5G RAN: Functional Split Orchestration Optimization

Salma Matoussi, Ilhem Fajjari, *Member, IEEE*, Salvatore Costanzo, Nadjib Aitsaadi, *Member, IEEE*, and Rami Langar, *Member, IEEE*

***Abstract*— 5G RAN aims to evolve new technologies spanning the Cloud infrastructure, virtualization techniques and Software Defined Network capabilities. Advanced solutions are introduced to split the functions of the Radio Access Network (RAN) between centralized and distributed locations. Such paradigms improve RAN flexibility and reduce the infrastructure deployment cost without impacting the user quality of service. We propose a novel functional split orchestration scheme that aims at minimizing the RAN deployment cost, while considering the requirements of its processing network functions and the capabilities of the Cloud infrastructure. With a fine grained approach on user basis, we show that the proposed solution optimizes both processing and bandwidth resource usage, while minimizing the overall energy consumption compared to i) cell-centric, ii) distributed and iii) centralized Cloud-RAN approaches. Moreover, we evaluate the effectiveness of our proposal in a 5G experimental prototype, based on Open Air Interface (OAI). We show that our solution achieves good performance in terms of total deployment cost and resolution time.**

***Index Terms*— C-RAN, NFV, functional split, heuristic, PSO, optimization, OAI.**

I. Introduction

GLOBAL mobile data traffic is forecasted to increase seven-fold between 2017 and 2022 [1]. In order to meet such a huge demand, next-generation mobile networks need to support a scaling system capacity. However, it is not expected that operators revenue-per-bit will cope with the cost-per-bit investments [1], which will shrink their CAPital EXpenditure (CAPEX) and OPerational EXpenditure (OPEX) budgets. This issue is especially relevant for the Radio Access Network (RAN), that is considered as the costliest and the most resource-demanding part of mobile networks [2].

Manuscript received April 15, 2019; revised November 20, 2019; accepted January 28, 2020. Date of publication June 3, 2020; date of current version June 29, 2020. This work was supported in part by the FUI SCORPION Project under Grant 17/00464 and in part by the CNRS PRESS Project under Grant 239953. This article was presented at the Proceedings of the 2018 IEEE International Conference on Communications (ICC 2018). *(Corresponding author: Salma Matoussi.)*

Salma Matoussi and Rami Langar are with LIGM CNRS-UMR 8049, University Gustave Eiffel (UGE), 77420 Marne-la-Vallée, France (e-mail: Salma.Matoussi@u-pem.fr; Rami.Langar@u-pem.fr).

Ilhem Fajjari and Salvatore Costanzo are with Orange-Labs, 92320 Châtillon, France (e-mail: Ilhem.Fajjari@orange.com; Salvatore.Costanzo@orange.com).

Nadjib Aitsaadi is with LI-PaRAD EA 7432, University UVSQ Paris-Saclay, F-78280 Guyancourt, France (e-mail: nadjib.aitsaadi@uvsq.fr).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2020.2999685

To face the aforementioned challenge, the Cloud RAN (C-RAN) architecture has been proposed in 2011 [2] aiming at leveraging the Cloud infrastructure capabilities to reduce the RAN deployment cost. C-RAN proposes to cloudify the BaseBand signal processing Units (BBUs), that are traditionally located near macro cells (eNodeBs). To do so, BBUs are virtualized and hosted in a remote Cloud site [3], leaving simple antenna units known as Remote Radio Heads (RRHs) at the access site connected via a fronthaul interface. With this scope, C-RAN favors the cooperation between co-located BBUs, enabling advanced coordinated signal processing. In doing so, user throughput and network performances [4] are enhanced while reducing energy and cost savings [5] thanks to the multiplexing gain of processing resources.

Despite the aforementioned benefits, BBUs cloudification directly impacts the provisioning approach in the fronthaul link. 5G fronthaul may require up to 157.3 Gb/s of throughput with 10 μs - 250 μs of latency [6]. Such stringent constraints would make the dark fiber almost the only applicable fronthaul solution which cost increases the CAPEX, thus counteracting the original cost saving principle of C-RAN.

In order to relax these excessive fronthaul constraints without losing the benefits from BBUs centralization, recent 5G contributors are proposing a hybrid C-RAN architecture that enables flexible BBU placement between the Cloud site and the access site. In a such approach, the BBUs are seen as a chain of virtual functions that can be splitted at many conceivable points, thus enabling a partial BBU centralization, while leaving some functions at the access site. Accordingly, new interfaces between BBU functions are being identified with reference to the traditional Long Term Evolution (LTE) architecture. These interfaces enable a set of functional splits [7]–[10], wherein some processing functions are kept in the access site, thus relaxing the bandwidth and latency requirements of the fronthaul link. However, functional split may still reduce the advantages from BBU centralization [5], while keeping some processing functions at the access site.

In this paper, we put forward a user-centric functional split solution that aims at optimizing the placement cost of BBU functions, while considering the Quality of Service (QoS) requirements of each user. By enabling the selection of BBU functional split for each type of traffic, data rates in the fronthaul link and computational requirements in each site become more tunable [10], which is a key to build cost effective RAN deployment solutions.

Our objective is to jointly minimize the fronthaul bandwidth and the computational resource consumption at both sites. This leads to a placement problem with contradictory goals. From one side, the more BBU functions are centralized, the more energy consumption and hence CAPEX and OPEX will be reduced. From the other side, high degree of centralization will increase the fronthaul traffic, thus resulting in higher connection costs. The main contributions of our paper can be summarized as follows:

- First, we design a cost efficient C-RAN architecture enabling on-demand deployment of RAN resources, while dealing with temporal load variation of users. This Agile C-RAN architecture, denoted by `AgilRAN` is multi-sited which is in compliance with the NGFI architecture [7]. `AgilRAN` is managed by a user centric split orchestration framework which performs BBU function placement and their interconnection taking into account real-time network state.
- Second, we model the user centric functional split problem as an Integer Linear Problem (ILP) which minimizes the network deployment cost in terms of computational and link resource usage. Our aim is to ensure the best trade-off between baseband function centralization and fronthaul network consumption.
- With high dense of user traffic, the resolution time becomes intractable. In order to operate in a polynomial time, we propose a heuristic based on Swarm Particle Optimization approach [12], denoted as `SPLIT-HPSO`. The algorithm consists of generating initially a set of potential solutions. Then, iteratively, the candidate solutions collaborate and evolve towards the best global solution. Our scheme is proved to be scalable running within four Transmission Time Interval (TTI) units which makes our solution operational. We expect that the optimization process is triggered periodically to optimize the deployment cost in a pro-active manner.
- Forth, we validate this proposal using an experimental C-RAN prototype, which makes use of Open Air Interface (OAI) [13], an open-source software implementation of LTE, spanning the full protocol stack of E-UTRAN and EPC [13]. Specifically, we evaluated quantitatively and qualitatively the bandwidth and computational consumption models for each type of splits in a first order. Next, we implement an orchestration architecture, which enables dynamic configuration of functional splits, according to the outputs of `SPLIT-HPSO`.

The remainder of this paper is organized as follows. In Section II, we provide an insight of the standardization activities on BBU functional splits and related works. In Section III, we describe the proposed `AgilRAN` architecture and the main components of our dynamic RAN Functional Split Orchestration solution. In Section IV, we describe our ILP formulation, which aims at minimizing jointly the bandwidth and computational resource allocation. In Section V, we detail the proposed heuristic `SPLIT-HPSO`, while a description of our simulation and experimental environments and major results are presented in Section VI. Finally, Section VII concludes the paper.

II. Related Work

A. Standardization Effort Towards a RAN Architecture With Functional Split

Third Generation Partnership Project (3GPP) has motivated the need for functional splits to build scalable cost effective RAN solutions. 3GPP has specified in [6] eight options for splitting the BBU stack, where high-level splits decentralize more functions for relaxing bandwidth utilization, while low-level splits maintain the centralization benefits at the expense of bandwidth utilization. Next Generation Mobile Networks (NGMN) alliance highlighted in [4] the need for a splitted fronthaul architecture and studied different split requirements in uplink and downlink. Next Generation Fronthaul Interface (NGFI) work group proposed in [7] an Ethernet-based Next Generation Fronthaul Interface architecture, consisting of Remote Radio Units (RRUs) and Radio Cloud Center (RCCs), connected through link aggregators called Radio Aggregation Units (RAUs). The RRU includes the antenna unit along with its computational resources, while the RCC includes the BBU pool. Moreover, Small Cell Forum (SCF) introduces in [8] a split between the RRC and PDCP layers offering the possibility to separate the control-plane and the data-plane. Based on these perspectives, it is foreseen that there is a strong interest to effectively opt for functional splits in next 5G RAN as telecom industry are pushing forward to leveraging the BBU split in providing a cost-effective transport network.

B. Resource Allocation Challenges in a RAN Architecture With Functional Split

Interestingly, many related works focus on analyzing the different split requirements and performances in Ethernet based fronthaul. In [10], authors evaluated quantitatively the computational cost of each BBU function and analyzed possible BBU splits for their bandwidth and latency requirements. In [14], authors evaluated the impact of different transport protocols on a splitted fronthaul, while in [15], authors analyzed the impact of different packetization strategies.

Authors in [16], propose a graph based framework to reduce the resource allocation computational cost in both access and Cloud sites. This framework takes into account both traffic load in the fronthaul link and the delay requirement for each cell, which are contradictory goals. To this end, a genetic algorithm is proposed, in order to place optimally the BBU functions across sites. However, this approach is based on the assumption that the resource allocation and delay requirements for each split are fixed and does not reflect the most basic properties of real RAN systems.

A network calculus approach is elaborated in [9] a multi objective function to minimize the deployment cost of BBU functions across multiple sites, while considering the strict delay requirements of critical services. Unfortunately, this work does not refer to a quantitative model for the computational requirement of each split.

In [17], authors propose a detailed Total Cost of Ownership (TCO) minimization model in a fiber based RAN with BBU splits, which takes into account quantitative models for computational and link resource requirements. However, this model still considers the use of splits with a coarse grained decision, as it generates a split per cell for all attached users.

Authors in [5], elaborate a teletraffic theory to analyze the gain of aggregating the fronthaul traffic of multiple cells with different split configurations. An objective function is elaborated for maximizing the energy and cost savings. To do that, the authors evaluate the allocated amount of computational and radio resources along with the generated data rate in the fronthaul. Interestingly, this work proves that the gain is function of the traffic profile and monitoring method. In other words, the user load and type of traffic deeply impacts the split gain. For this reason, we conclude that a fine-grained split approach per user basis will certainly achieve higher benefits. However, this approach is not evaluated in [5].

Differently from above works, [18] proposes a model with user split orchestration that aims at minimizing the system energy and bandwidth consumption in the fronthaul link. The elaborated model is based on quantitative models to calculate the computational and link requirements for each split. However, this work relies on unrealistic split model as the platform control function which includes the MAC scheduler is assumed to be a user centric processing which is not accurate. In this paper, we stick to the adopted analytical models [8] and [10] and we elaborate a practical scheme based on realistic properties of each baseband processing function. Moreover, authors in [18] assume that the radio resource allocation is fixed for each user. Whereas, in our work, we take into consideration the traffic load variation and analyze its impact during the split decision which is the key for an efficient user-centric approach. In [19], the same authors elaborate an end-to-end delay model to analyze the impact of a user delay request on split decision, which impacts the total cost and energy consumption. The model is evaluated for a single user having optimal network conditions, therefore, the split decision is still considered as per cell basis.

