Cost-efficient Slicing in Virtual Radio Access Networks

Cost-efficient Slicing in Virtual Radio Access Networks

Somreeta Pramanik[a], Adlen Ksentini[b], Carla Fabiana Chiasserini[a]

[a] *Politecnico di Torino, Torino, Italy*
[b] *Communication Systems Department, EURECOM, Sophia Antipolis, France*

Abstract

Network slicing is a promising technique that has vastly increased the manifoldness of network services to be supported through isolated slices in a shared radio access network (RAN). Due to resource isolation, effective resource allocation for coexisting multiple network slices is essential to maximize network resource efficiency. However, the increased network flexibility and programmability offered by virtualized radio access networks (vRANs) come at the expense of a higher consumption of computing resources at the network edge. Additionally, the relationship between resource efficiency and computing cost minimization is still fuzzy. In this paper, we first perform extensive experiments using the vRAN testbed we developed and assess the vRAN resource consumption under different settings and a varying number of users. Then, leveraging our experimental findings, we formulate the problem of cost-efficient network slice dimensioning, named cost-efficient slicing (CES), which maximizes the difference between total utility and CPU cost of network slices. Numerical results confirm that our solution leads to a cost-efficient resource slicing, while also accomplishing performance isolation and guaranteeing the target data rate and delay specified in the service level agreements.

Keywords: Virtual RAN, 5G, network slicing, resource allocation, experimental measurements, optimization

1. Introduction

The future of next-generation cellular networks (5G/B5G) heavily relies on virtualization of network functions and on the slicing of resources for the support of a wide range of services. In-fact, the economic benefits of virtualizing the network infrastructure can be significant, with the RAN representing

an important transformation opportunity. This has resulted in virtualized radio access networks (vRANs) turning into a de-facto sought-after technology for the realization of the emerging open radio access network paradigm [1]. Indeed, the level of virtualization and flexibility that characterize a vRAN make it a perfect fit for the openness and intelligence concepts that are at the basis of the O-RAN architecture [2]. It is therefore expected that open standard radio frequency interfaces, combined with vRAN technologies, will further increase operational savings and increase the scalability of RANs. However, the increased network flexibility and programmability allowed by vRANs come at the cost of a higher consumption of computing resources at the network edge by the vRAN itself [3]. This is a critical aspect that has scarcely been addressed so far: most of the implementations do not account for the demand for computational resources imposed by the radio allocation and, hence, computing resources are typically pooled inefficiently [4, 5]. It follows that the gains currently attainable by a vRAN are far from optimal, preventing its deployment at scale.

Resource allocation at the vRAN is naturally and strictly linked with the concept of network slicing – a key paradigm to guarantee differentiated quality of service (QoS) and service level agreements (SLAs). Network slicing indeed enables multiple logical networks corresponding to different network services running on top of a common physical network infrastructure, with the possibility to customize slices to satisfy various SLAs through isolation techniques [6]. Although network slicing is well researched, slicing the RAN resources is still challenging and requires further study [7]. On one hand, inherent radio spectrum scarcity promotes that all slices share a limited amount of radio resources on demand to ensure efficient utilization. On the other hand, as computing resources in the edge are limited, a cost-efficient resource allocation among the slices is crucial. Towards this latter goal, a slicing strategy is required, by which the operational cost (in terms of, e.g., CPU usage) can be minimized when availability of computing resources is sufficient, and the SLAs are fulfilled when there is a deficit of computing capacity.

In fact, if the service in a slice has elasticity [8], then the resource demand of the slice can change depending upon the operational cost, in order to maximize the slice profit. This inspires us to deeply explore the relationship between computing resource cost and slice dimensioning. While the state-of-the-art [9, 10] on network slicing mainly focuses on offering a satisfying level of QoS or QoE, flow routing and VNF placement, as well as inter-slice

radio resource allocation, none of the existing works designs a cost-efficient slicing strategy accounting for the real-world dependency between the cost of computing resources at the network edge and the ability of the vRAN to support different network slices.

To effectively tackle the above issues, it is essential to dynamically adapt the resource allocation to the various service slices, and the temporal variations of their traffic demand, across virtual Radio Points of Access (RPAs) [11]. Towards this goal, a first, fundamental step is to gain a better hands-on understanding of the behavior of vRPAs and the relation between radio and computing resource dynamics, as well as their dependency upon such factors as radio channel conditions and user's traffic demand. The second required step is to develop a radio slicing strategy that efficiently supports different slices, fulfilling their performance requirements while accounting for the use of computing resources at the network edge. In this work, we address the above challenges, aiming at answering the following research questions:

1. What are the computing requirements of a vRAN, as different settings in terms of number of occupied resource blocks (RBs) and type of modulation and coding scheme (MCS) are adopted?

