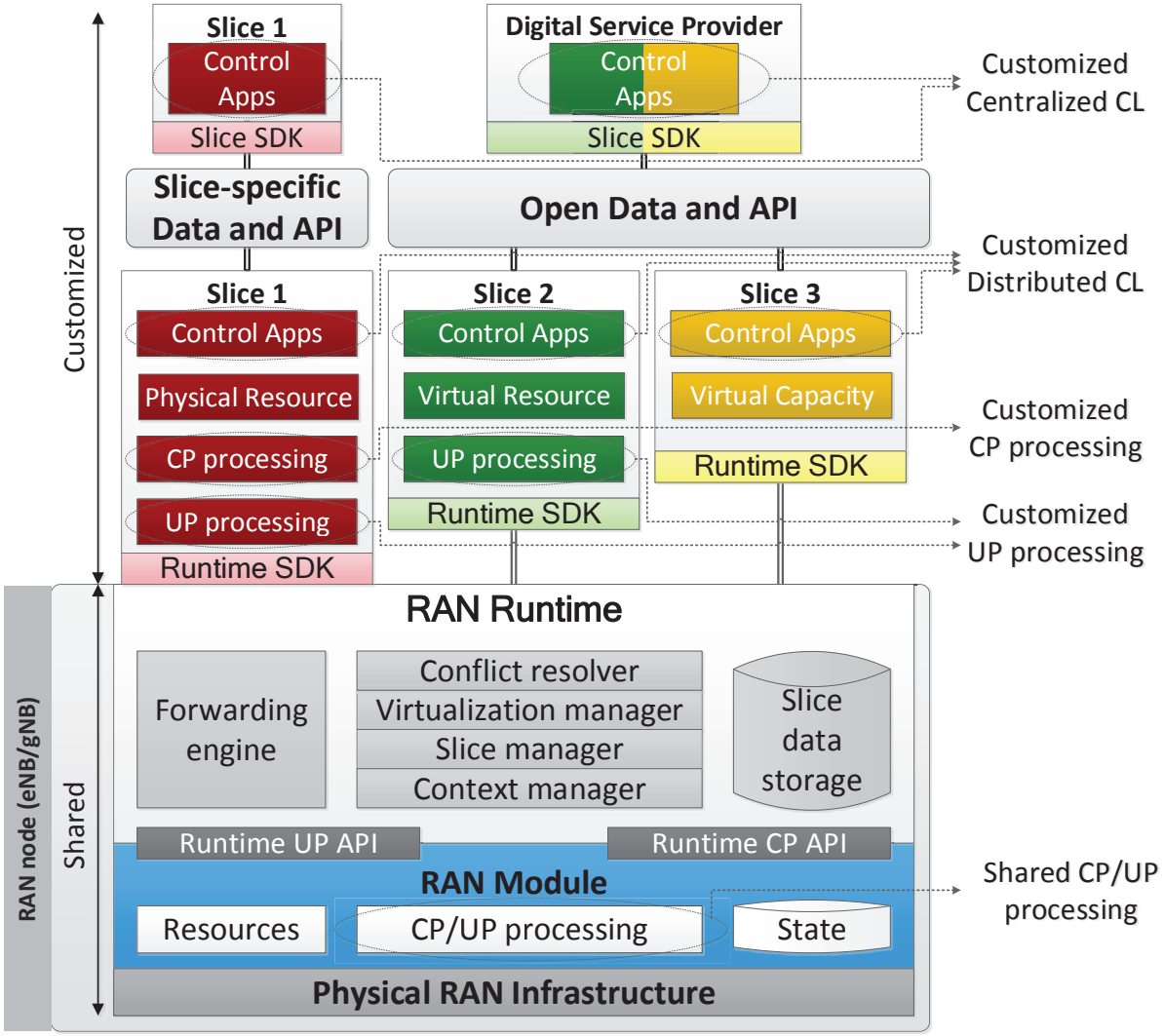


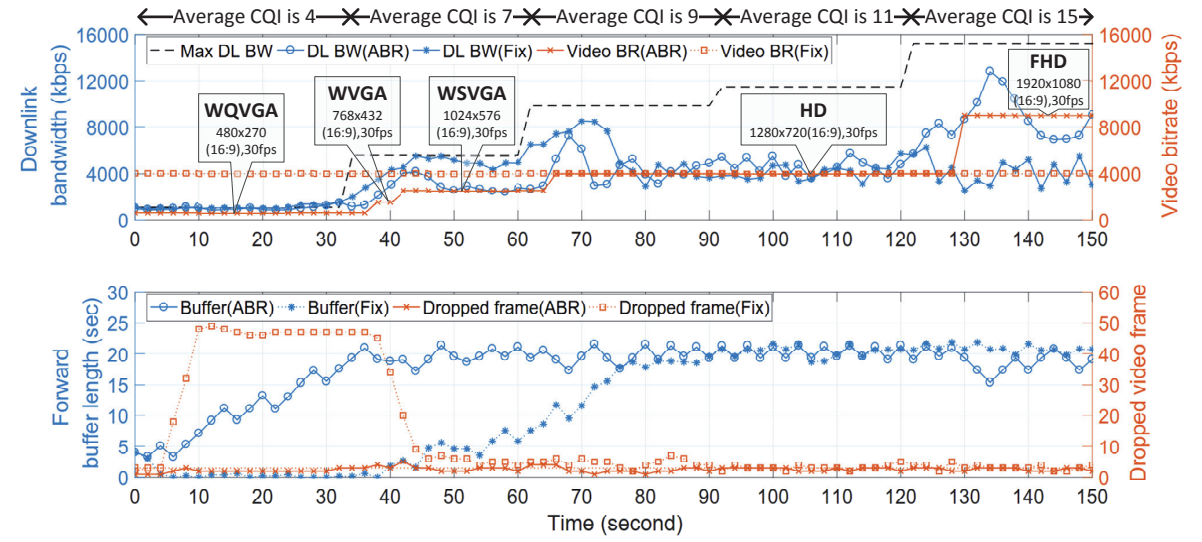
Figure A.17: Architecture de RAN Runtime et exemple avec trois instances de Slice.

sont menées à des fins de comparaison en faisant varier le CQI de 4 a 15. Dans le cas d'un tel chaînage d'applications, la qualité vidéo peut être modifiée de WQVGA, WVGA, WSVGA, HD à FHD avec peu des images vidéos perdu. Cependant, celui avec une qualité vidéo HD fixe souffrira d'une grande quantité d'images perdues et d'une longueur de tampon quasi nulle (lorsque le CQI est de 4) et d'une qualité vidéo inférieure (lorsque le CQI est au maximum 15).

Les travaux liés à ce chapitre sont les suivants:

- C.-Y. Chang and N. Nikaein, "5G Control Apps: Enabling Multiservice Programmability in a Disaggregated Radio Access Network," *IEEE Vehicular Technology Magazine*, vol. 13, no. 3, Dec. 2018.
- C.-Y. Chang, Ł. Kułacz, R. Schmidt, A. Kliks, and N. Nikaein, "Spectrum management application - A tool for flexible and efficient resource utilization," in *Proceedings of 2018 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018, pp.1-7.

CQI value	Downlink (Mbps)	Uplink (Mbps)
4	1.08	0.69
7	5.59	2.49
9	9.88	4.47
11	11.47	6.04
15	15.22	8.08



Acronym

WQVGA : Wide Quarter Video Graphics Array
WVGA : Wide Video Graphics Array
WSVGA : Wide Super Video Graphics Array
HD : High Definition
FHD : Full High Definition

Figure A.18: Débit maximal mesuré pour chaque valeur de CQI et les résultats obtenus.