C. Summary of Related Work

As a summary, the aforementioned methods [5], [10], [16], [17] propose different approaches and models for minimizing the energy and deployment cost of C-RAN, while supporting the BBU splitting. However, the split decision is mainly taken with coarse grained on cell basis. As for [18], even the authors formulate their approach on a user basis, the elaborated split model seems unpractical with unrealistic assumptions. In [19], unfortunately there is only one user considered. Consequently, this would be seen as a monolithic approach, unlike our proposal in which we opt for a user-centric functional split approach based on analytical models and a practical scheme reflecting the RAN real properties.

In this paper, we further quantify the user split gain as function of the traffic load with reference to a quantitative model and compare it to the baseline cell splits. To do so, we propose a novel user-centric functional split orchestration approach for C-RAN (`SPLIT-HPSO`), which is able to manage and operate a RAN based on user functional split. We refer to the latest advances of Software Defined Network (SDN) to monitor and control the C-RAN network state and we consider the ETSI NFVI standard [20] for orchestrating the BBU functional splits on-demand. Then, we elaborate an heuristic based algorithm to evaluate our approach in large scale high density of users. In addition, we validate the proposed solution within our experimental C-RAN prototype, which makes use of OAI [13] and FlexRAN Controller [21].

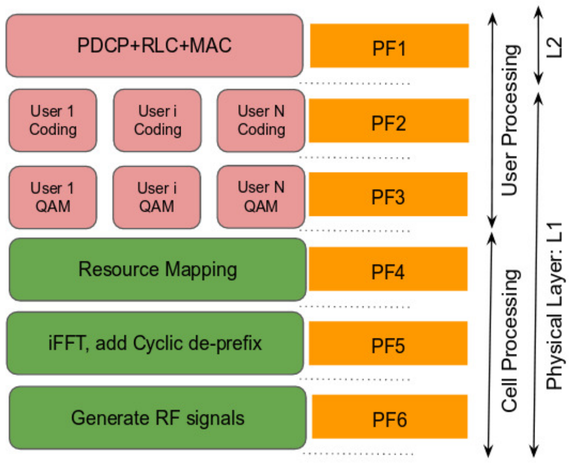


Fig. 1. User-centric functional split.

III. Agile C-RAN Architecture for a User-Centric Functional Split

In this section, we describe our proposed Agile multi-sited RAN architecture, denoted as `AgilRAN`, introduced in our previous work [11]. Note that similar C-RAN architectures have been proposed in the literature, such as in [22]. The latter has been designed by mobile operators to respond to their 5G vision for building a virtualized RAN on open hardware, with embedded Artificial Intelligence powered radio control to enable SDN-like based capabilities [23]–[27]. However, in our proposed approach, we go a step further and adopt a fine grained resource provisioning approach to ensure a user level orchestration of RAN network functions.

A. Baseband Processing Decomposition Model

Baseband processing functions have different properties depending on the processed data and the layer hosting them. As a matter of fact, some Processing Functions (PF) perform at a user level, denoted by User-centric Processing Functions (UPF), i.e., once executed, they deal with one user at a time. Other PFs operate on a cell level, described as Cell-centric Processing Functions (CPF). For the sake of simplicity, we assume a DownLink (DL) processing. Hereafter, we detail all the PFs depicted in Fig. 1.

- PF_1 is a CPF that conducts the platform control processing with Packet Data Convergence Protocol (PDCP), Radio Link Control (RLC) and Medium Access Control (MAC). It is worth noting that, PDCP and RLC functions are UPFs. In contrast, DL scheduler belonging to the MAC layer is rather a CPF since it needs to maintain the cell's state for scheduling.

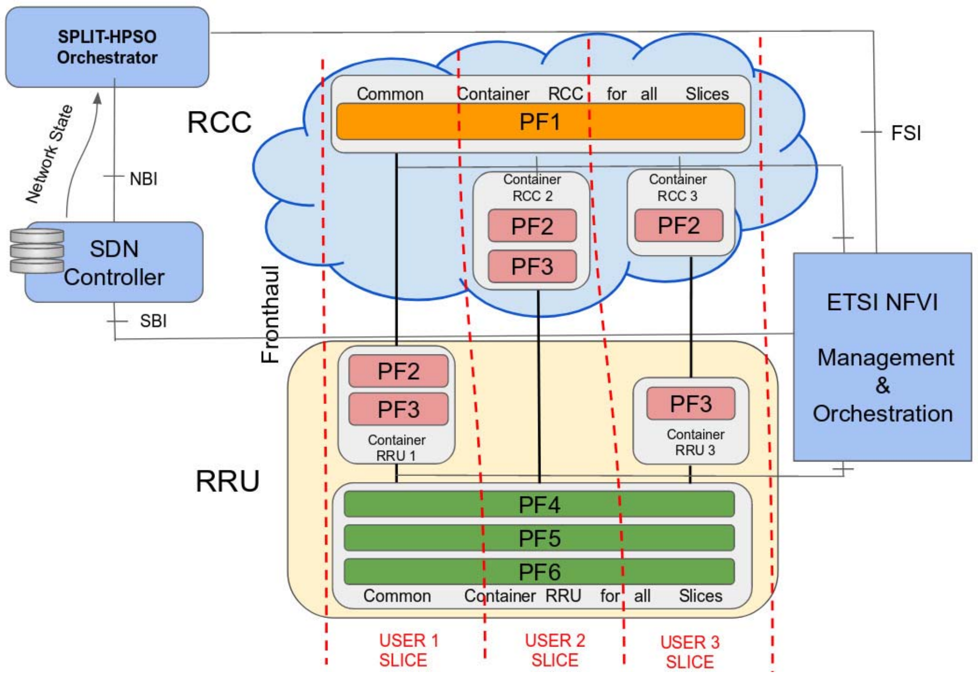


Fig. 2. `AgilRAN` architecture.

- PF_2 is an UPF that performs mainly the encoding.
- PF_3 is an UPF that includes Quadrature Amplitude Modulation (QAM).
- PF_4 corresponds to the resource mapping function. This CPF starts the cell processing by mapping the symbols on radio resource elements.
- PF_5 is a CPF that adds the cyclic prefix and transforms symbols from frequency domain to time domain using iFFT Algorithm.
- PF_6 is a CPF that generates the RF signals and conducts the parallel-to-serial (P/S) conversion among others.

B. `AgilRAN` Architecture

In what follows, we provide an overview of our RAN architecture. `AgilRAN` adopts a “highly disaggregated” RAN model where the BBU is re-architected to be decomposed into network functions. The latter are instantiated into containers to perform either cell or user processing tasks. In doing so, BBU microservices can easily interact with each other and scale separately, which make them Cloud native.

As illustrated in Fig. 2, `AgilRAN` is characterized by two layers in which processing RAN functions can be instantiated. The lower layer is referred to the Remote Radio Units (RRUs), where a number of data units is serving a set of antennas. The upper layer is a Radio Cloud Center (RCC) where network functions of multiple cells can be aggregated for a centralized processing. Note that in Fig. 2, only one RRU data unit is illustrated corresponding to a server, which could be whether Commercial Off-The-Shelf (COTS) or equipped with accelerators, e.g., FPGA (Field Programmable Gate Array).

We rely on Network Function Virtualization (NFV), and more specifically, on the container technology (e.g., LinuX Container LXC and Docker [28]) to enable the virtualization of fine-grained RAN network functions in both RRU and RCC. It is worth noting that a container-based virtual environment guarantees higher performances compared to virtual-machines based environments, as they run directly on the kernel, use less memory and make run-time execution more efficient. Being packaged in containers instead of virtual machines, PFs can be dynamically instantiated and destroyed within few microseconds. Indeed, according to our experiments, we have quantified the average deployment time of a container-based PF to 1.8 μs $\pm$ 0.2 μs.

On top of our `AgilRAN` architecture, we build our `SPLIT-HPSO` orchestrator module that can dynamically deploy a pool of docker containers both on RRU and RCC while ensuring a high scalability level. The latter is responsible for the placement of baseband network functions and their interconnection taking into account the real-time network state (e.g., system load, radio conditions, resource capacity, etc.). We propose to perform our optimization process in proactive manner and to trigger it periodically after a predefined time period T. The latter parameter reflects the balance point between the optimization process cost and the performance of the global system. Note that the impact of the time interval T on the system is out of the scope of this work.

As shown in Fig. 2, and by focusing on one execution of the optimization process, `SPLIT-HPSO` communicates the optimal splitting decision per each user via the Functional Split Interface (FSI). Afterworlds, the ETSI NFV orchestrator is responsible for the decision’s execution by creating the according user slice. Once the user leaves the cell, the orchestrator triggers the destruction of containers and deletes the slice. It is straightforward to see that such a user-centric approach offers a high level agility and enhances considerably the resource usage compared to a cell-centric approach. Technical details about our architecture and a video of our user-centric functional split demonstration are available here [29].

IV. Problem Formulation

In this section, we formulate the user-centric functional split orchestration problem. First, we describe the computation model of each processing function in Section III-A. Then, we detail the split possibilities and their corresponding bandwidth requirements. Afterwards, we describe the power consumption model characterizing both RRU and RCC sites. Finally, we detail the problem formalization based on an ILP model.

A. Functional Split Model

In order to quantitatively study the computational requirement for each split, we refer to the conducted analysis in [10] expressing the amount of computational resources, g_i, for each PF_i in Giga Operations Per Second (GOPS) as follows:

$$PF_1 : g_1 = G_1^{ref}.\frac{A}{A_{ref}}$$
$$PF_2 : g_2(M_i, L_i) = G_2^{ref}.\frac{B}{B_{ref}}.\frac{M_i}{M_{ref}}.\frac{A}{A_{ref}}.\frac{L_i}{L_{ref}}$$
$$PF_3 : g_3(L_i) = G_3^{ref}.\frac{B}{B_{ref}}.(\frac{A}{A_{ref}})^2.\frac{L_i}{L_{ref}}$$
$$PF_4 : g_4(\{L_i\}) = G_4^{ref}.\frac{B}{B_{ref}}.\frac{A}{A_{ref}}.\sum_{i}^{Users}\frac{L_i}{L_{ref}}$$
$$PF_5 : g_5 = G_5^{ref}.\frac{B}{B_{ref}}.\frac{A}{A_{ref}}$$
$$PF_6 : g_6 = G_6^{ref}.\frac{B}{B_{ref}}.\frac{A}{A_{ref}}$$

It is worth noting that G_i^{ref} refers to the PF_i's GOPS value in the reference scenario. This constant is multiplied by scaling parameters which includes the carrier bandwidth (B), number of antennas (A), user traffic load (L) and modulation (M). As mentioned earlier, PF_5 and PF_6 are cell-centric for time-domain along with PF_1 that corresponds to a platform control processing. Hence, their computational requirement is load independent. In contrast, PF_2, PF_3 and PF_4 are processing in frequency domain so they take into account only frequency carriers having data signals, which make them load dependent.