2. *How do the computing requirements of a vRAN change as the number of connected users varies?*

3. *How can radio resources be sliced to support traffic flows with different characteristics and QoS/SLA requirements?*

We address the first two questions by investigating the behavior of a vRPA using a test-bed implementation and conducting a thorough measurement campaign. In particular, we leverage a srsRAN [12] implementation of an eNB and investigate its CPU consumption under different experimental settings. It is worth noting that CPU utilization is a key metric used to track the system performance behavior, however modern processor technology is much more complex, as a single processor package may encompass multiple cores with dynamically changing frequencies. These technological advances can thus change the behavior of CPU utilization reporting mechanisms. Nevertheless, our analysis is carried out in the same environment, which will not only provide qualitative insights but quantitative predictions as well.

We then address the third question by developing a model that captures the main aspects of a 5G vRAN, and incorporates the relation that we were able to derive from our experiments between CPU utilization and number of users and of radio resources allocated to the deployed slices. By leveraging such model, we formulate and solve an optimization problem for resource

usage reduction, while providing each slice with the requested QoS (namely, data rate and delay) in an isolated fashion. Specifically, our problem dynamically allocates resources to slices so as to maximize the profit of each slice, i.e., the difference between a slice utility (depending on its turn on the slice QoS requirements) and the CPU consumption due to the deployment of the slice itself on the vRAN. To formulate CES, we leverage our experimental findings and use our CPU cost function which is dependent upon the number of occupied RBs and number of connected user equipments (UEs).

To summarize our contributions are as follows.

- We develop an srsRAN-based experimental test-bed and perform extensive experiments, in order to profile the performance limits of the eNB in terms of processing and throughput. We show that the CPU utilization of the eNB increases with the MCS index, number of occupied RBs, and importantly, with the number of connected users.
- Using empirical data, we define regression models to predict the percentage of CPU utilization of the virtual eNB, as the number of connected users and of allocated RBs varies. In so doing, we obtain a prediction accuracy of 99% for CPU utilization.
- Leveraging our experimental findings and the aforementioned approximated models, we formulate the CES problem, which aims at maximizing the slice profit. Importantly, the CES solution turns out to be able to guarantee a high level of isolation among the deployed slices.

The rest of the paper is organized as follows. Section 2 provides an overview of the existing literature, while highlighting the novelty of our work. Section 3 introduces the design and implementation of the vRAN test-bed that we used to derive our results and the experimental findings presented in Section 4. Section 5 describes the vRAN slicing model and optimization, while Section 6 compares the performance of CES to that of static resource slicing (SRS) under different scenarios. Finally, Section 7 draws our conclusions and presents directions for future research.

2. Related Work

Our work relates to two main research directions: modeling and assessment of vRAN performance, and network slicing.

vRAN performance. Owing to the intricate relationship between radio and computing resource dynamics, and the advantages offered by vRANs, several works have aimed at investigating and optimizing such a virtual system. While the studies in [11, 13, 14, 15, 16, 17] focus on evaluating the performance of a vRAN through experiments, the works in [4, 18, 19, 20, 3, 21] provide an insight onto the theoretical framework.

More specifically, [11] is one of the first experimental studies that characterize the potential savings in compute resources when exploiting the variations in the processing load across base stations. Interestingly, [13] presents a linear model to calculate the uplink processing time for a single user in terms of the sub-carrier load, MCS index, and number of antennas. The linear model is then used to develop RT-OPEX, a C-RAN scheduling algorithm. The impact of MCS and SNR on real-time C-RAN processing (i.e., CPU) is studied in [14], along with a mathematical model for predicting the decoding time. The work in [15] profiles instead the performance of a C-RAN in terms of CPU and memory usage, as the iperf transmission bandwidth increases. In [16], the authors investigated the CPU consumption of the baseband unit (BBU) under various conditions for the C-RANs, and characterized the computational demand in terms of throughput. Instead, the work in [17] introduces a processing time model considering the MCS, the number of RBs, and the CPU frequency. However, no such work has investigated and characterized the performance of a vRAN in terms of CPU and memory usage as the number of connected users increases and under diverse settings.