Appendix B

Acronyms

2G	Second Generation
3G	Third Generation
3GPP	Third Generation Partnership Project
4G	Fourth Generation
5G	Fifth Generation
5GC	Fifth Generation Core Network
5GPPP	Fifth Generation Public-Private Partnership
ABR	Adaptive Bit Rate
ACB	Aggregated Common Bound
ACK	ACKnowledgment
API	Application Programming Interface
ASB	Aggregated Strict Bound
AWGN	Additive White Gaussian Noise
AxC	Antenna-Carriers
BB	Branch-and-Bound
BBU	BaseBand Unit
BS	Base Station
C-RAN	Cloud Radio Access Network
CAPEX	CAPital EXpenditure
CCE	Control Channel Element

CDF Cumulative Distribution Function

CI Control Information

CL Control Logic

CLB Conservative Lower Bound

CN Core Network

CoMP Coordinated Multi-Point

COTS Commercial-Of-The-Shelf

CP Control Plane

CPRI Common Public Radio Interface

CPU Central Processing Unit

CQI Channel Quality Indicator

CSI Channel State Information

CSMF Communication Service Management Function

CU Centralized Unit

D-RAN Distributed Radio Access Network

DAS Distributed Antenna System

DASH Dynamic Adaptive Streaming over HTTP

DECOR DEdicated COre Network

DFT Discrete Fourier Transform

DL Downlink

DPDK DataPlane Development Kit

DRB Dedicated Radio Bearer

DU Distributed Unit

E2E End-to-End

EDD Earliest Due Date

eDECOR Evolved DEdicated COre Network

eMBB Enhanced Mobile BroadBand

eNB Evolved Node B

ETSI European Telecommunications Standards Institute

F-RAN Fog Radio Access Network

FCFS First-Come First-Served

FDD Frequency Division Duplexing

FH Fronthaul

FHD Full High Definition

FIFO First-In-First-Out

FPGA Field-Programmable Gate Array

FPS Frame Per Second

gNB next Generation Node B

GPP General Purpose Processor

GPU Graphics Processing Unit

GSMA Global System for Mobile communications Association

GWCN GateWay Core Network

HARQ Hybrid Automatic Repeat reQuest

HD High Definition

HiD High Density

HLS High Level Split

HTTP HyperText Transfer Protocol

I/Q In-phase and Quadrature

IaaS Infrastructure-as-a-Service

ICI Inter-Carrier Interference

IDFT Inverse Discrete Fourier Transform

IEEE Institute of Electrical and Electronics Engineers

IMSI International Mobile Subscriber Identity

IMT International Mobile Telecommunications

IT Information Technology

ITU-R International Telecommunication Union Radiocommunication sector

ITU-T International Telecommunication Union Telecommunication standardization sector

KPI Key Performance Indicator

LAD Look-Ahead Depth

LLR Log-Likelihood Ratio

LLS Low Level Split

LoD Low Density

LPB Lower Provisioning Bound

LRB Least Remaining Bit

LST Least Slack Time

LTE Long Term Evolution

LTE-A Long Term Evolution-Advanced

M-CORD Mobile Central Office Re-architected as a Datacenter

M2M Machine-to-Machine

MAC Medium Access Control

MANO Management and Orchestration

MCS Modulation Coding Scheme

MEC Multi-access Edge Computing

MeD Medium Density

MH Midhaul

MIB Master Information Block

MIMO Multiple-Input Multiple-Output

MMKP Multi-dimensional Multiple-choice Knapsack Problem

MMSE Minimum Mean Square Error

mMTC Massive Machine-Type Communication

mmWave millimeter-Wave

MOCN Multi-Operator Core Network

MOP Multi-Objective Placement

MORAN Multi-Operator Radio Access Network

MTU Maximum Transmission Unit

MVNO Mobile Virtual Network Operator

NACK Negative ACKnowledgement

NFV Network Function Virtualization

NGFI Next Generation Fronthaul Interface

NGMN Next Generation Mobile Network

NPU Network Processing Unit

NSD Network Service Descriptor

NSI Network Slice Instance

NSMF Network Slice Management Function

NSSI Network Slice Subnet Instance

NSSMF Network Slice Subnet Management Function

NST Network Slice Template

NVS Network Virtualization Substrate

OAI OpenAirInterface

OBSAI Open Base Station Architecture Initiative

OFDM Orthogonal Frequency-Division Multiplexing

ONF Open Networking Foundation

OPEX OPerating EXpense

OPNFV Open Platform for Network Function Virtualization

ORI Open Radio Interface

OS Operating System

OSM Open Source Management and orchestration

OTT Over-The-Top

PaaS Platform-as-a-Service

PAPR Peak-to-Average Power Ratio

PDCP Packet Data Convergence Protocol

PDF Probability Distribution Function

PDU Protocol Data Unit

PHY PHYsical layer

PLMN Public Land Mobile Network

PNF Physical Network Function

PnP Plug-and-Play

PRACH Physical Random Access CHannel

PRB Physical Resource Block

PTP Precision Time Protocol

PUCCH Physical Uplink Control CHannel

PUSCH Physical Uplink shared CHannel

QoE Quality of Experience

QoS Quality of Service

RAN Radio Access Network

RAT Radio Access Technology

RBG Resource Block Group

REST Representational State Transfer

RF Radio Frequency

RLC Radio Link Control

RoE Radio over Ethernet

RRC Radio Resource Control

RRH Remote Radio Head

RRM Radio Resource Management

RRU Remote Radio Unit

RS Reference Signal

RTT Round Trip Time

RU Radio Head

Rx Receiver

S-NSSAI Single-Network Slice Selection Assistance Information

SaaS Software-as-a-Service

SCF Small Cell Forum

SCS Sub-Carrier Spacing

SD slice differentiator

SD-RAN Software-Defined Radio Access Network

SDAP Service Data Adaptation Protocol

SDK Software Development Kit

SDN Software-Defined Networking

SIB System Information Block

SINR Signal to Interference plus Noise Ratio

SLA Service Level Agreement

SLIB SLice Information Base

SMA Spectrum Management Application

SPT Shortest Processing Time

SST Slice/Service Type

syncE Synchronous Ethernet

TBS Transport Block Size

TIA Telecom Infra Project

TM Transmission Mode

TN Transport Network

TTI Transmission Time Interval

Tx Transmitter

UDN Ultra-Dense Network

UDP User Datagram Protocol

UE User Equipment

UL Uplink

uMTC Ultra-reliable MTC

UP User Plane

uRLLC Ultra-Reliable and Low Latency Communication

V2X Vehicular-to-everything

VNF Virtual Network Function

VOD Video-On-Demand

vRB Virtualized Resource Block

vRBG Virtualized Resource Block Group

vTBS Virtual Transport Block Size

WQVGA wide quarter video graphics array

WSVGA Wide Super Video Graphics Array

WVGA Wide Video Graphics Array

XaaS Everything-as-a-Service

xMBB eXtreme Mobile BroadBand

Bibliography

[1] ITU-R M.2410, "Minimum requirements related to technical performance for IMT-2020 radio interface(s)," Nov. 2017.

[2] Cisco, "Visual networking index: Global mobile data traffic forecast, 2016–2021," White Paper, Feb. 2017.

[3] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?" *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[4] ITU-R M.1645, "Framework and overall objectives of the future development of IMT-2000 and systems beyond IMT-2000," Jun. 2003.