As for the bandwidth requirement model for each type of split, we refer to [8] that quantitatively approach the estimated bandwidth on the fronthaul link as follows:

$$Split_0 : f_0(\{L_i\}) = \alpha_0.\sum_{i}^{Users} L_i$$
$$Split_1 : f_1(L_i) = \alpha_1.L_i$$
$$Split_2 : f_2(M_i, L_i) = \alpha_2(M_i).n_{RB}.L_i + \beta_1$$
$$Split_3 : f_3(L_i) = \alpha_3.A\ .n_{RB}.L_i + \beta_2.A$$
$$Split_4 : f_4 = \alpha_4\ .A.n_{RB}$$
$$Split_5 : f_5 = \alpha_5.A\ .n_s$$
$$Split_6 : f_6 = \alpha_6.A\ .n_s$$

where the coefficients α_i and β_i are constants for the model with reference to [8]. n_{RB} corresponds to the total number of Resource Blocks (RBs) and n_s refers to the sampling rate. We recall that $Split_0$ refers to the deployment scheme, where all PFs are fully decentralized at RRU. In contrast, $Split_6$ corresponds to the conventional C-RAN, where all PFs are fully centralized in RCC. $Split_1$ corresponds to the placement of PF_1 at RCC, while keeping PF_i, $2 \leq i \leq 6$ at RRU. When $Split_2$ is triggered, only PF_1 and PF_2 are placed in the Cloud. $Split_3$ and $Split_4$ refer to the PF_3-PF_4 and PF_4-PF_5 splits, respectively. Finally, $Split_5$ corresponds to the instantiation of PF_6 at RRU, while moving PF_i, $1 \leq i \leq 5$ to RCC. We recall that $Split_4$, $Split_5$ and $Split_6$ are generated from cell-centric processing functions. The latter form a sequence of functions in the physical layer, where user signals are multiplexed, generating a constant bit rate in the fronthaul link.

B. Power Consumption Model

Based on the reference model presented in [10], the calculated BBU complexity in GOPS can be multiplied by a technology-dependent factor P_f expressing the number of operations that can be performed per second and per Watt (W). This factor is equal to 40 GOPS/W for the reference case and default technology, i.e., 65 nm for General Purpose CMOS (Complementary Metal-Oxide-Semiconductor). Intuitively, we would place the processing functions in sites with less power cost.textbf To do that, we characterize a deployment site by an energy efficient indicator that expresses its Power Usefulness Effectiveness value (PUE).

By denoting Pow_{in} as the input power for a given site and Pow_{out} as the output power after server processing, PUE is expressed as follows, $PUE = Pow_{in}/Pow_{out}$. Consequently, the smaller is the PUE values, the lower is the power consumption of IT resources, and hence the better is the site power.

C. User-Centric Functional Split Problem Formulation

We consider a set of N users connected to one cell in the `AgilRAN` infrastructure. Each user generates a load L_i, which corresponds to an amount of RBs allocated in DL. We assume a set of K splits, as explained in Section III-A. We model the RRU as a set of data units characterized by a computational capacity of C_{RRU} GOPS, maximum of power consumption P_A and Power Usefulness Effectiveness PUE_U. We recall that each data unit is dedicated to one cell. The RRU is connected to RCC, which is characterized by computational capacity of C_{RCC} GOPS, maximum of power consumption P_E and Power Usefulness Effectiveness PUE_C. The two sites are connected via a fronthaul link of capacity B Mbps. The amount of GOPS consumed at RRU (respectively RCC) for the split k of user $_i$ is denoted by A_i^k (respectively E_i^k). Besides, R_i^k corresponds to the fronthaul bandwidth generated by the split k of user i.

Our aim is to minimize jointly i) the overall cost of RAN function placement defined as the sum of computational cost across sites and ii) the bandwidth consumption on fronthaul across virtual user slices, which are contradictory objectives. To do so, a binary matrix X of optimal splits is generated where $x_i^k = 1$ when split k is selected for user i and 0 otherwise. In some cases, when a cell split is activated then all users should be affected to this split. Hence, we need to also model a set of cell splits K_{cell} as a subset of the total split options K. We define $y_j\ \forall j \in \{0,..,K_{cell}\}$ as a binary variable that takes the value 1 when a cell split is activated and 0 otherwise.

We model the problem of RAN function placement, while ensuring a trade off between computational and bandwidth consumption cost as an optimization problem. The latter is formulated as an ILP detailed hereafter:

$$\mathcal{LP}_0\text{: Minimize: } \alpha.PUE_U.\frac{P_{RRU}}{P_U} + \beta.PUE_C\frac{P_{RCC}}{P_C} + \gamma.\frac{FH_{Rate}}{B}$$

$$\text{subject to: } \sum_{k=0}^{K} x_i^k = 1 \quad \forall i \in N \tag{1}$$

$$\sum_{j=0}^{K_{cell}} y_j \leq 1 \tag{2}$$

$$\sum_{i=1}^{N} x_i^j = N y_j \quad \forall j \in K_{cell} \tag{3}$$

$$\sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k R_i^k \leq B \tag{4}$$

$$\sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k A_i^k \leq C_{RRU} \tag{5}$$

$$\sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k E_i^k \leq C_{RCC} \tag{6}$$

$$x_i^k \in \{0,1\} \quad \forall i \in N\ \forall k \in K \tag{7}$$

$$y_j \in \{0,1\} \quad \forall j \in K_{cell} \tag{8}$$

$$\text{where:} P_{RRU} = (1/P_f).\sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k A_i^k \tag{9}$$

$$P_{RCC} = (1/P_f).\sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k E_i^k \tag{10}$$

$$FH_{Rate} = \sum_{i=1}^{N}\sum_{k=0}^{K} x_i^k R_i^k \tag{11}$$

Constraint (1) expresses that only one split can be selected for each UE$_i$. (2) denotes that at most one cell split can be possibly chosen. In (3), means all attached users should deploy the activated cell split. (4), (5) and (6) express the upper bound limit for the total generated rate in the fronthaul link, the RRU and RCC computational resource requirements, respectively. Afterwards, we calculate the total amount of consumed power in RRU and RCC, P_{RRU} and P_{RCC} respectively. Indeed, the total resource computational demand is divided by the Power factor P_f as shown in (9) and (10). The total generated rate on the fronthaul link is expressed as FH_{Rate} in (11).

The objective function aims to find the trade off between the centralization level weighted by β and the RCC PUE factor PUE_C and between the decentralization level weighted by α and the RRU PUE factor PUE_U. It is worth noting that we take into account the traffic load on the fronthaul by calibrating the weighting factor γ. The latter can affect in its turn the computational and power requirement in both RRU and RCC. This is a contradictory goal that is optimized if we find the appropriate set of user splits.

V. Proposal: SPLIT-HPSO Algorithm

A. Particle Swarm Optimization Algorithm

In this section, we solve the optimization problem of user-centric functional split formulated in the previous Section IV using Particle Swarm Optimization Algorithm [12]. Indeed, our formulated problem is ILP, so it is nondeterministic polynomial hard at high scale number of users if the aim is to solve it directly with a general-purpose ILP solver [30]. Thus, we need to design an algorithm to solve it in a polynomial time by generating a near-optimal solution.

In this context, we make use of to Particle Swarm Optimization Algorithm, which is a population-based stochastic approach. More specifically, once a set of random initial solutions are generated, there is a need of collaboration between them in order to share internal information and optimize the common objective function.

B. Particle Design

We propose hereafter an adaptive approach of the Particle Swarm Optimization Algorithm to solve our problem. The particles are candidate solutions that collaborate and evolve along the algorithm iterations in order to find the best solution optimizing the total deployment cost expressed in $\mathcal{LP}_0$. Each particle p; $p \in \{1,..,P\}$ is characterized by a position $\mathcal{SPLIT}_p$, a velocity $\mathcal{VL}_p$ and the local best visited position $\mathcal{SPLIT}_{Lbest,p}$. The first component (position) presents the candidate solution configuration. The second one (velocity) is the change vector that allows the particle evolve to the next position. The third component (best local position) is to memorize the local best solution configuration made so far, which is evaluated by its Cost, $\mathcal{C}(\mathcal{SPLIT}_{Lbest,p})$, with reference to the objective function expressed in $\mathcal{LP}_0$. We define $\mathcal{SPLIT}_{Gbest}$ as the best solution configuration among all the best local solutions of particles $\mathcal{SPLIT}_{Lbest,p}$; $\forall\ p \in \{1,..,P\}$.

When the algorithm is performed, each particle p iteratively collaborates with the others in order to define its new velocity component $\mathcal{VL}_p$. This process is formulated in equation (E1) where the new velocity $\mathcal{VL}_{new,p}$ is constructed based on the old velocity $\mathcal{VL}_{old,p}$ of previous iteration, $\mathcal{SPLIT}_p$, $\mathcal{SPLIT}_{Lbest,p}$ and $\mathcal{SPLIT}_{Gbest}$. In equation (E1), u_1 and u_2 are coefficients to improve the random nature of the evolution process. This is essential to insure investigating all the search space before converging to the near-optimal configuration. Once the new velocity is determined, the new position $\mathcal{SPLIT}_p$ is updated according to equation (E2).

$$\mathcal{VL}_{new,p} = \mathcal{VL}_{old,p} \cap [u_1 \otimes (\mathcal{SPLIT}_{Lbest,p} \ominus \mathcal{SPLIT}_p) + u_2 \otimes (\mathcal{SPLIT}_{Gbest} \ominus \mathcal{SPLIT}_p)] \tag{E1}$$

$$\mathcal{SPLIT}_p = \mathcal{SPLIT}_p \oplus \mathcal{VL}_{new,p} \tag{E2}$$

Algorithm 1 SPLIT-HPSO

```
1 Inputs: Users MCS with proportions of allocated RBs
2 Output: $\mathcal{S}_{opt}$ set of optimal user splits
3 Begin:
    for $k$ in $K_{cell}$ do
      if $\mathcal{C}(k) < \mathcal{C}(\mathcal{SPLIT}_{Gbest})$ then
        $\mathcal{SPLIT}_{Gbest} \leftarrow$ k
      end if
    end for
    for $p = 1$ to $P$ do
      for $i = 1$ to $N$ do
        $\mathcal{SPLIT}_{p,i} \leftarrow$ k $\in$ random($K_{user}$)
        $\mathcal{VL}_{p,i} \leftarrow$ random($[-|K_{user}|, |K_{user}|]$)
      end for
    end for
  repeat
     for $p = 1$ to $P$ do
       if $\mathcal{C}(\mathcal{SPLIT}_{p}) < \mathcal{C}(\mathcal{SPLIT}_{Lbest,p})$ then
         $\mathcal{SPLIT}_{Lbest,p} \leftarrow \mathcal{SPLIT}_{p}$
       end if
       if $\mathcal{C}(\mathcal{SPLIT}_{Lbest,p}) < \mathcal{C}(\mathcal{SPLIT}_{Gbest})$ then
         $\mathcal{SPLIT}_{Gbest} \leftarrow \mathcal{SPLIT}_{Lbest,p}$
       end if
     end for
     for $p = 1$ to $P$ do
       Update the velocity $\mathcal{VL}_{p}$ according to Algorithm 2
       for $i = 1$ to $N$ do
         $\mathcal{SPLIT}_{p,i} \leftarrow \mathcal{SPLIT}_{p,i} + \mathcal{VL}_{p,i}$
       end for
     end for
  until $iter$ = $ITER_{MAX}$;
  $\mathcal{S}_{opt} \leftarrow \mathcal{SPLIT}_{Gbest}$ is the optimal solution
```