As far as analytical models and algorithmic solutions for the optimization of a vRAN are concerned, [4, 22] set a theoretical basis for CPU-aware radio resource control. In particular, [4] aims at reducing the level of variability of the computational load, by jointly optimizing the selection of the MCS index and the allocation of the physical resource blocks (PRBs). [18, 23] investigate instead the trade-off between the consumption of data processing resources and achievable data rates, taking into account specifically the processing requirements of forward error correction (FEC) on the uplink. A computationally aware MCS selection policy is proposed that reduces the computational complexity requirements, at the cost of slightly decreased spectral efficiency in [18]. The above works rely on the same model relating computational requirements and SNR, and they neglect variations on the arrival bit-rate load. This issue is addressed instead in [24], which combines real-time traffic classification and CPU scheduling in a mobile edge computing setup. However, [24] relies on a simplistic base-band processing model and does not include an

experimental validation. An analytical framework, FuidRAN, is presented in [19], which jointly selects the function split and routing policy, tailored to the available network and computing resources. However, the model is provided for one user only. In [20], a novel reinforcement learning framework is presented, which efficiently allocates radio resources to multiple users in terms of link, MCS index, RBs and airtime for packet transmissions in heterogeneous vRANs. A related relevant contribution is also given in [3] where the vrAIn solution that dynamically learns the optimal allocation of computing and radio resources in order to meet the target level of QoS. A novel pipeline architecture for 5G distributed units (DUs) is presented in [21] to guarantee a minimum set of signals that preserve synchronization between the DU and its users, during computing capacity shortages. This study relies on techniques that require predictable computing to provide carrier-grade reliability.

In conclusion, no previous work characterizes the computing requirements of vRANs with respect to complete contextual dynamics (namely, traffic load and number of users). Importantly, designing resource allocation schemes ignoring such requirements may severely hamper the system performance in terms of throughput.

Network slicing. Owing to its perennial relevance in the future of mobile communication standards, network slicing has received a great deal of attention. Several studies have tackled core networks slicing, and many have focused on placement and management of virtual network functions (VNFs) [25, 26]. Significant challenges instead still exist in the design and management of RAN slicing. Such challenges include how to avoid potential radio resource sharing conflicts, and how to efficiently use radio resources while accounting for the dynamics of service traffic flows as well as the performance isolation among slices.

A survey of solutions for radio resource slicing can be found in [27, 28]. Various resource allocation schemes have been designed to achieve resource isolation among slices, so that the QoS of a slice is not affected by others. Radio resource sharing among multiple tenants is addressed in [29], with the aim to achieve fairness and maximize network utility. Rost et al. [30] deal with the architectural principle of allocating dedicated and shared network functions to slices in both core and access networks. The deployment of service function chains for services, and the related resource allocation, are instead intensively discussed in [9], [10], [31], [32]. To jointly optimize resource allocation in terms of flow routing and VNF placement for multiple co-existing slices, [31] proposes a slice dimensioning scheme, and solves the

dimensioning problem with resource pricing mechanism where the pricing functions are the node and link cost. The study in [33] considers the network slicing problem by maximizing the minimum expected rate (spectral efficiency) of eMBB users over time, while guaranteeing the provisions of uRLLC traffic.

A network slicing framework, named New Radio flexibility (NRflex), is presented in [34], which enables users to benefit from multi-service applications and leverages 5G new radio (NR) numerology to achieve uRLLC services latency while meeting the throughput requirements of eMBB services. A joint eMBB/URLLC scheduling problem is considered in [33] for various eMBB rate loss models, while the uRLLC and the eMBB traffic are dynamically multiplexed through punctured scheduling. The work in [35] proposes a dynamic joint functional split and RAN slicing algorithm with the aim to maximize the throughput by jointly selecting the optimal functional split and routing path from an end user to the central unit. A new slicing scheme aimed at slice performance isolation, as well as efficient capacity utilization and fair resource allocation among users, can instead be found in [36].

We remark that in the majority of the existing works on RAN slicing, radio resources are shared based on a fixed assignment scheme. Arguably, this static resource slicing paradigm can avoid the potential radio resource sharing conflicts among the co-existing RAN slices and thus achieve perfect performance isolation among different slices. However, it is critical to design a cost-efficient RAN slicing control strategy in which the radio resources can be dynamically shared among network slices.

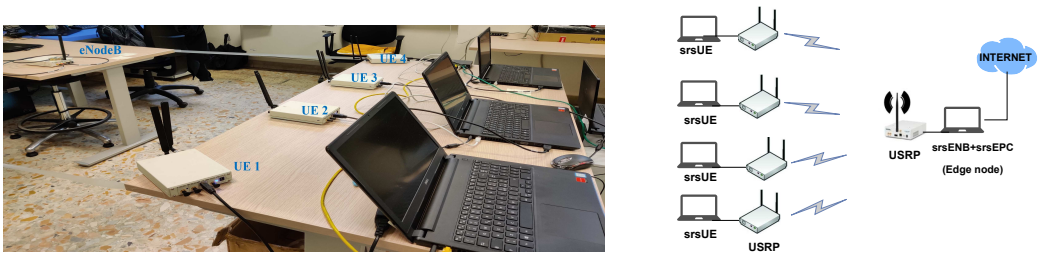
Finally, we mention that a preliminary version of our study has appeared in our conference paper [37] where we have investigated the CPU and memory requirements of vRANs through our srsRAN-based test-bed. In this work, instead, by drawing on our experimental results we characterize the CPU consumption of vRANs, and we develop a cost-efficient radio slicing strategy.