[5] ITU-R M.2134, "Requirements related to technical system performance for IMT-Advanced radio interface(s)," Nov. 2008.

[6] ITU-R M.2083, "Framework and overall objectives of the future development of IMT for 2020 and beyond," Sep. 2015.

[7] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 6, pp. 1201–1221, Jun. 2017.

[8] P. Marsch, I. Da Silva, O. Bulakci, M. Tesanovic, S. E. El Ayoubi, T. Rosowski, A. Kaloxylos, and M. Boldi, "5G radio access network architecture: Design guidelines and key considerations," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 24–32, Nov. 2016.

[9] ITU-R M.2412, "Guidelines for evaluation of radio interface technologies for IMT-2020," Nov. 2017.

[10] F. Rancy, "IMT for 2020 and beyond," in *5G Outlook-Innovations and Applications*. River Publishers, Jun. 2016, ch. 6, pp. 69–84.

[11] A. Minokuchi, S. Isobe, H. Takahashi, and S. Nagata, "5G standardization trends at 3GPP," *NTT DOCOMO Technical Journal*, vol. 19, no. 3, pp. 5–12, Jan. 2018.

[12] IEEE Standards Association, "IEEE standards activities in 5G," May 2018.

[13] E. Hossain and M. Hasan, "5G cellular: Key enabling technologies and research challenges," *IEEE Instrumentation Measurement Magazine*, vol. 18, no. 3, pp. 11–21, Jun. 2015.

[14] I. F. Akyildiz, S. Nie, S.-C. Lin, and M. Chandrasekaran, "5G roadmap: 10 key enabling technologies," *Computer Networks*, vol. 106, pp. 17–48, Sep. 2016.

[15] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, thirdquarter 2016.

[16] A. Morgado, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "A survey of 5G technologies: regulatory, standardization and industrial perspectives," *Digital Communications and Networks*, vol. 4, no. 2, pp. 87–97, Sep. 2018.

[17] P. Demestichas, A. Georgakopoulos, K. Tsagkaris, and S. Kotrotsos, "Intelligent 5G networks: Managing 5G wirelessmobile broadband," *IEEE Vehicular Technology Magazine*, vol. 10, no. 3, pp. 41–50, Sep. 2015.

[18] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, thirdquarter 2017.

[19] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, Thirdquarter 2014.

[20] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016.

[21] 5GPPP Architecture Working Group, "View on 5G architecture (version 2.0)," Jul. 2017.

[22] *TS 28.530 Management and orchestration; Concepts, use cases and requirements (Release 15)*, 3GPP, Oct. 2018.

[23] *TS 23.214 Architecture enhancements for control and user plane separation of EPC nodes (Release 14)*, 3GPP, Sep. 2016.

[24] NGMN Alliance, "5G white paper," Feb. 2015.

[25] ITU-T Y.3150, "High-level technical characteristics of network softwarization for IMT-2020," Jan. 2018.

[26] D. Soldani and A. Manzalini, "Horizon 2020 and beyond: On the 5G operating system for a true digital society," *IEEE Vehicular Technology Magazine*, vol. 10, no. 1, pp. 32–42, Mar. 2015.

[27] L. Gavrilovska, V. Rakovic, and V. Atanasovski, "Visions towards 5G: Technical requirements and potential enablers," *Wireless Personal Communications*, vol. 87, no. 3, pp. 731–757, Apr. 2016.

[28] China Mobile Research Institute, "C-RAN: The road towards green RAN (version 2.5)," White Paper, Oct. 2011.

[29] C.-Y. Chang, R. Schiavi, N. Nikaein, T. Spyropoulos, and C. Bonnet, "Impact of packetization and functional split on C-RAN fronthaul performance," in *Proceedings of 2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–7.

[30] SCF, "Small cell virtualization functional splits and use cases," Jun. 2015.

[31] D. Wubben, P. Rost, J. S. Bartelt, M. Lalam, V. Savin, M. Gorgoglione, A. Dekorsy, and G. Fettweis, "Benefits and impact of cloud computing on 5G signal processing: Flexible centralization through Cloud-RAN," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 35–44, Nov. 2014.

[32] 3GPP, *TR 38.801 Study on new radio access technology: Radio access architecture and interfaces (Release 14)*, Mar. 2017.

[33] J. Wu, Z. Zhang, Y. Hong, and Y. Wen, "Cloud radio access network (C-RAN): A primer," *IEEE Network*, vol. 29, no. 1, pp. 35–41, Jan. 2015.

[34] O. Simeone, A. Maeder, M. Peng, O. Sahin, and W. Yu, "Cloud radio access network: Virtualizing wireless access for dense heterogeneous systems," *Journal of Communications and Networks*, vol. 18, no. 2, pp. 135–149, Apr. 2016.

[35] C. Ranaweera, E. Wong, A. Nirmalathas, C. Jayasundara, and C. Lim, "5G C-RAN with optical fronthaul: An analysis from a deployment perspective," *Journal of Lightwave Technology*, vol. 36, no. 11, pp. 2059–2068, Jun. 2018.

[36] I. A. Alimi, A. L. Teixeira, and P. P. Monteiro, "Toward an efficient C-RAN optical fronthaul for the future networks: A tutorial on technologies, requirements, challenges, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 708–769, Firstquarter 2018.

[37] D. Pliatsios, P. Sarigiannidis, S. Goudos, and G. K. Karagiannidis, "Realizing 5G vision through cloud RAN: Technologies, challenges, and trends," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, 2018.

[38] L. Guangjie, Z. Senjie, Y. Xuebin, L. Fanglan, N. Tin-fook, Z. Sunny, and K. Chen, "Architecture of GPP based, scalable, large-scale C-RAN BBU pool," in *Proceedings of 2012 IEEE Globecom Workshops*, Dec. 2012, pp. 267–272.

[39] B. Cheng, X. Mi, X. Xu, Z. Xu, X. Xu, and M. Zhao, "A real-time implementation of comp transmission based on cloud-ran infrastructure," in *Proceedings of 2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Aug. 2014, pp. 1033–1038.

[40] L. S. Ferreira, D. Pichon, A. Hatefi, A. Gomes, D. C. Dimitrova, T. Braun, G. Karagiannis, M. Karimzadeh, M. Branco, and L. M. Correia, "An architecture to offer cloud-based radio access network as a service," in *Proceedings of 2014 European Conference on Networks and Communications (EuCNC)*, Jun. 2014, pp. 1–5.

[41] S. Bhaumik, S. P. Chandrabose, M. K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, and T. Woo, "CloudIQ: A framework for processing base stations in a data

center," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (Mobicom'12)*, Aug. 2012, pp. 125–136.

[42] K. C. Garikipati, K. Fawaz, and K. G. Shin, "RT-OPEX: Flexible scheduling for Cloud-RAN processing," in *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies (CoNEXT'16)*, Dec. 2016, pp. 267–280.

[43] N. Yu, Z. Song, H. Du, H. Huang, and X. Jia, "Multi-resource allocation in cloud radio access networks," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[44] A. E. Kalør, M. I. Agurto, N. K. Pratas, J. J. Nielsen, and P. Popovski, "Statistical multiplexing of computations in C-RAN with tradeoffs in latency and energy," in *Proceedings of 2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 772–777.

[45] K. Wang and Y. Cen, "Real-time partitioned scheduling in Cloud-RAN with hard deadline constraint," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2017, pp. 1–6.

[46] X. Wang, K. Wang, S. Wu, D. Sheng, H. Jin, K. Yang, and S. Ou, "Dynamic resource scheduling in mobile edge cloud with cloud radio access network," *IEEE Transactions on Parallel and Distributed Systems*, 2018, to be published.