Algorithm 2 Velocity Update

```
    Output: $\mathcal{VL}_{new,p}$ of particle p
    Begin:
    Generate $u_1$ and $u_2$ with $u_1 + u_2 < 1$
    for $i = 1$ to $N$ do
      if $u_1 > u_2$ then
        $\mathcal{VL}_{new,p,i} \leftarrow \mathcal{SPLIT}_{Lbest,p,i} - \mathcal{SPLIT}_{p,i}$
      else
        $\mathcal{VL}_{new,p,i} \leftarrow \mathcal{SPLIT}_{Gbest,i} - \mathcal{SPLIT}_{p,i}$
      end if
    end for
    Sort users descending with respect to user load.
    for $i = 1$ to $N$ do
      if ($\mathcal{VL}_{old,p,i} \neq \mathcal{VL}_{new,p,i}$) then
        Calculate $C_1 = \mathcal{C}(\mathcal{SPLIT}_{p,i} + \mathcal{VL}_{new,p,i})$
        Calculate $C_2 = \mathcal{C}(\mathcal{SPLIT}_{p,i} + \mathcal{VL}_{old,p,i})$
        if ($C_2 < C_1$) then
          $\mathcal{VL}_{new,p,i} \leftarrow \mathcal{VL}_{old,p,i}$
        end if
      end if
    end for
```

Considering the inherent characteristics of our problem, we divide the set of splits K into a subset of cell splits K_{cell} and a subset of user splits K_{user}. That is, $K = K_{cell} \cup K_{user}$ and $K_{cell} \cap K_{user} = \emptyset$. Specifically, in our proposed algorithm, a particle is designed as a matrix of $[N * K_{user}]$. Initially, each user is affected a random user split k in K_{user}, meaning that for particle p and user i, $\mathcal{SPLIT}_{p,i,k} = 1$ and $\mathcal{SPLIT}_{p,i,k'} = 0$; $\forall\ k' \neq k$. The velocity of each particle is a vector of $[N]$ that calculates for each user, the number of transitions to meet the new user split. Assuming that a user is affected a user split $k = 3$. If the velocity component in particle p for user i is $\mathcal{VL}_{p,i} = -2$, then the new user split should be $\mathcal{SPLIT}_{p,i} = k + \mathcal{VL}_{p,i} = 1$. Based on this, we define the velocity space as $[-|K_{user}|, |K_{user}|]$ to limit the allowed split transitions.

C. Functional Split Orchestration Based on Hybrid Particle Swarm Optimization SPLIT-HPSO

Our proposed algorithm works as follows. We first evaluate the cost $\mathcal{C}(k)$ of each cell split $k \in K_{cell}$ and update the global best solution $\mathcal{SPLIT}_{Gbest}$ by choosing the cell split k with the lowest cost. In a second stage, we search for the best solution configuration among all possible user splits k in K_{user}. The final best solution is either a cell split from K_{cell} or a combination of user splits from K_{user}. The second stage is described as follows. Initially, each user i in particle p is assigned a random user split k and a velocity value $\mathcal{VL}_{p,i}$. Then, iteratively,

- Each particle p updates its local best solution $\mathcal{SPLITT}_{Lbest,p}$.
- The global best solution is updated accordingly.
- Each particle p updates its velocity $\mathcal{VL}_p$ according to Algorithm 2.
- Each particle p updates its new solution configuration $\mathcal{SPLIT}_p$ by considering the new calculated velocity.

The major steps of our proposed algorithm are described in Algorithm 1, where the procedure for implementing SPLIT-HPSO is giving.

D. Velocity Update Strategy

The velocity update for each particle in Algorithm 1 is roughly described and we propose to detail it in Algorithm 2. The velocity update is a complex step, where two phases are integrated: $exploitation$ and $exploration$. The first phase is expressed in ($\mathcal{SPLIT}_{Lbest,p} \ominus \mathcal{SPLIT}_p$). Thanks to the latter, we determine the velocity update vector to move from current configuration $\mathcal{SPLIT}_p$ to the best local configuration $\mathcal{SPLIT}_{Lbest,p}$. The second phase is expressed in ($\mathcal{SPLIT}_{Gbest} \ominus \mathcal{SPLIT}_p$). Similarly, we determine the velocity update vector to move from current configuration $\mathcal{SPLIT}_p$ to the best global configuration $\mathcal{SPLIT}_{Gbest}$. Note that these two phases are weighted by random values u_1 and u_2. Hence, one particle can move, in each iteration, towards its best local position with a probability u_1 or towards its global position with a probability u_2. The aim here is to not fall into a local optima. In Algorithm 2, we propose a heuristic based velocity update that embeds a local optimizer to expedite the convergence. For each user, we evaluate the gain from keeping the actual user split and the gain from moving to the user split of the best particle. More specifically, if the cost of actual user split is lower than the cost proposed by the best solution, then such a split is kept. The new velocity component selects, for each user, either the actual split configuration or the user split of best particle, favoring the lowest deployment cost.

At the end, each candidate solution is evolved towards finding the best global configuration by applying the SPLIT-HPSO heuristic. In each iteration, we build a new and feasible configuration in a way that constraints are satisfied in each step. The runtime of the proposed approach is computed as $\mathcal{O}(P\ N^2\ ITER_{MAX})$. Such an approach can further optimize the best solution and hence, fasten the algorithm convergence as will be shown in the simulation results.

VI. Performance Evaluation

In this section, we evaluate the performance of our SPLIT-HPSO proposal making use of both system-level simulations and an experimental platform based on OAI [13]. In the following, we first define the performance metrics as well as the baselines used for performance comparison. Second, we detail the simulation environment and the generated results. To evaluate the efficiency of SPLIT-HPSO, we compare it with most prominent strategies in C-RAN. Third, we describe our experimental platform and provide the details of our emulation environment. Finally, we describe the results of the experimental prototype.

A. Simulation Performances

1) Simulation Baselines and Performance Metrics: To assess the effectiveness of our approach while increasing the number of end users, we compare it with our proposal in [11], denoted by optimal-split, and different cell-centric configurations. It is worth noting that our proposal in [11] performs an exhaustive research to reach the optimal solution. To achieve its objective, it makes use of Branch-and-Cut algorithm after relaxing the integer variables of our ILP problem. Besides, 7 cell-centric configurations are considered for our performance evaluation: Distributed-RAN (D-RAN) corresponding to $Split_0$, Cloud-RAN (C-RAN) corresponding to $Split_6$, in addition to $Split_1$, $Split_2$, $Split_3$, $Split_4$ and $Split_5$. Note that these related strategies are detailed in Section III.

Hereafter, we define the metrics used to gauge the performance of our proposal in the simulation environment.

- $\mathbb{C}$ corresponds to the total deployment Cost as defined in the objective function in Section IV: $\mathbb{C} = \alpha.PUE_U.\frac{P_{RRU}}{P_U} + \beta.PUE_C\frac{P_{RCC}}{P_C} + \gamma.\frac{FH_{Rate}}{B}$.
- $\mathbb{P}$ corresponds to the percentage of deployed user splits and quantifies the rate of each type of split.
- $\mathbb{F}$ refers the total fronthaul throughput measured in Mbps.
- $\mathbb{T}$ measures the average computation time in milliseconds (ms) to solve one instance of the user-centric problem.

2) Simulation Environment: We designed and implemented a Java-based discrete event simulator to evaluate SPLIT-HPSO performances, while varying the number of connected UEs. Besides, we integrated SPLIT-HPSO and the related splitting strategies: $Split_0$, $Split_1$, $Split_2$, $Split_3$, $Split_4$, $Split_5$ and $Split_6$. We compared the aforementioned strategies with respect to the defined performance metrics. Similarly to [10] and [31], we consider the following benchmark scenario: We consider one RRU of capacity C_{RRU} equals to 1060 GOPS and a PUE_U equals to 2.3. RCC has a capacity C_{RCC} equals to 1060 GOPS and a PUE_C equals to 1.5. Both RRU and RCC are connected via a fronthaul link of 1228.8 Mbps corresponding to the highest required bandwidth [8]. The network configuration is detailed in Table I.

TABLE I
Simulation Parameters

Parameters	Values
G_1^{ref}, G_2^{ref},G_3^{ref},G_4^{ref}	200 GOPS, 20 GOPS, 10 GOPS
G_4^{ref}, G_5^{ref}, G_6^{ref}	30 GOPS, 80 GOPS, 720 GOPS
A_{ref}, A	1 antenna
Ref. Bandwidth (B_{ref})	20 MHz
Bandwidth (B)	20 MHz
Total number of RBs, n_{RB}	100 for data (PDSCH)
L_{ref}	1 (Full load)
(L)	variable in [0;1]
Transport blocks	1 TBS per sub-frame
Sampling rate	30.72 MHz
Headers per IP packet	PDCP(2 bytes), RLC(5 bytes), MAC(2 bytes)
DL FAPI overhead per UE	1.5 Mbps
Number of Res for PCFICH	$PCFICH_{REs}$ = 16
PHICH group	$PHICH_{REs}$ = 12
Aggregation level 4	$PDCCH_{REs}$ = 144

TABLE II
Modulation Order

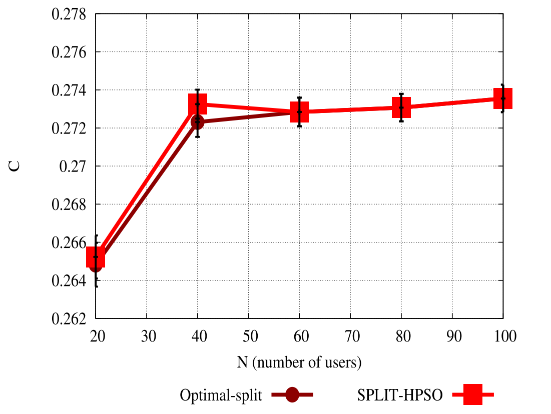
1-6	QPSK	2	0-9
7-9	16-QAM	4	10-16
10-15	64-QAM	6	17-28

Furthermore, we consider that N static UEs, varying in the range of $[20; 100]$, are randomly placed within the coverage of one cell. For each UE, we calculate the MCS index I_{MCS}, the modulation order Q_m of UEs as stated in Table II. During each algorithm execution, each UE generates a service demand, which consists of a) video traffic with a throughput varying in the range of $[2; 13]$ Mbps and b) web traffic varying in the range of $[0.032; 0.064]$ Mbps according to [32]. Note that UEs asking video traffic are considered as "high loaded" UEs. They are affected a higher amount of radio resources and their proportion may vary between 3% and 15%. In the same way, we define as "low loaded" UEs those asking for web traffic. We make use of a proportional fair scheduler to compute the number of RBs to be allocated for each UE in Downlink traffic. Moreover, we assume that the proportion of UEs asking for a video traffic R inside the cell may vary between 0% and 100%.

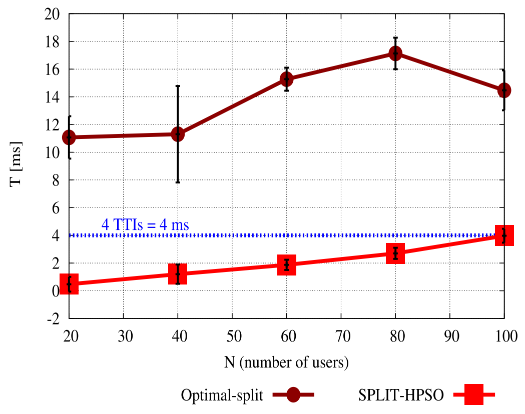
SPLIT-HPSO parameters are set to $ITER_{MAX}$ equals to 15 iterations and a population P of 10 particles. The coefficients u_1 and u_2 are uniformly distributed in $U(0, 0.2)$ and $U(0, 0.8)$, respectively. Note that we plot the average of 30 simulations with the confidence level set to 95%. Tiny confidence intervals are not shown in the following figures.

3) Simulation Results:

a) Convergence analysis: In what follows, we vary N in $[20; 100]$ with a rate of UEs generating a video traffic $R = 50\%$ in each iteration. We set the weights α, β and γ to the values 0.1, 0.1 and 0.8 respectively, which corresponds to a high deployment unit cost in the fronthaul link. We aim to

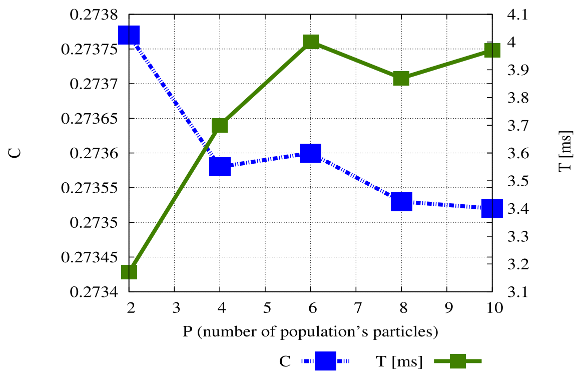


(a) Deployment Cost $\mathbb{C}$ (P = 10, $ITER_{MAX}$ = 15)

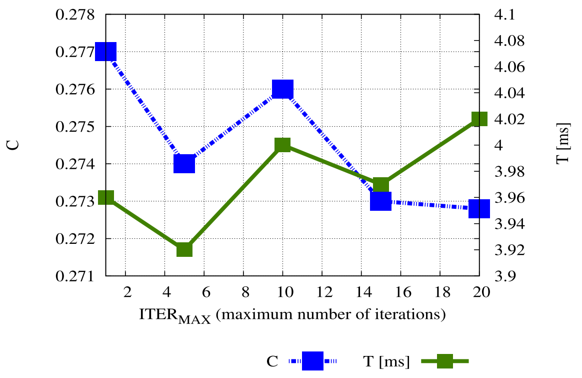


(b) Computation Time $\mathbb{T}$ (P = 10, $ITER_{MAX}$ = 15)

Fig. 3. SPLIT-HPSO convergence analysis.



(a) N = 100, $ITER_{MAX}$ = 15



(b) N = 100, P = 10

Fig. 4. Trade off between the deployment cost $\mathbb{C}$ and computation time $\mathbb{T}$ for SPLIT-HPSO.

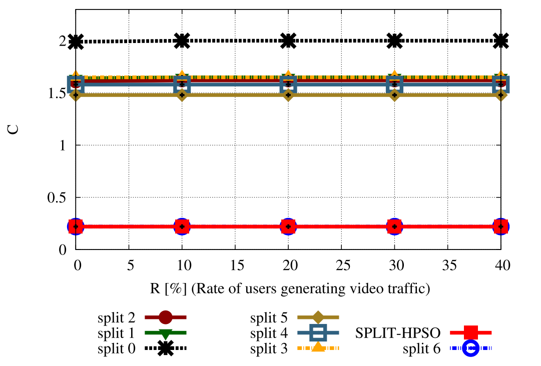
evaluate the performance of SPLIT-HPSO in case of high density of UEs.

In Fig. 3.(a), we compare SPLIT-HPSO to our proposal, optimal-split [11], which converges to the optimal solution. It is straightforward to see that our solution generates near optimal solutions when the number of UEs N is lower that 60. Within this range, the optimal solution provides a user split configuration that ensures a total cost, $\mathbb{C}$, lower than our proposed approach. Whereas, when N is higher than 60, our proposed approach achieves the same cost deployment of the optimal solution.

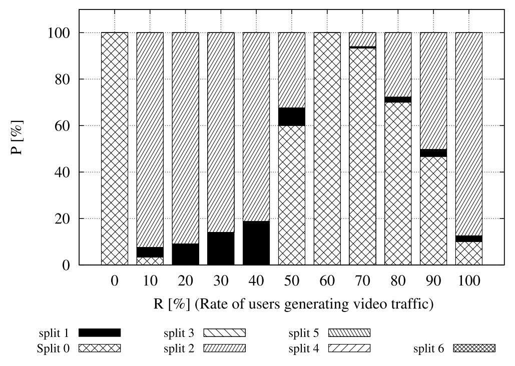
With regards to scalability, Fig. 3.(b) illustrates the resolution time $\mathbb{T}$ of the different strategies versus the number of UEs N. Note that the Transmission Time Interval (TTI) in C-RAN is equal to 1 millisecond according to [33]. It is straightforward to see that the non-scalable optimal solution takes a significantly longer time than SPLIT-HPSO to solve one instance of the optimization problem. Indeed, the optimal solution struggles to scale, as it takes values in $[14; 20]$ milliseconds to solve instances of N higher than 60 UEs. In contrast, SPLIT-HPSO can easily solve any size of instance (i.e., N in $[20, 100]$) in the range of $[0; 4]$ milliseconds. Eventually, by speeding up the computation time up to 5 orders of magnitude than Optimal-Split, SPLIT-HPSO is able to take an up-to-date decision and execute it after 4 TTI period. Unfortunately, Optimal-Split is not able to do so since its decision, once taken, will be already obsolete and hence not applicable.

Fig. 4 evaluates the impact of the number of particles P and the number of iterations $ITER_{MAX}$ on the solution quality (i.e., the deployment cost). However, we should also take into account the resolution time. Indeed, our aim is to find the trade-off between the deployment cost and the resolution time. Fig. 4.(a) assesses the efficiency of SPLIT-HPSO while varying the number of particles P. Indeed, for a fixed number of UEs (i.e., $N = 100$) and fixed number of iterations (i.e., $ITER_{MAX} = 15$), we can observe that the deployment cost $\mathbb{C}$ decreases when P increases. This proves that the size of P impacts the quality of the solution. However, it is straightforward to see that while increasing the number of particles, the computation time $\mathbb{T}$ increases. It is clear to see that when the number of particles P is higher than 8, the deployment cost $\mathbb{C}$ becomes stable. Such a behavior is predictable, as the solution quality is enhanced as soon as the number of particles is increased, which in turn, requires more computation time to solve the problem.

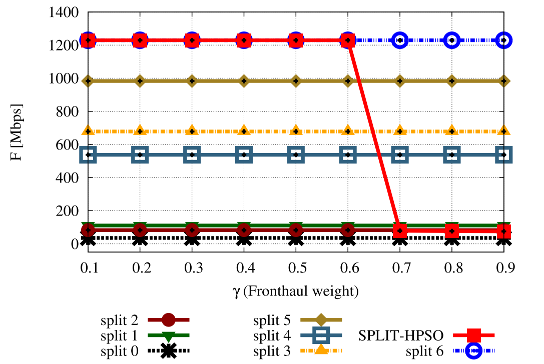
Fig. 4.(b) assesses the convergence behavior of SPLIT-HPSO while varying the number of iterations



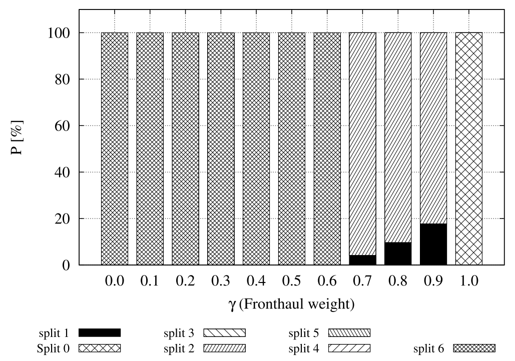
(a) Deployment Cost $\mathbb{C}$ (α = 0.8, β = 0.1 , γ = 0.1)



(b) Percentage of splits $\mathbb{P}$ [%] (α = 0.1, β = 0.1 , γ = 0.8)



(c) Fronthaul Throughput $\mathbb{F}$ [Mbps] (β = 0.1, R = 50%)



(d) Percentage of splits $\mathbb{P}$ [%] (β = 0.1, R = 50%)

Fig. 5. SPLIT-HPSO performance evaluation (N = 50).

$ITER_{MAX}$. Indeed, for a fixed number of UEs (i.e., $N = 100$) and a fixed number of particles (i.e., $P = 10$), we can observe that the deployment cost $\mathbb{C}$ decreases when $ITER_{MAX}$ increases. This proves that the solution quality is iteratively enhanced, however, impacting the increase of the computation time $\mathbb{T}$. It is interesting to see that starting from $ITER_{MAX} = 15$, the deployment cost is lightly decreased while the computation cost is highly increased.

b) Scenario penalizing the power consumption: We have performed extensive simulations in order to gauge the impact of the rate R of UEs generating video traffic on the split decision. Furthermore, we evaluated the impact of the weights α, β and γ on the total cost deployment $\mathbb{C}$ in order to analyze the trade off between link and power consumption. Baseline methods correspond to the different cell-centric configurations.

In what following, we assume that all the previously described parameters are kept static (i.e., $P = 10$ and $ITER_{MAX} = 15$) during the simulation and only the rate R of UEs generating video traffic is varying. In this scenario, the number of UEs is fixed to 50 and R is varying in $[0\%; 100\%]$. The weights α, β and γ are set to 0.8, 0.1 and 0.1 respectively.

In order to emphasize the gap between our proposal and the related strategies, we evaluate, in Fig. 5 (a), the deployment cost $\mathbb{C}$ while increasing the rate of video UEs R. We notice that SPLIT-HPSO achieves the same deployment cost as $split_6$. Besides, it reduces this cost by 85.14% compared with the second strategy corresponding to $split_5$. This can be explained by the fact that the increase of the weight α considerably impacts the deployment cost $\mathbb{C}$ in RRU. Consequently, a fully centralized deployment will achieve the best performances.

c) Scenario penalizing the link consumption: In the same way, we assume that all the previously described parameters are kept static (i.e., $P = 10$ and $ITER_{MAX} = 15$, $N = 50$) during the simulation and only the rate R of UEs generating video traffic is varying in $[0\%; 100\%]$. We set the weights α, β and γ to 0.1, 0.1 and 0.8 respectively. We aim to analyze the split selection strategy of SPLIT-HPSO when the unit cost of the fronthaul link is high.

Fig. 5.(b) depicts the percentage P of deployed user splits versus the rate R of UEs generating video traffic. It is straightforward to see that SPLIT-HPSO selects $split_0$ when there is no users requiring video traffic. In this configuration, the load served for the existing UEs is distributed almost equally. SPLIT-HPSO opts for $split_0$ to lower the costly traffic in the fronthaul link. Whereas, when R is in $[10\%; 40\%]$, we notice that $Split_2$ is predominantly selected to serve the UEs generating web traffic, while $split_1$ is selected to serve UEs generating video traffic. Note that, $split_0$ can be selected when $R = 10\%$. Moreover, when the R exceeds 50%, the competition on radio resources intensifies and the radio resource becomes scares. Hence, the served load for UEs generating video traffic is reduced and the load distribution becomes almost equal between all the cell's UEs. Eventually, the $split_0$ is frequently selected when the load distribution is equal among UEs.