[47] B. M. Khorsandi and C. Raffaelli, "Bbu location algorithms for survivable 5G C-RAN over WDM," *Computer Networks*, vol. 144, pp. 53–63, Oct. 2018.

[48] J. Santos, D. Dinis, D. Riscado, G. Anjos, D. Belo, A. S. Oliveira, P. Monteiro, and N. B. Carvalho, "A flexible physical layer and fronthaul research testbed for C-RAN," *Microprocessors and Microsystems*, vol. 52, pp. 480–490, Jul. 2017.

[49] K. Sundaresan, M. Y. Arslan, S. Singh, S. Rangarajan, and S. V. Krishnamurthy, "FluidNet: A flexible cloud-based radio access network for small cells," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 915–928, Apr. 2016.

[50] C. Liu, K. Sundaresan, M. Jiang, S. Rangarajan, and G.-K. Chang, "The case for reconfigurable backhaul in cloud-RAN based small cell networks," in *2013 Proceedings IEEE INFOCOM*, Apr. 2013, pp. 1124–1132.

[51] X. Wang, C. Cavdar, L. Wang, M. Tornatore, Y. Zhao, H. S. Chung, H. H. Lee, S. Park, B. Mukherjee *et al.*, "Joint allocation of radio and optical resources in virtualized cloud RAN with CoMP," in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016, pp. 1–6.

[52] Y. Wang, X. Zhang, and D. Yang, "Evaluation methodology for fast switching cloud RAN systems," *IEEE Communications Letters*, vol. 21, no. 11, pp. 2404–2407, Nov. 2017.

[53] L. Chen, D. Yang, D. Zhang, C. Wang, J. Li *et al.*, "Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization," *Journal of Network and Computer Applications*, vol. 121, pp. 59–69, Nov. 2018.

[54] B. Lin, X. Pan, R. He, and S. Li, "Joint wireless-optical infrastructure deployment and layout planning for cloud-radio access networks," in *Proceedings of 2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Aug. 2014, pp. 1027–1032.

[55] Z. H. Fakhri, M. Khan, F. Sabir, and H. Al-Raweshidy, "A resource allocation mechanism for cloud radio access network based on cell differentiation and integration concept," *IEEE Transactions on Network Science and Engineering*, 2017, to be published.

[56] M. Khan, Z. H. Fakhri, and H. S. Al-Raweshidy, "Semistatic cell differentiation and integration with dynamic BBU-RRH mapping in cloud radio access network," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 289–303, Mar. 2018.

[57] R. Al-obaidi, A. Checko, H. Holm, and H. Christiansen, "Optimizing cloud-RAN deployments in real-life scenarios using microwave radio," in *Proceedings of 2015 European Conference on Networks and Communications (EuCNC)*, Jun. 2015, pp. 159–163.

[58] A. Abdelnasser and E. Hossain, "Resource allocation for an OFDMA cloud-RAN of small cells underlaying a macrocell," *IEEE Transactions on Mobile Computing*, vol. 15, no. 11, pp. 2837–2850, Nov. 2016.

[59] A. He, L. Wang, Y. Chen, K.-K. Wong, and M. Elkashlan, "Throughput and energy efficiency for S-FFR in massive MIMO enabled heterogeneous C-RAN," in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016, pp. 1–6.

[60] Y. Zhong, T. Q. Quek, and W. Zhang, "Complementary networking for C-RAN: spectrum efficiency, delay and system cost," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4639–4653, Jul. 2017.

[61] B.-S. Huang, Y.-H. Chiang, and W. Liao, "Remote radio head (RRH) deployment in flexible C-RAN under limited fronthaul capacity," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[62] D. Kim, Y. Yang, K. W. Sung, and J. Kang, "Cooperation strategies for partly wireless C-RAN," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1248–1251, Jun. 2018.

[63] Y. Shi, J. Zhang, and K. B. Letaief, "Group sparse beamforming for green cloud-RAN," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 2809–2823, May 2014.

[64] C. Pan, H. Zhu, N. J. Gomes, and J. Wang, "Joint user selection and energy minimization for ultra-dense multi-channel C-RAN with incomplete CSI," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 8, pp. 1809–1824, Aug. 2017.

[65] A. Younis, T. X. Tran, and D. Pompili, "Bandwidth and energy-aware resource allocation for cloud radio access networks," *IEEE Transactions on Wireless Communications*, 2018, to be published.

[66] W. Xia, J. Zhang, T. Q. Quek, S. Jin, and H. Zhu, "Power minimization based joint task scheduling and resource allocation in downlink C-RAN," *IEEE Transactions on Wireless Communications*, 2018, to be published.

[67] K. Guo, M. Sheng, J. Tang, T. Q. Quek, and Z. Qiu, "On the interplay between communication and computation in green C-RAN with limited fronthaul and computation capacity," *IEEE Transactions on Communications*, vol. 66, no. 7, pp. 3201–3216, Jul. 2018.

[68] C. Bluemm, Y. Zhang, P. Alvarez, M. Ruffini, and L. A. DaSilva, "Dynamic energy savings in Cloud-RAN: An experimental assessment and implementation," in *Proceedings of 2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 791–796.

[69] N. Saxena, A. Roy, and H. Kim, "Traffic-aware cloud RAN: A key for green 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 1010–1021, Apr. 2016.

[70] X. Lin and S. Wang, "Efficient remote radio head switching scheme in cloud radio access network: A load balancing perspective," in *Proceedings of IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, May 2017, pp. 1–9.

[71] Z. Li, D. Grace, and P. Mitchell, "Hotspot-oriented green frameworks for ultrasmall cell cloud radio access networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 703–717, Jan. 2018.

[72] J.-Y. Lin, C.-H. Lee, and H.-W. Tsao, "On the optimization of user-centric energy-efficient C-RAN," in *Proceedings of 2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[73] R. Jiao, X. Wen, Z. Lu, Y. Chen, H. Shao, and W. Jing, "Dynamic user-centric clustering algorithm based on energy efficiency in Cloud-RAN," in *Proceedings of 2017 24th International Conference on Telecommunications (ICT)*, May 2017, pp. 1–7.

[74] J. Choi, I. Sohn, and K. B. Lee, "Adaptive remote radio head control for cloud radio access networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, pp. 173:1–173:10, Jun. 2016.

[75] T. X. Tran and D. Pompili, "Dynamic radio cooperation for user-centric Cloud-RAN with computing resource sharing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 4, pp. 2379–2393, Apr. 2017.

[76] J. Duan, X. Lagrange, and F. Guilloud, "Analysis of different user grouping algorithms in a C-RAN downlink system," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[77] S. Parsaeefard, R. Dawadi, M. Derakhshani, T. Le-Ngoc, and M. Baghani, "Dynamic resource allocation for virtualized wireless networks in massive-mimo-aided and fronthaul-limited c-ran," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 10, pp. 9512–9520, Oct. 2017.

[78] Y. Wang, M. Peng, and K. Zhang, "Economy-efficient resource allocation in cloud radio access networks with fronthaul capacity constraints," in *Proceedings of 2016 International Conference on Computer, Information and Telecommunication Systems (CITS)*, Jul. 2016, pp. 1–5.

[79] U. S. Hashmi, S. A. R. Zaidi, and A. Imran, "User-centric cloud RAN: An analytical framework for optimizing area spectral and energy efficiency," *IEEE Access*, vol. 6, pp. 19 859–19 875, 2018.