Indeed, $split_0$ is a default solution to lower the costly link consumption. When the load distribution is not equal among

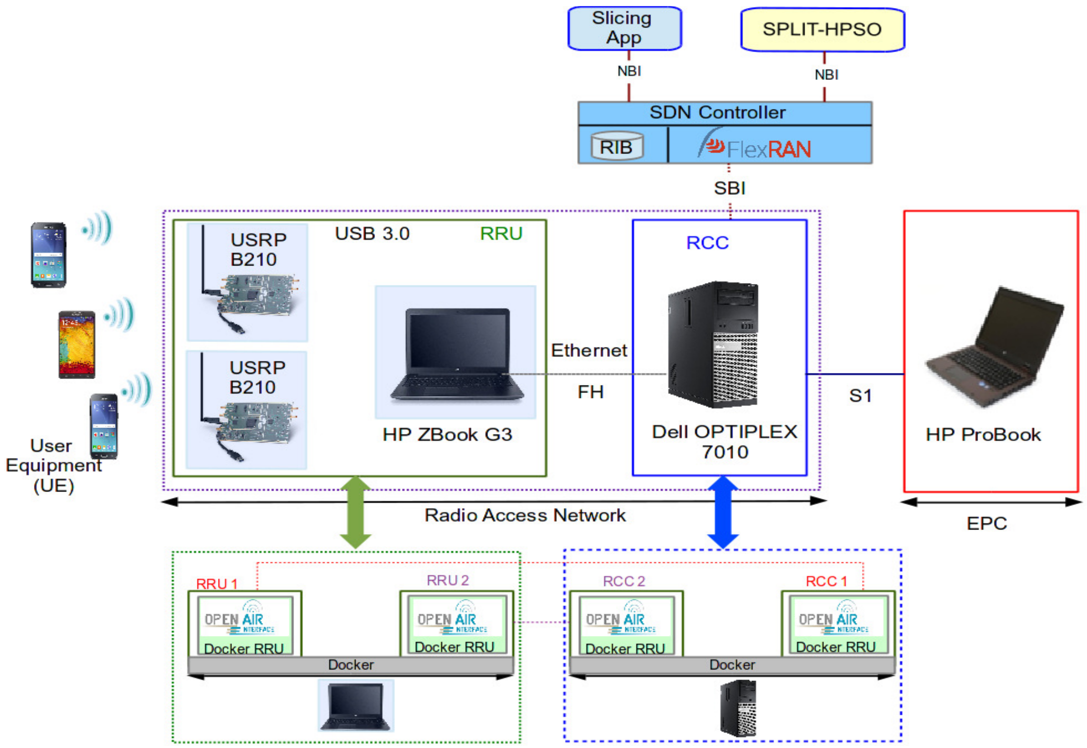


Fig. 6. Testbed architecture.

UEs, the computational consumption of some user processing functions UPF will grow accordingly. Such fact will increase the RRU deployment cost being weighted by α. Hence, $split_0$ is no more a cost effective solution as it deploys all the PFs for high loaded users at RRU. To contract this side-effect, our algorithm finds a satisfactory trade off where it affects both $split_1$ and $split_2$ to its UEs as following. $Split_1$ is deployed priory for UEs with higher loads. $Split_2$ is deployed for UEs with low loads.

d) Trade off between power and link consumption: We assume that all the previously described parameters are kept static (i.e., $P = 10$ and $ITER_{MAX} = 15$, $N = 50$) during the simulation. R is fixed to 50. We aim to understand the trade off between power and fronthaul link consumption. To achieve our objective, we assume that the RCC power consumption weight β is fixed as low as possible to the value 0.01 as data centers are natively efficient in power consumption. We assume that γ is increasing in the range of $[0, 1]$ while α is decreasing in the range of $[0, 1]$. We aim to generate the according split decision and analyze the solution in terms of deployment cost $\mathbb{C}$.

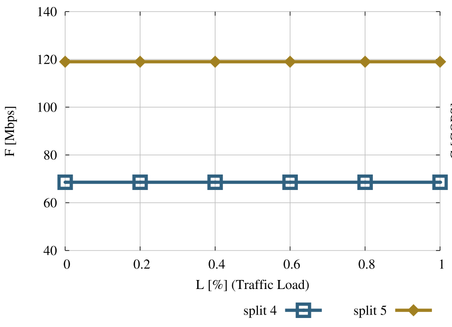
As depicted in Fig. 5.(c), the link consumption $\mathbb{F}$ is stationary for cell splits. Fig. 5.(d) shows that our solution adopts $split_6$ when the fronthaul weight is lower than 0.6 meaning the RRU weight is higher than 0.3. Then, when the fronthaul consumption weight γ is higher than 0.6, the algorithm adopts mainly $split_1$ and $split_2$ until γ reaches 0.9 in order to minimize the traffic in the fronthaul. This explains the important decrease in the link consumption shown in Fig. 5.(c). $Split_1$ is attributed to UEs generating video traffic while $split_2$ is attributed to UEs generating web traffic. When $\gamma = 1$, The fronthaul consumption is highly penalized which explains the adoption of $split_0$ is this case.

B. Experimental Evaluation

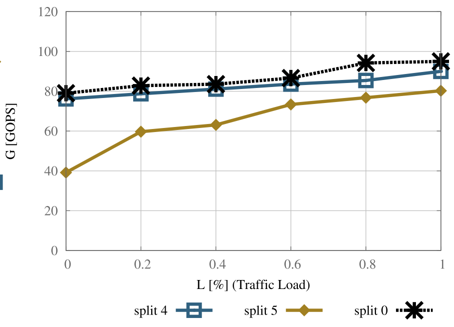
1) Experimental Baselines and Performance Metrics: We validate the feasibility of our approach in a C-RAN prototype based on OAI. Note that the current version of our prototype supports up to 3 types of splits, referred to as LTE ($split_0$), IF4p5 ($split_4$) and IF5 ($split_5$). Accordingly, these cell-centric configurations are considered for our performance evaluation. We also define the following additional metrics.

- $\mathbb{G}$ corresponds to the computational cost of RRU or RCC measured in GOPS.
- $\mathbb{D}$ refers to the served throughput measured in Mbps.

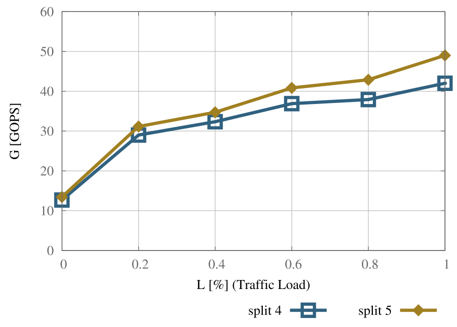
2) Prototype Description: As illustrated in Fig. 6, our prototype makes use of the OAI software [13] and it is compliant with the NGFI C-RAN architecture [7]. The RRU consists of a “Ubuntu 14.04” laptop, equipped with a CPU Intel $i7-6500U$ 4-core (@2.50 GHz), a Random Access Memory (RAM) of 16 GB and 1 Gigabit Ethernet card. The RRU is connected to a Universal Software Radio Peripheral (USRP) B210 card [34] (hereinafter referred to as RRH) via an USB 3.0 interface. By means of 2.4 GHz antennas co-located with the USRP card, the RRU irradiates the 4G signal to the whole cell. The RRU is in turn connected to RCC through an Ethernet “category 5e” patch cable, supporting up to 1000 Mbps. The RCC consists of a server with an Intel i7-3770 8-core (@3 GHz) CPU, a RAM of 16 GB and running with the same operating system as RRU. The RCC is connected to a second laptop, running “Ubuntu 16.04”, equipped with a CPU Intel $i5 - vPro$, 4-core (@2.5 GHz). The latter implements the functionalities of the Evolved Packet Core (EPC), according to the OAI software [13]. Our prototype is connected to



(a) Fronthaul Throughput, $\mathbb{F}$ [Mbps]



(b) Computational Cost at RRU , $\mathbb{G}$ [GOPS]



(c) Computational Cost at RCC, $\mathbb{G}$ [GOPS]

Fig. 7. Key performance indicators in our C-RAN prototype.

3 smartphones that act as Commercial Off-The-Shelf (COTS) users (UEs).

In order to enable the flexibility required by the proposed `AgilRAN` architecture, we virtualized the functions of the LTE protocol stack, by leveraging the Docker technology [28]. The latter is a tool that allows packing applications with all their dependencies in containers. They can share the kernel of the host operating system, while providing user space isolation. Such an isolation feature enables running multiple virtual RRUs within the same host, bypassing the limit of the classic hardware-based implementation of OAI software. Moreover, RRU and RCC containers run directly on top of the kernel, letting us to match the strict performance requirements of the OAI software [13]. Indeed, we have quantified the average deployment time of a container-based PF to 1.8 μs $\pm$ 0.2 μs. Our prototype relies on 2 USRP cards, making it possible to instantiate a maximum of 2 RRU containers at the same host, each connected to a different RCC container.

In order to implement the features of the `AgilRAN` architecture, i.e., enable on-demand functional split deployment, we store multiple images of OAI RRU and RCC at the RRU and RCC hosts, respectively, for each split configuration. As stayted in [14], the Round Trip Time (RTT) of a flow transmission in a one-hop 1 Gbps ethernet-based fronthaul for 5 MHz bandwidth is 300 μs with compression and 550 μs without. Accordingly, the requirement delay of all deployed splits are respected with reference to [8]. According to the output of `SPLIT-HPSO`, our prototype runs the appropriate instance of RRU and RCC images, connecting them via the appropriate fronthaul interface.

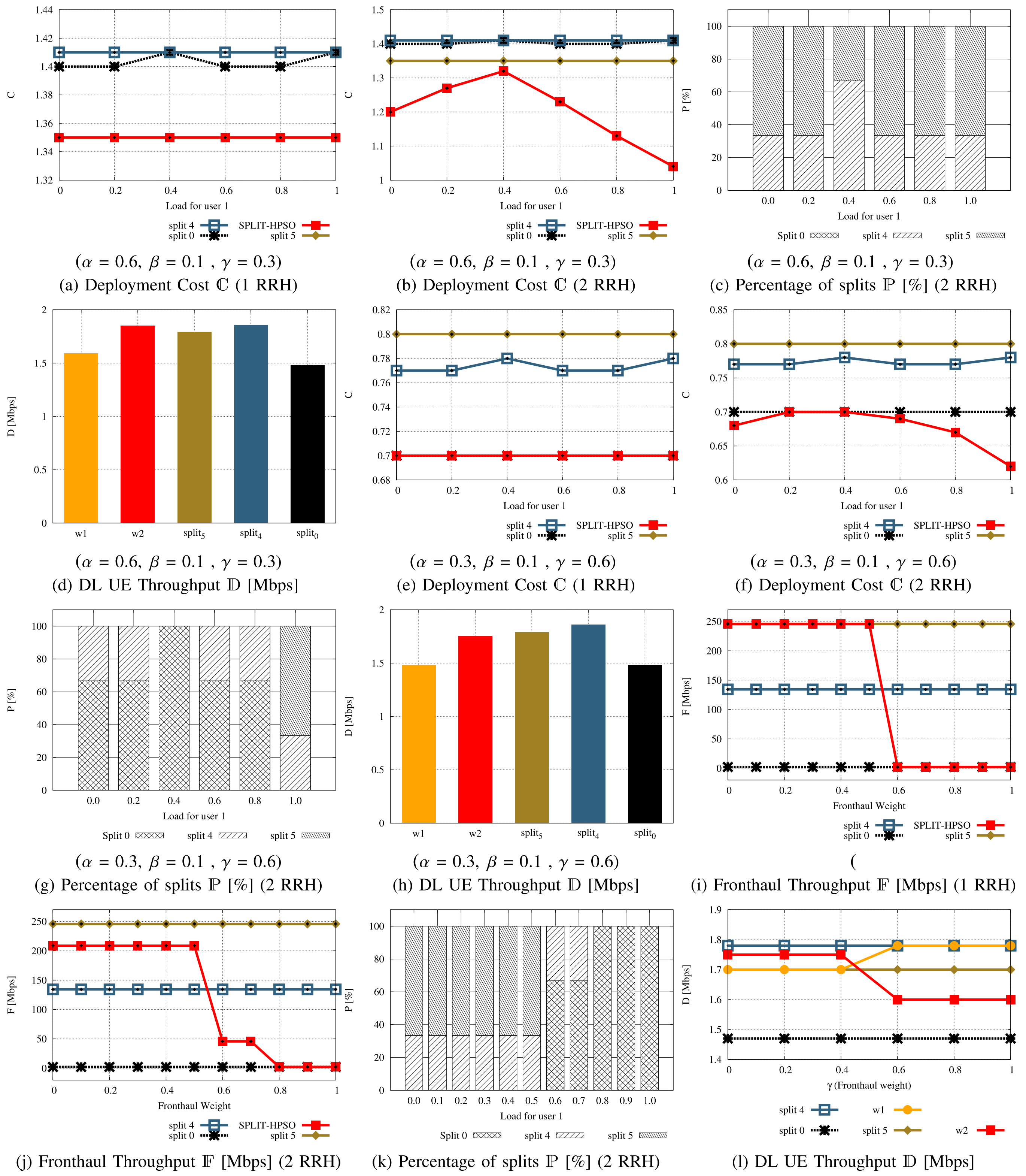
Note that our prototype leverages the SDN paradigm to enable remote allocation of the bandwidth resources in the RCC host. In fact, as it can be seen from Fig. 6, the RCC node is connected to a specific SDN controller, named FlexRAN [21], through a southbound interface (SBI) using Google Protobuf [21]. By means of such an SBI interface, the FlexRAN Controller can easily interact with RCC and hence, collect information about the RAN network state. Moreover, FlexRAN makes available a set of REST North-Bound Interfaces (NBIs), which can be used to manage the RAN environment in an abstract way. From the execution point of view, we are constrained by some limitation in the OAI software which does not support the dynamic configuration. In order to partially overcome such a limitation, we have deployed a script that shuts down and re-activate the OAI base station on-demand.

On the top of FlexRAN, we have implemented 2 northbound applications, referred to as `SPLIT-HPSO` and `Slicing APP` respectively. The former implements the proposed algorithm, while the latter provides the Application Programming Interface (API) for configuring the bandwidth allocation process among different slices in a centralized fashion. More details on the `Slicing APP` can be found in [35], [36].

3) Impact of Functional Splits on Fronthaul Cost: By means of experiments, we evaluate the impact of the different functional splits on the fronthaul traffic, computation cost and power consumption. We set a bandwidth of 5 MHz, while all the UEs are supposed to stream videos from a web-server for the whole duration of the experiments. We are interested in evaluating the impact of the allocated radio resources on the fronthaul bandwidth for each split configuration. To this end, we limit the upper bound of the available RBs at each TTI. This is made possible by using our SDN Slicing APP [35], [36]. Accordingly, we assume that all the users belong to the same slice, and we evaluate the KPI of our prototype by varying the upper limit of bandwidth utilization.

Fig. 7.(a) shows the average of one-way consumed bandwidth by the fronthaul interface for each functional split, by varying the aforementioned upper bandwidth limit. The fronthaul traffic is measured by using the "nload" linux tool, that provides an average (over 300 ms) of the consumed bandwidth in a given network interface. As it can be seen, the fronthaul bandwidth is constant and traffic independent.

Fig. 7.(b) and (c) show the impact of the users' traffic load on the computational cost G (i.e. CPU load) at RRU and RCC respectively, for each type of split. Note that the CPU load metric is made available by the "Docker stats" tool. From Fig. 7.(b), it can be observed that the computation amount needed by $split_0$ at RRU is higher than the computation amount required by the $split_4$ and $split_5$ respectively. This is expected, since $split_4$ and $split_5$ assume to move a set of physical layer (PHY) functions from RRU to RCC. Interestingly, Fig. 7.(b) shows that $split_5$ at RRU outperforms both $split_0$ and $split_4$. Moreover, the $split_5$ gain is higher in lower traffic load scenarios, while decreasing with higher traffic load.



(α = 0.6, β = 0.1 , γ = 0.3)
(a) Deployment Cost $\mathbb{C}$ (1 RRH)

(α = 0.6, β = 0.1 , γ = 0.3)
(b) Deployment Cost $\mathbb{C}$ (2 RRH)

(α = 0.6, β = 0.1 , γ = 0.3)
(c) Percentage of splits $\mathbb{P}$ [%] (2 RRH)

(α = 0.6, β = 0.1 , γ = 0.3)
(d) DL UE Throughput $\mathbb{D}$ [Mbps]

(α = 0.3, β = 0.1 , γ = 0.6)
(e) Deployment Cost $\mathbb{C}$ (1 RRH)

(α = 0.3, β = 0.1 , γ = 0.6)
(f) Deployment Cost $\mathbb{C}$ (2 RRH)

(α = 0.3, β = 0.1 , γ = 0.6)
(g) Percentage of splits $\mathbb{P}$ [%] (2 RRH)

(α = 0.3, β = 0.1 , γ = 0.6)
(h) DL UE Throughput $\mathbb{D}$ [Mbps]

(
(i) Fronthaul Throughput $\mathbb{F}$ [Mbps] (1 RRH)

(j) Fronthaul Throughput $\mathbb{F}$ [Mbps] (2 RRH)

(k) Percentage of splits $\mathbb{P}$ [%] (2 RRH)

(l) DL UE Throughput $\mathbb{D}$ [Mbps]

Fig. 8. SPLIT-HPSO experimental evaluation.

It is worth noting that different from the fronthaul bandwidth model, the computation model is load-dependent. The $split_5$ requires an up to double amount of computation at RRU when 100% of spectrum is used as compared to the scenario with no traffic. Different from RRU, the $split_5$ requires more computation resources at RCC than the $split_4$. This is expected since in the $split_5$ case more PHY layer functions are moved to the Cloud. Note that $split_0$ does not execute any functions at RCC, therefore the cost impact of $split_0$ at RCC will be considered null in our case.

We assume that the input values for our orchestrator are i) users Transport block size (TBS) and the radio load which are values given by the real OAI scheduler, ii) fronthaul status expressed in terms of current Residual link throughput and iii) CPU Residual load converted to computational capacity at both RRU and RCC nodes. Based on these inputs, our aim is to evaluate the optimal split decision for each RRH and its impact on the UEs' throughput. We assume that PUE_U and PUE_C are set to 2.5 and 1.2, respectively. First, we generate one cell split in one execution time to attach 3 UEs to one RRH. Then, we enable our algorithm to generate multiple cell splits, enabling the 3 UEs to be attached to 2 RRHs, with possibly 2 different cell splits.

4) Experimental Results: We expose here the results of a set of experiments conducted in our prototype to validate the feasibility of the proposed approach. Note that our prototype makes use of a 5 MHz carrier bandwidth, that is shared among 2 RRHs, while using SISO antenna mode. In the baseline scenarios (i.e., $split_0$, $split_4$ and $split_5$), we assume that a static split is deployed for all the UEs. Moreover, in a scenario with 1 RRH, only 1 split can be selected by our solution SPLIT-HPSO, while 2 different splits can be selected in a scenario with 2 RRHs. In what follows, 3 static smartphone UEs, are statically located in the proximity of the RRHs, while we assume the UE load varies as follows:

1) Load for UE 1 proportionally increases, getting 0%, 20%, 40%, 60%, 80%, 100% of the available RBs.
2) UE 2 and UE 3 shares the remaining RBs with $2/3$ for UE 2 and $1/3$ for UE 3.

a) Scenario penalizing the power consumption: We set the weights α, β and γ to 0.6, 0.1 and 0.3, respectively. Fig. 8.(a) shows the deployment cost $\mathbb{C}$ for each scheme when only one RRH is deployed. In this scenario, our solution opts for $split_5$ to lower the deployment cost $\mathbb{C}$. Indeed, the high value of α makes the deployment at RRU a costly solution. Fig. 8.(b) shows that our solution can further lower the deployment cost $\mathbb{C}$ in a scenario with 2 RRHs. Indeed, it can be observed from Fig. 8.(c) that our solution chooses mainly $split_4$ and $split_5$ to migrate more functions to RCC. The $split_0$ is excluded here as it is evaluated as a costly solution. Fig. 8.(d) shows the average throughput in downlink (DL) of all UEs. Note that UE throughput has been measured by the Android tool "Simple System Monitor" running at the UE smarthphone. Looking at the baseline scenarios, we observe that $split_4$ offers a better throughput performance. Consequently, when AgilRAN opts for $split_5$ (i.e. when 1 RRH is deployed), the UE throughput is lower than the optimal baseline. However, when AgilRAN opts for $split_4$ for 2 UEs and $split_5$ for 1 UE, the served throughput is increased.

b) Scenario penalizing the link consumption: We set the weights α, β and γ to 0.3, 0.1 and 0.6, respectively.

As it can be seen from Fig. 8.(e), our solution opts for $split_0$ to lower the deployment cost $\mathbb{C}$. In fact, when the fronthaul link weight γ is high, the link consumption becomes a costly solution, which favors the decentralized scheme. From Fig. 8.(f), it can be observed that when 2 RRHs are deployed, $\mathbb{C}$ is load variable for SPLIT-HPSO which employs heterogeneous split decision. As shown in Fig. 8.(g), our algorithm chooses mainly splits 4 and 0 to decrease the link consumption. It can be observed from Fig. 8.(h), that the average UE throughput in AgilRAN is lower than the baseline scenarios, when 1 RRH is used (w1). However, the performance of AgilRAN is significantly improved in a scenario with 2 RRHs (w2), thanks to the reduced interference level compared to the scenario with only 1 RRH.

c) Trade off between power and link consumption: In this experiment, we assume that the load of each UE is fixed as follows: 0.8 for UE 1, 0.13 for UE 2 and 0.06 for UE 3, respectively. We assume that β is fixed as low as possible to the value 0.01, as data centers are natively efficient in power consumption. Accordingly, we assume that γ increases in the range $[0, 1]$, while α decreases in the same range.