[80] Y. Li, M. C. Gursoy, and S. Velipasalar, "Intercell interference-aware scheduling for delay sensitive applications in c-ran," in *Proceedings of 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Sep. 2017, pp. 1–5.

[81] A. M. Mahmood, A. Al-Yasiri, and O. Y. Alani, "Latency reduction by dynamic channel estimator selection in C-RAN networks using fuzzy logic," *Computer Networks*, vol. 138, pp. 44–56, Jun. 2018.

[82] A. Carreras, I. M. Delgado-Luque, F. J. Martín-Vega, M. C. Aguayo-Torres, G. Gómez, J. T. Entrambasaguas, E. Atxutegi, R. Solozabal, B. Blanco, J. O. Fajardo *et al.*, "Impact of front-haul delays in non-ideal cloud radio access networks," *Wireless Personal Communications*, pp. 1–18, Jun. 2018.

[83] L. Liu and R. Zhang, "Downlink sinr balancing in C-RAN under limited fronthaul capacity," in *Proceedings of 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016, pp. 3506–3510.

[84] Q.-D. Vu, K.-G. Nguyen, and M. Juntti, "Weighted max–min fairness for C-RAN multicasting under limited fronthaul constraints," *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 1534–1548, Apr. 2018.

[85] T. Kolding, L. C. Gimenez, and K. I. Pedersen, "Optimizing synchronous handover in cloud RAN," in *Proceedings of 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Sep. 2017, pp. 1–6.

[86] Z. Wang, D. W. K. Ng, V. W. Wong, and R. Schober, "Transmit beamforming for QoE improvement in C-RAN with mobile virtual network operators," in *Proceedings of 2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[87] M. Y. Lyazidi, N. Aitsaadi, and R. Langar, "A dynamic resource allocation framework in LTE downlink for cloud-radio access network," *Computer Networks*, vol. 140, pp. 101–111, Jul. 2018.

[88] U. Dötsch, M. Doll, H.-P. Mayer, F. Schaich, J. Segel, and P. Sehier, "Quantitative analysis of split base station processing and determination of advantageous architectures for LTE," *Bell Labs Technical Journal*, vol. 18, no. 1, pp. 105–128, Jun. 2013.

[89] J. Bartelt, P. Rost, D. Wubben, J. Lessmann, B. Melis, and G. Fettweis, "Fronthaul and backhaul requirements of flexibly centralized radio access networks," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 105–111, Oct. 2015.

[90] K. Miyamoto, S. Kuwano, J. Terada, and A. Otaka, "Analysis of mobile fronthaul bandwidth and wireless transmission performance in split-PHY processing architecture," *Optics express*, vol. 24, no. 2, pp. 1261–1268, Jan. 2016.

[91] A. Checko, A. P. Avramova, M. S. Berger, and H. L. Christiansen, "Evaluating C-RAN fronthaul functional splits in terms of network level energy and cost savings," *Journal of Communications and Networks*, vol. 18, no. 2, pp. 162–172, Apr. 2016.

[92] D. Wang, Y. Wang, S. Meng, and X. Zhang, "Game based wireless fronthaul C-RAN baseband function splitting and placement," in *Proceedings of 2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, Dec. 2016, pp. 400–404.

[93] A. Garcia-Saavedra, X. Costa-Perez, D. J. Leith, and G. Iosifidis, "FluidRAN: Optimized vRAN/MEC orchestration," in *Proceedings of IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1–9.

[94] G. Mountaser, M. L. Rosas, T. Mahmoodi, and M. Dohler, "On the feasibility of MAC and PHY split in Cloud RAN," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2017, pp. 1–6.

[95] N. Makris, P. Basaras, T. Korakis, N. Nikaein, and L. Tassiulas, "Experimental evaluation of functional splits for 5G Cloud-RANs," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[96] A. Garcia-Saavedra, J. X. Salvat, X. Li, and X. Costa-Perez, "WizHaul: On the centralization degree of cloud RAN next generation fronthaul," *IEEE Transactions on Mobile Computing*, vol. 17, no. 10, pp. 2452–2466, Oct. 2018.

[97] J. Kang, O. Simeone, J. Kang, and S. Shamai, "Control-data separation across edge and cloud for uplink communications in C-RAN," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2017, pp. 1–6.

[98] S. Khalili and O. Simeone, "Uplink HARQ for cloud ran via separation of control and data planes," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4005–4016, May 2017.

[99] J. Yao and N. Ansari, "Joint caching in fronthaul and backhaul constrained C-RAN," in *Proceedings of GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec. 2017, pp. 1–6.

[100] Z. Zhang, D. Liu, and Y. Yuan, "Layered hierarchical caching for SVC-based HTTP adaptive streaming over c-ran," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2017, pp. 1–6.

[101] S.-H. Park, O. Simeone, O. Sahin, and S. S. Shitz, "Fronthaul compression for cloud radio access networks: Signal processing advances inspired by network information theory," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 69–79, Nov. 2014.

[102] X. Rao and V. K. Lau, "Distributed fronthaul compression and joint signal recovery in cloud-ran," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 1056–1065, Feb. 2015.

[103] Z. Utkovski, O. Simeone, T. Dimitrova, and P. Popovski, "Random access in C-RAN for user activity detection with limited-capacity fronthaul," *IEEE Signal Processing Letters*, vol. 24, no. 1, pp. 17–21, Jan. 2017.

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

[105] T. Li, C. S. Magurawalage, K. Wang, K. Xu, K. Yang, and H. Wang, "On efficient offloading control in cloud radio access network with mobile edge computing," in *Proceedings of 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Jun. 2017, pp. 2258–2263.

[106] A. Shojaeifard, K.-K. Wong, W. Yu, G. Zheng, and J. Tang, "Full-duplex cloud radio access network: Stochastic design and analysis," *IEEE Transactions on Wireless Communications*, 2018, to be published.

[107] J. Kang, O. Simeone, J. Kang, and S. Shamai, "Layered downlink precoding for C-RAN systems with full dimensional mimo," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2170–2182, Mar. 2017.

[108] S. Shrestha, J. Lee, and S. Chong, "Virtualization and slicing of wireless mesh network," in *Proceedings of International Conference on Future Internet Technologies*, 2008.

[109] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T.-Y. Huang, P. Kazemian, M. Kobayashi, J. Naous *et al.*, "Carving research slices out of your production networks with OpenFlow," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 129–130, Jan. 2010.

[110] ITU-T focus group, "IMT-2020 deliverables," 2017.

[111] NGMN Alliance, "Description of network slicing concept (version 1.0.8)," Sep. 2016.

[112] GSMA, "Network slicing use case requirements," Apr. 2018.

[113] *TR 38.804 Study on new radio access technology: Radio Interface Protocol Aspects (Release 14)*, 3GPP, Mar. 2017.

[114] ONF, "M-cord as an open reference solution for 5g enablement," CORD Project White paper, Feb. 2017.

[115] *TS 23.251 Network sharing; Architecture and functional description (Release 15)*, 3GPP, Jun. 2018.

[116] M. Hoffmann and M. Staufer, "Network virtualization for future mobile networks: General architecture and applications," in *Proceedings of 2011 IEEE International Conference on Communications Workshops (ICC)*, Jun. 2011, pp. 1–5.

[117] A. Khan, W. Kellerer, K. Kozu, and M. Yabusaki, "Network sharing in the next mobile network: TCO reduction, management flexibility, and operational independence," *IEEE Communications Magazine*, vol. 49, no. 10, pp. 134–142, Oct. 2011.