As depicted in Fig. 8.(i) and (j), the link consumption is constant for cell splits 4 and 5 as they are load independent and with a small variations for $plit_0$, that is not visible here due to the large-scale. As shown in Fig. 8.(i), AgilRAN adopts for $split_5$ till γ reaches 0.5, in case of 1 RRH. Then, it adopts a full $split_0$ decision when γ is higher than 0.5. Fig. 8.(k) shows that with 2 RRHs, AgilRAN adopts both split 5 ($2/3$ of the time) and $split_4$ ($1/3$ of the time) till the γ reaches 0.5. Then, it gradually adopts $split_0$, till the γ reaches 0.8. Starting from this value, a full $split_0$ decision is made. Finally, Fig. 8.(l) shows the average of UE throughput. As it can be observed, in a scenario with 1 RRH (w1), the UE throughput increases with higher values of γ, while the opposite behavior is observed when 2 RRHs are employed (w2). This is explained by the nature of the split decision, which favors $split_0$ for higher values of γ.

We recall that the goal of the aforementioned experiments is to validate the feasibility to implement the dynamic features of AgilRAN architecture in a real prototype, that in our first implementation does not take into account the radio resource allocation process. Therefore, the UE throughput is not optimized. We will take into account the radio allocation process in future works.

VII. Conclusion

The massive adoption of Cloud technology in mobile access networks has driven the operators and vendors to work together in order to make Radio Access Network (RAN) ecosystem more agile. In this respect, we put forward AgilRAN, a flexible RAN architecture which makes use of Cloud capabilities to enable on-demand deployment of RAN resources while dealing with temporal load variation of users. Thanks to AgilRAN, baseband processing chain is virtualized and splitted in a fine-grained manner. The desagregated basesband processing is then deployed while minimizing jointly the power consumption and fronthaul traffic. To achieve our objective, we propose a heuristic based approach, denoted as SPLIT-HPSO, to optimize the split of the baseband units, while considering both the requirements of its processing network functions and the capabilities of the Cloud infrastructure. Based on Particle Swarm Optimization, SPLIT-HPSO is scalable and achieves optimized user-centric split solution in a satisfactory time. Based on extensive simulations, we have shown that SPLIT-HPSO achieves good performances in terms of total deployment cost and resolution time. Besides, to assess the feasibility of our approach, we implement AgilRAN in an experimental C-RAN platform. Obtained results have proven that our solution ensures a fine-grained link and computational resource allocation while achieving a low deployment cost.

REFERENCES

[1] *Global Mobile Data Traffic Forecast Update, 2017–2022 White Paper*, Cisco Visual Networking, San Jose, CA, USA, Feb. 2019.

[2] *C-RAN The Road Towards Green RAN White Paper Version 2.5*, China Mobile Research Institute, Beijing, China, Oct. 2011.

[3] *Network Functions Virtualization and Software Management White Paper*, Ericsson,Stockholm, Sweden, Dec. 2014.

[4] *CoMP Evaluation and Enhancement*, NGMN, Frankfurt, Germany, Mar. 2015.

[5] 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," *J. Commun. Netw.*, vol. 18, no. 2, pp. 162–172, Apr. 2016.

[6] *Study on New Radio Access Technology: Radio Access Architecture and Interfaces*, Standard 3GPP TR 38.801, Mar. 2017.

[7] *Standard for Packet-based Fronthaul Transport Networks*, Standard 1914.1-2019, NGFI Working Group, 2019.

[8] *Small Cell Virtualization: Functional Splits and Use Cases*, Small Cell Forum, London, U.K., Jan. 2016.

[9] A. Tzanakaki *et al.*, "5G infrastructures supporting end-user and operational services: The 5G-XHaul architectural perspective," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC)*, May 2016, pp. 57–62.

[10] U. Dotsch, 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 Tech. J.*, vol. 18, no. 1, pp. 105–128, Jun. 2013.

[11] S. Matoussi, I. Fajjari, S. Costanzo, N. Aitsaadi, and R. Langar, "A user centric virtual network function orchestration for agile 5G cloud-RAN," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–7.

[12] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci. MHS*, Oct. 1995, pp. 39–43.

[13] (Jul. 2015). *OpenAirInterface Simulator/Emulator*. [Online]. Available: http://www.openairinterface.org/

[14] 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 *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–7.

[15] C.-Y. Chang, R. Schiavi, N. Nikaein, T. Spyropoulos, and C. Bonnet, "Impact of packetization and functional split on C-RAN fronthaul performance," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–7.

[16] J. Liu, S. Zhou, J. Gong, Z. Niu, and S. Xu, "Graph-based framework for flexible baseband function splitting and placement in C-RAN," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 1958–1963.

[17] X. Wang, L. Wang, S. E. Elayoubi, A. Conte, B. Mukherjee, and C. Cavdar, "Centralize or distribute? A techno-economic study to design a low-cost cloud radio access network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–7.

[18] X. Wang, A. Alabbasi, and C. Cavdar, "Interplay of energy and bandwidth consumption in CRAN with optimal function split," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[19] A. Alabbasi and C. Cavdar, "Delay-aware green hybrid CRAN," in *Proc. 15th Int. Symp. Modeling Optim. Mobile, Ad Hoc, Wireless Netw. (WiOpt)*, May 2017, pp. 1–7.

[20] *Network Functions Virtualization (NFV) Management and Orchestration*, ETSI GS NFV-MAN, Paris, France, Dec. 2014.

[21] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis, "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *Proc. 12th Int. Conf. Emerg. Netw. Exp. Technol.*, Dec. 2016, pp. 427–441.

[22] *O-RAN Alliance*. Accessed: Dec. 2019. [Online]. Available: https://www.o-ran.org/

[23] 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 Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[24] R. A. Addad, D. L. C. Dutra, T. Taleb, M. Bagaa, and H. Flinck, "MIRA!: An SDN-based framework for cross-domain fast migration of ultra-low latency 5G services," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[25] I. Farris, T. Taleb, M. Bagaa, and H. Flick, "Optimizing service replication for mobile delay-sensitive applications in 5G edge network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[26] I. Afolabi, A. Ksentini, M. Bagaa, T. Taleb, M. Corici, and A. Nakao, "Towards 5G network slicing over multiple-domains," *IEICE Trans. Commun.*, vol. 100, no. 11, pp. 1992–2006, Nov. 2017.

[27] M. Bagaa, T. Taleb, A. Laghrissi, A. Ksentini, and H. Flinck, "Coalitional game for the creation of efficient virtual core network slices in 5G mobile systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 469–484, Mar. 2018.

[28] *Docker Containers*. Accessed: Aug. 2016. [Online]. Available: http://www.docker.com

[29] *SDR*. Accessed: Feb. 2019. [Online]. Available: https://sdr-lab.u-pem.fr/splitting-C-RAN.html

[30] A. Mignotte and O. Peyran, "Reducing the complexity of ILP formulations for synthesis," in *Proc. 10th Int. Symp. Syst. Synth.*, Sep. 1997, pp. 58–64.

[31] *Data Centers Efficiency*. Accessed: Oct. 2017. [Online]. Available: https://www.google.com/about/datacenters/efficiency/internal/

[32] B. Rouzbehani, L. M. Correia, and L. Caeiro, "A modified proportional fair radio resource management scheme in virtual RANs," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2017, pp. 1–5.

[33] *Study on New Radio Access Technology; Radio Interface Protocol Aspects (Release 14)*, Standard 3GPP TR 38.804 V1.0.0, 2018.

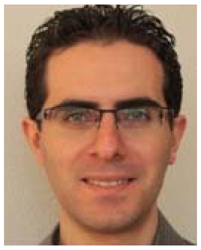
[34] *USRP B200/B210 Specification Sheet*. Accessed: Oct. 2017. [Online]. Available: https://www.ettus.com/product/details/UB200-KIT

[35] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar, "A network slicing prototype for a flexible cloud radio access network," in *Proc. 15th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2018, pp. 1–4.

[36] S. Costanzo, S. Cherrier, and R. Langar, "Network slicing orchestration of IoT-BeC3applications and eMBB services in C-RAN," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2019, pp. 975–976.

Salma Matoussi received the Engineer Diploma and M.S. degrees (Hons.) in computer science and networking from the National School of Computer Science, Tunisia, in 2012 and 2013, respectively. She is currently pursuing the Ph.D. degree with Sorbonne University, France. She worked as a Software Engineer at ILIADE Consulting. She has been a Research Engineer with University Gustave Eiffel, France, since 2019. Her research interests include network orchestration and resource allocation optimization in cloud-RAN.

Ilhem Fajjari (Member, IEEE) received the Ph.D. degree (Hons.) in computer sciences from Pierre and Marie Curie University (Paris 6), France, in 2012. From 2012 to 2014, she worked as a Research Project Leader on network virtualization with VirtuOR Startup. She is a Research Project Leader on cloud-native network function orchestration with Orange Labs. Her main research interests include cloud, network function virtualization, orchestration, and optimization of communication networks. She is a TPC Co-Chair of the IEEE/IFIP CIoT'18. She is an active TPC member of several international conferences including, IEEE ICC, IEEE GLOBECOM, IEEE LCN, IFIP WWIC, IEEE SACONET, IEEE SCNS IEEE/IFIP CIoT, and ISNCC and a reviewer of the majority of the main IEEE and IFIP conferences and journals. She was a Co-Guest Editor of the special issue on cloud edge computing in the IoT in *Annals of Telecommunications* (Springer).



Salvatore Costanzo was a member of the LIGM Laboratory, University of Paris Est Marne-la-Valle (UPEM), France, where he was involved in the design and testing of SDN-based resource management solutions for the RAN. From 2016 to 2018, he was a member of the LIP6 Laboratory, Sorbonne University, Paris, focusing on software-defined radio prototyping solutions. He is a Research Engineer with Orange Labs, working on the design of 5G radio access network (RAN) architectures. He has been an Early Stage Researcher Fellow in the context of the European FP7 Marie Curie Project "CROSSFIRE" and a Visiting Researcher at NEC Laboratories Europe. He is currently involved in 5G open source initiatives, focusing on cloud RAN and network slicing architectures.

Nadjib Aitsaadi (Member, IEEE) is a Full Professor of computer science with UVSQ Paris-Saclay University. He is a co-author of many IEEE/IFIP major journals and conferences. His main research interest includes the optimization of QoS in cellular and wired networks. Also, he is involved in many European projects, such as SARWS, TILAS, GOLDFISH, and so on. He is very active in the IEEE ComSoc Information Infrastructure and Networking Technical Committee (TCIIN) and he is currently the Vice-Chair.

Rami Langar (Member, IEEE) received the M.Sc. degree in network and computer science from Sorbonne University in 2002, and the Ph.D. degree in network and computer science from Telecom ParisTech, Paris, France, in 2006. He was an Associate Professor at LIP6, University Pierre and Marie Curie (now Sorbonne University) from 2008 to 2016, and a Post-Doctoral Research Fellow at the School of Computer Science, University of Waterloo, Waterloo, ON, Canada, from 2006 to 2008. He is currently a Full Professor at University Gustave Eiffel (UGE), France. His research interests include resource management in future wireless systems, Cloud-RAN, network slicing in 5G/5G+, software-defined wireless networks, and mobile Cloud offloading. He is involved in many European and National French research projects, such as X2RAIL-4 (H2020), MobileCloud (FP7), GOLDFISH (FP7), ANR ABCD, FUI PODIUM, FUI ELASTIC, and FUI SCORPION. He was a co-recipient of the Best Paper Award at the IEEE/IFIP International Conference on Network and Service Management 2014 (IEEE/IFIP CNSM 2014) and the Chair of the IEEE ComSoc Technical Committee on Information Infrastructure and Networking (TCIIN) from January 2018 to December 2019.