[118] L. Doyle, J. Kibilda, T. K. Forde, and L. DaSilva, "Spectrum without bounds, networks without borders," *Proceedings of the IEEE*, vol. 102, no. 3, pp. 351–365, Mar. 2014.

[119] T. Frisanco, P. Tafertshofer, P. Lurin, and R. Ang, "Infrastructure sharing and shared operations for mobile network operators from a deployment and operations view," in *Proceedings of NOMS 2008 - 2008 IEEE Network Operations and Management Symposium*, Apr. 2008, pp. 129–136.

[120] D.-E. Meddour, T. Rasheed, and Y. Gourhant, "On the role of infrastructure sharing for mobile network operators in emerging markets," *Computer Networks*, vol. 55, no. 7, pp. 1576–1591, May 2011.

[121] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "NVS: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333–1346, Oct. 2012.

[122] T. Guo and R. Arnott, "Active LTE RAN sharing with partial resource reservation," in *Proceedings of 2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, Sep. 2013, pp. 1–5.

[123] X. Costa-Pérez, J. Swetina, T. Guo, R. Mahindra, and S. Rangarajan, "Radio access network virtualization for future mobile carrier networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 27–35, Jul. 2013.

[124] R. Mahindra, M. A. Khojastepour, H. Zhang, and S. Rangarajan, "Radio access network sharing in cellular networks," in *Proceedings of 2013 21st IEEE International Conference on Network Protocols (ICNP)*, Oct. 2013, pp. 1–10.

[125] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "CellSlice: Cellular wireless resource slicing for active RAN sharing," in *Proceedings of 2013 Fifth International Conference on Communication Systems and Networks (COMSNETS)*, Jan. 2013, pp. 1–10.

[126] J. He and W. Song, "AppRAN: Application-oriented radio access network sharing in mobile networks," in *Proceedings of 2015 IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 3788–3794.

[127] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, May 2017.

[128] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing & softwarization: A survey on principles, enabling technologies & solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, thirdquarter 2018.

[129] A. Kaloxylos, "A survey and an analysis of network slicing in 5G networks," *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 60–65, Mar. 2018.

[130] A. Devlic, A. Hamidian, D. Liang, M. Eriksson, A. Consoli, and J. Lundstedt, "NESMO: Network slicing management and orchestration framework," in *Proceedings of 2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 1202–1208.

[131] T. Taleb, B. Mada, M.-I. Corici, A. Nakao, and H. Flinck, "PERMIT: Network slicing for personalized 5G mobile telecommunications," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 88–93, May 2017.

[132] X. An, R. Trivisonno, H. Einsiedler, D. von Hugo, K. Haensge, X. Huang, Q. Shen, D. Corujo, K. Mahmood, D. Trossen *et al.*, "Architecture modularisation for next generation mobile networks," in *Proceedings of 2017 European Conference on Networks and Communications (EuCNC)*, Jul. 2017, pp. 1–6.

[133] I. Chih-Lin, S. Han, Z. Xu, S. Wang, Q. Sun, and Y. Chen, "New paradigm of 5G wireless Internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 474–482, Mar. 2016.

[134] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, Jul. 2016.

[135] S. Sharma, R. Miller, and A. Francini, "A cloud-native approach to 5G network slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 120–127, Aug. 2017.

[136] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32–39, Jul. 2016.

[137] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, "Mobile traffic forecasting for maximizing 5g network slicing resource utilization," in *Proceedings of INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, May 2017, pp. 1–9.

[138] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung, "Network slicing based 5G and future mobile networks: Mobility, resource management, and challenges," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 138–145, Aug. 2017.

[139] C. Campolo, A. Molinaro, A. Iera, and F. Menichella, "5G network slicing for vehicle-to-everything services," *IEEE Wireless Communications*, vol. 24, no. 6, pp. 38–45, Dec. 2017.

[140] A. Tzanakaki, M. Anastasopoulos, I. Berberana, D. Syrivelis, P. Flegkas, T. Korakis, D. C. Mur, I. Demirkol, J. Gutiérrez, E. Grass *et al.*, "Wireless-optical network convergence: Enabling the 5G architecture to support operational and end-user services," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 184–192, Oct. 2017.

[141] A. Rostami, P. Ohlen, K. Wang, Z. Ghebretensae, B. Skubic, M. Santos, and A. Vidal, "Orchestration of RAN and transport networks for 5G: An SDN approach," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 64–70, Apr. 2017.

[142] K. Han, S. Li, S. Tang, H. Huang, S. Zhao, G. Fu, and Z. Zhu, "Application-driven end-to-end slicing: When wireless network virtualization orchestrates with NFV-based mobile edge computing," *IEEE Access*, vol. 6, pp. 26 567–26 577, 2018.

[143] V. Petrov, M. A. Lema, M. Gapeyenko, K. Antonakoglou, D. Moltchanov, F. Sardis, A. Samuylov, S. Andreev, Y. Koucheryavy, and M. Dohler, "Achieving end-to-end reliability of mission-critical traffic in softwarized 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 485–501, Mar. 2018.

[144] G. García, M. Gramaglia, P. Serrano, and A. Banchs, "POSENS: A practical open-source solution for end-to-end network slicing," *IEEE Wireless Communications*, 2018, to be published.

[145] K. Katsalis, N. Nikaein, E. Schiller, R. Favraud, and T. I. Braun, "5G architectural design patterns," in *Proceedings of 2016 IEEE International Conference on Communications Workshops (ICC)*, May 2016, pp. 32–37.

[146] M. Yang, Y. Li, D. Jin, L. Su, S. Ma, and L. Zeng, "OpenRAN: A software-defined RAN architecture via virtualization," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 549–550, Aug. 2013.

[147] I. F. Akyildiz, P. Wang, and S.-C. Lin, "SoftAir: A software defined networking architecture for 5G wireless systems," *Computer Networks*, vol. 85, pp. 1–18, Jul. 2015.

[148] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "SoftRAN: Software defined radio access network," in *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN '13)*, Aug. 2013, pp. 25–30.

[149] V. Yazıcı, U. C. Kozat, and M. O. Sunay, "A new control plane for 5G network architecture with a case study on unified handoff, mobility, and routing management," *IEEE communications magazine*, vol. 52, no. 11, pp. 76–85, Nov. 2014.

[150] R. Yu, G. Xue, M. Bennis, X. Chen, and Z. Han, "HSDRAN: Hierarchical software-defined radio access network for distributed optimization," *IEEE Transactions on Vehicular Technology*, 2018, to be published.

[151] T. Chen, H. Zhang, X. Chen, and O. Tirkkonen, "SoftMobile: Control evolution for future heterogeneous mobile networks," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 70–78, Dec. 2014.

[152] M. Bansal, J. Mehlman, S. Katti, and P. Levis, "OpenRadio: A programmable wireless dataplane," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, Aug. 2012, pp. 109–114.

[153] W. Wu, L. E. Li, A. Panda, and S. Shenker, "PRAN: Programmable radio access networks," in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks (HotNets-XIII)*, Oct. 2014, pp. 6:1–6:7.

[154] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. P. Kontovasilis, "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies (CoNEXT '16)*, Dec. 2016, pp. 427–441.

[155] Y. Zaki, L. Zhao, C. Goerg, and A. Timm-Giel, "LTE mobile network virtualization," *Mobile Networks and Applications*, vol. 16, no. 4, pp. 424–432, Aug. 2011.

[156] C. Liang and F. R. Yu, "Wireless virtualization for next generation mobile cellular networks," *IEEE wireless communications*, vol. 22, no. 1, pp. 61–69, Feb. 2015.

[157] P. Rost, I. Berberana, A. Maeder, H. Paul, V. Suryaprakash, M. Valenti, D. Wübben, A. Dekorsy, and G. Fettweis, "Benefits and challenges of virtualization in 5G radio access networks," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 75–82, Dec. 2015.

[158] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network store: Exploring slicing in future 5G networks," in *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture (MobiArch'15)*, Sep. 2015, pp. 8–13.

[159] K. Katsalis, N. Nikaein, E. J. Schiller, A. Ksentini, and T. Braun, "Network slices towards 5G communications: Slicing the LTE network," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 146–154, Aug. 2017.

[160] C. Ramirez-Perez and V. Ramos, "SDN meets SDR in self-organizing networks: Fitting the pieces of network management," *IEEE Communications Magazine*, vol. 54, no. 1, pp. 48–57, Jan. 2016.

[161] J. Liu, T. Zhao, S. Zhou, Y. Cheng, and Z. Niu, "CONCERT: A cloud-based architecture for next-generation cellular systems," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 14–22, Dec. 2014.

[162] A. Tzanakaki, M. P. Anastasopoulos, and D. Simeonidou, "Optical networking interconnecting disaggregated compute resources: An enabler of the 5G vision," in *Proceedings of 2017 International Conference on Optical Network Design and Modeling (ONDM)*, May 2017, pp. 1–6.

[163] N. Gkatzios, M. Anastasopoulos, A. Tzanakaki, and D. Simeonidou, "Compute resource disaggregation: An enabler for efficient 5G RAN softwarisation," in *Proceedings of 2018 European Conference on Networks and Communications (EuCNC)*, Jun. 2018, pp. 1–5.

[164] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, Oct. 2014.

[165] N. Nikaein, C.-Y. Chang, and K. Alexandris, "Mosaic5G: Agile and flexible service platforms for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 3, pp. 29–34, Sep. 2018.

[166] N. Nikaein, "Processing radio access network functions in the cloud: Critical issues and modeling," in *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services (MCS'15)*, Sep. 2015, pp. 36–43.

[167] CPRI, "Interface specification v7.0," Oct. 2015.

[168] ORI, "Ori interface specification; part 1: Low layers (release 4)," Oct. 2014.

[169] OBSAI, "Bts system reference document version 2.0," 2006.

[170] CPRI, "ecpri interface specification v1.0," Aug. 2017.

[171] IEEE 1914 working group. P1914.3: Standard for radio over Ethernet encapsulations and mappings. [Online]. Available: http://sites.ieee.org/sagroups-1914/p1914-3/

[172] China Mobile Research Institute, "Next generation fronthaul interface (version 1.0)," White Paper, Jun. 2015.

[173] xRAN Fronthaul Working Group, "xran fronthaul control, user and synchronization plane (version 2.0)," Jul. 2018.

[174] I. Chih-Lin, Y. Yuan, J. Huang, S. Ma, C. Cui, and R. Duan, "Rethink fronthaul for soft RAN," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 82–88, Sep. 2015.

[175] NGMN Alliance, "Further study on critical C-RAN technologies (version 1.0)," 2015.

[176] I. Chih-Lin, J. Huang, R. Duan, C. Cui, J. X. Jiang, and L. Li, "Recent progress on C-RAN centralization and cloudification," *IEEE Access*, vol. 2, pp. 1030–1039, 2014.

[177] 3GPP, *TS 38.401 NG-RAN; Architecture description (Release 15)*, Jan. 2018.

[178] ——, *TR 36.756 Study on architecture evolution for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) (Release 15)*, Sep. 2017.

[179] *TR 38.816 Study on CU-DU lower layer split for NR (Release 15)*, 3GPP, Jan. 2018.

[180] Telecom Infra Project, "Creating an ecosystem for vRANs supporting non-ideal fronthaul," 2018.

[181] NGMN Alliance, "Guidelines for LTE backhaul traffic estimation (version 0.4.2)," Feb. 2011.

[182] *TS 25.942 Radio Frequency (RF) system scenarios (Release 10)*, 3GPP, Jul. 2012.

[183] *TR 36.814 Further advancements for E-UTRA physical layer aspects (Release 9)*, 3GPP, Mar. 2010.

[184] *TR 36.942 Radio Frequency (RF) system scenarios (Release 10)*, 3GPP, Jul. 2012.

[185] NGMN Alliance, "NGMN radio access performance evaluation methodology (version 1.0.0)," Jan. 2008.

[186] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-dense networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2522–2545, Fourthquarter 2016.

[187] Nokia, "Ultra dense network (UDN)," White Paper, 2016.

[188] NGMN Alliance, "RAN evolution project backhaul and fronthaul evolution (version 1.01)," Mar. 2015.

[189] N. J. Gomes, P. Chanclou, P. Turnbull, A. Magee, and V. Jungnickel, "Fronthaul evolution: From CPRI to Ethernet," *Optical Fiber Technology*, vol. 26, pp. 50–58, Dec. 2015.

[190] J. Liu, S. Xu, S. Zhou, and Z. Niu, "Redesigning fronthaul for next-generation networks: Beyond baseband samples and point-to-point links," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 90–97, Oct. 2015.

[191] "Ieee standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pp. 1–300, Jul. 2008.

[192] ITU-T G.711, "Pulse code modulation (PCM) of voice frequencies," Nov. 1988.

[193] Intel. DPDK: Dataplane development kit. [Online]. Available: https://www.dpdk.org/

[194] L. Rizzo, "Netmap: A novel framework for fast packet I/O," in *Proceedings of the 2012 USENIX Conference on Annual Technical Conference*, 2012, pp. 101–112.

[195] ITU-T G.8262, "Timing characteristics of synchronous Ethernet equipment slave clock," Jan. 2015.

[196] S. Venkatesan, A. Lozano, and R. Valenzuela, "Network MIMO: Overcoming intercell interference in indoor wireless systems," in *Proceedings of 2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers (ACSSC'07)*, Nov. 2007, pp. 83–87.

[197] *TS 36.104 Base Station (BS) radio transmission and reception (Release 15)*, 3GPP, Sep. 2017.

[198] A. Wilzeck, Q. Cai, M. Schiewer, and T. Kaiser, "Effect of multiple carrier frequency offsets in MIMO SC-FDMA systems," in *Proceedings of the International ITG/IEEE Workshop on Smart Antennas*, 2007.

[199] M. Rupp, S. Schwarz, and M. Taranetz, *The Vienna LTE-Advanced Simulators*. Springer, 2016.

[200] X. Wang and H. V. Poor, "Iterative (Turbo) Soft Interference Cancellation and Decoding for Coded CDMA," *IEEE Transactions on communications*, vol. 47, no. 7, pp. 1046–1061, Jul. 1999.

[201] N. Kim, Y. Lee, and H. Park, "Performance analysis of mimo system with linear mmse receiver," *IEEE Transactions on Wireless Communications*, vol. 7, no. 11, pp. 4474–4478, Nov. 2008.

[202] A. M. Tulino and S. Verdú, *Random matrix theory and wireless communications*. Now Publishers Inc., 2004.

[203] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack problems*. Springer, 2004.

[204] J. Arora, *Introduction to Optimum Design*. Elsevier, 2012.

[205] NGMN Alliance, "Perspectives on vertical industries and implications for 5g (version 2.0)," Sep. 2016.

[206] 3GPP, *TR 28.801 Study on management and orchestration of network slicing for next generation network (Release 15)*, Sep. 2017.

[207] A. Nakao, P. Du, Y. Kiriha, F. Granelli, A. A. Gebremariam, T. Taleb, and M. Bagaa, "End-to-end network slicing for 5G mobile networks," *Journal of Information Processing*, vol. 25, pp. 153–163, 2017.

[208] *TR 23.799 Study on Architecture for Next Generation System (Release 14)*, 3GPP, Dec. 2016.

[209] V. G. Nguyen and Y. H. Kim, "Slicing the next mobile packet core network," in *Proceedings of 2014 11th International Symposium on Wireless Communications Systems (ISWCS)*, Aug. 2014, pp. 901–904.

[210] T. Taleb, M. Corici, C. Parada, A. Jamakovic, S. Ruffino, G. Karagiannis, and T. Magedanz, "EASE: EPC as a service to ease mobile core network deployment over cloud," *IEEE Network*, vol. 29, no. 2, pp. 78–88, Mar. 2015.

[211] Z. A. Qazi, M. Walls, A. Panda, V. Sekar, S. Ratnasamy, and S. Shenker, "A high performance packet core for next generation cellular networks," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM'17)*, ser. SIGCOMM '17, Aug. 2017, pp. 348–361.

[212] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on RAN: Flexibility and resources abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.

[213] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom'17)*, Oct. 2017, pp. 127–140.

[214] *TR 23.707 Architecture enhancements for dedicated core networks; Stage 2 (Release 13)*, 3GPP, Dec. 2014.

[215] *TR 23.711 Enhancements of Dedicated Core Networks selection mechanism (Release 14)*, 3GPP, Sep. 2016.

[216] A. Gudipati, L. E. Li, and S. Katti, "RadioVisor: A slicing plane for radio access network," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking (HotSDN'14)*, Aug. 2014, pp. 237–238.

[217] A. Aijaz, "Hap – SliceR: A radio resource slicing framework for 5G networks with haptic communications," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2285–2296, Sep. 2018.

[218] D. Marabissi and R. Fantacci, "Heterogeneous public safety network architecture based on RAN slicing," *IEEE Access*, vol. 5, pp. 24 668–24 677, 2017.

[219] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, N. Sastry, O. Holland, S. Tayade, B. Han, D. Bega *et al.*, "Network slicing to enable scalability and flexibility in 5g mobile networks," *IEEE Communications magazine*, vol. 55, no. 5, pp. 72–79, May 2017.

[220] O. Sallent, J. Perez-Romero, R. Ferrus, and R. Agusti, "On radio access network slicing from a radio resource management perspective," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 166–174, Oct. 2017.

[221] R. Ferrus, O. Sallent, J. Perez-Romero, and R. Agusti, "On 5G radio access network slicing: Radio interface protocol features and configuration," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 184–192, May 2018.

[222] M. Kablan, A. Alsudais, E. Keller, and F. Le, "Stateless network functions: Breaking the tight coupling of state and processing," in *Proceedings of 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, Mar. 2017, pp. 97–112.

[223] J. Kim, D. Kim, and S. Choi, "3GPP SA2 architecture and functions for 5G mobile communication system," *ICT Express*, vol. 3, no. 1, pp. 1–8, Mar. 2017.

[224] A. A. Zaidi, R. Baldemair, H. Tullberg, H. Bjorkegren, L. Sundstrom, J. Medbo, C. Kilinc, and I. Da Silva, "Waveform and numerology to support 5G services and requirements," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 90–98, Nov. 2016.

[225] *TS 38.211 NR; Physical channels and modulation (Release 15)*, 3GPP, Jan. 2018.

[226] J. Moysen, M. Garcia-Lozano, L. Giupponi, and S. Ruiz, "Conflict resolution in mobile networks: A self-coordination framework based on non-dominated solutions and machine learning for data analytics," *IEEE Computational Intelligence Magazine*, vol. 13, no. 2, pp. 52–64, May 2018.

[227] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 99–110, Aug. 2013.

[228] O. Arouk, N. Nikaein, and T. Turletti, "Multi-objective placement of virtual network function chains in 5G," in *Proceedings of 2017 IEEE 6th International Conference on Cloud Networking (CloudNet)*, Sep. 2017, pp. 1–6.

[229] A. Caprara and M. Monaci, "On the two-dimensional knapsack problem," *Operations Research Letters*, vol. 32, no. 1, pp. 5–14, 2004.

[230] J. Van De Belt, H. Ahmadi, and L. E. Doyle, "A dynamic embedding algorithm for wireless network virtualization," in *Proceedings of 2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, Sep. 2014, pp. 1–6.

[231] M. Yang, Y. Li, L. Zeng, D. Jin, and L. Su, "Karnaugh-map like online embedding algorithm of wireless virtualization," in *Proceedings of The 15th International Symposium on Wireless Personal Multimedia Communications*, Sep. 2012, pp. 594–598.

[232] *TS 23.501 System Architecture for the 5G System; Stage 2 (Release 15)*, 3GPP, Jul. 2017.

[233] ETSI NFV ISG, *GS NFV 002 Network Functions Virtualisation (NFV); Architectural Framework (v 1.2.1)*, Dec. 2014.

[234] ONF, *TR-526 Applying SDN architecture to 5G slicing*, Apr. 2016.

[235] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming abstractions for software-defined wireless networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, Jun. 2015.

[236] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN-key technology enablers for 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, Nov. 2017.

[237] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith, "srsLTE: An open-source platform for LTE evolution and experimentation," in *Proceedings of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization (WiNTECH '16)*, Oct. 2016, pp. 25–32.

[238] A. Sutton, "5G network architecture," *Journal of The Institute of Telecommunications Professionals*, vol. 12, no. 1, pp. 9–15, 2018.

[239] *TR 38.876 Study on eNB(s) Architecture Evolution for E-UTRAN and NG-RAN (Release 15)*, 3GPP, May 2018.

[240] 3GPP, *TS 38.470 NG-RAN; F1 general aspects and principles (Release 15)*, Jan. 2018.

[241] A. Al-Saadi, R. Setchi, and Y. Hicks, "Semantic reasoning in cognitive networks for heterogeneous wireless mesh systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 3, pp. 374–389, Sep. 2017.

[242] N. Dmitry and S.-S. Manfred, "On micro-services architecture," *International Journal of Open Information Technologies*, vol. 2, no. 9, pp. 24–27, 2014.

[243] H. Ishii, Y. Kishiyama, and H. Takahashi, "A novel architecture for LTE-B: C-plane/U-plane split and phantom cell concept," in *Proceedings of 2012 IEEE Globecom Workshops*, Dec. 2012, pp. 624–630.

[244] Z. Niu, Y. Wu, J. Gong, and Z. Yang, "Cell zooming for cost-efficient green cellular networks," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 74–79, Nov. 2010.

[245] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: Physical layer informed adaptive video streaming over LTE," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*, Sep. 2015, pp. 413–425.

[246] nPerf. What's nPerf speed test? How does it work? [Online]. Available: https://www.nperf.com/en/

[247] ETSI ESI ISG, *Improved operator experience through experiential networked intelligence (ENI)*, White Paper, Oct. 2017.

[248] M. Shehata, A. Elbanna, F. Musumeci, and M. Tornatore, "C-RAN baseband pooling: Cost model and multiplexing gain analysis," in *Proceedings of 2017 19th International Conference on Transparent Optical Networks (ICTON)*, Jul. 2017, pp. 1–4.