3. vRAN Test-bed

We now introduce our vRAN test-bed using srsRAN, detailing the test-bed architecture and configuration, and the adopted experimental methods.

3.1. Test-bed architecture

Figure 1(a) provides a snapshot of the test-bed we developed, while Figure 1(b) represents its architecture. We leverage software defined radio (SDR)



(a) (b)

Figure 1: vRAN test-bed implementation using srsRAN: (a) snapshot of our test-bed highlighting the edge node hosting the virtual eNB and EPC, and four UEs; (b) test-bed architecture including UEs, virtual eNB and virtual EPC

interfaces enabling point-to-point communications between vRPA and UE. An vRPA implements the necessary processing stack to transfer data to/from UEs. In our case, the vRPA acts as a virtual eNB implemented at the edge of the network. The connectivity between the vRPA and UEs are supported by means of an LTE radio link implemented using the srsRAN [12], an open-source SDR LTE stack implementation offering Evolved Packet Core (EPC), eNB, and UE applications. It is compliant with LTE Release 9 and supports up to 20-MHz bandwidth channels as well as transmission modes from 1 to 4, all using the frequency division duplexing (FDD) configuration. As RF front-end, Ettus Universal Software Radio Peripheral (USRP) B210 devices are used to perform up/down-conversion, filtering, amplification and AD/DA conversion of the UE and eNB LTE signals. All the RF front-ends are connected to the vRPA and the UEs via USB 3.0. Then, physical layer is implemented through a set of OFDMA-modulated channels, using RB filling across ten 1-ms subframes forming a frame.

The edge host and the mobile terminals are each installed in Ubuntu 18.04 systems. The edge host is equipped with an Intel i7-7700HQ 4-cores CPU and 8 GB of DDR4 RAM, while the UEs feature an Intel i7-8550U 4-cores CPU and 16 GB of DDR4 RAM. Each Ubuntu system is connected to USRP B210 boards using USRP Hardware Driver v3.15. In order to faciltate the experiments, all performance management features in the BIOS (e.g., Intel@TurboBoost, Hyper-thread control, Intel SpeedStep) are enabled and C-states have been turned off. The CPU governor of the edge host and the UEs are set to performance mode to allow for maximum computing power and throughput. Moreover, real-time thread priorities are enabled in the srsRAN as the applications (srsENB and srsUE) are executed with root privileges. A

set of threads are created in srsRAN for performance and priority management reasons. Also, we monitor the level of CPU consumption and ensure that, during our experiments, the allocated CPU is sufficient to keep up with the required data rate so as to avoid severe system failures during the radio data transfer. Finally, in order to establish a stable connection, we set the transmit gain (tx_gain) at the eNB to its maximum value.

3.2. Monitoring the srsRAN eNB and UEs

To monitor the behavior and track the performance of the vRAN entities, we leverage some of the useful features of srsRAN (e.g., detailed log system with per-layer log levels, MAC layer Wireshark packet capture, command-line trace metrics, detailed input configuration file). The eNB is configured in band 7 (FDD) and the transmission bandwidth has been set to 10 MHz, corresponding to 50 RBs. In order to determine the successful connection between eNB and UE, the RRC states are observed. Specifically, when the UEs are successfully paired to the eNB, the RRC connection setup message is seen. As experimental set-up, we connected 30 dB attenuators to the antennas of each network node; furthermore, the UEs were placed close enough to the eNB so as to ensure high values of SINR (≥ 25 dB). Finally, we focus on downlink (DL) data transfer and used iperf for data packet generation.

4. Experimental Evaluation and Analysis

In this section, first we present the performance of the vRPA, i.e., the srsRAN eNB, in terms of CPU utilization as the number of occupied RBs and the MCS index vary, when a single UE or multiple UEs are connected. A similar evaluation for the memory utilization can be found in our conference paper [37]. The results have been obtained by averaging over 10 experiments; in every plot, both the average value of the presented performance metric and the corresponding 95% confidence interval are shown.

4.1. CPU utilization

The CPU usage is analysed using an Intel Core i7-7700HQ 2.80 GHz CPU. It is calculated using the `top` process in Linux, which provides a dynamic real-time view of a running system managed by the kernel. Specifically, the percentage of consumed CPU is collected by sampling the `top` reports every second, over a period of 200 seconds.

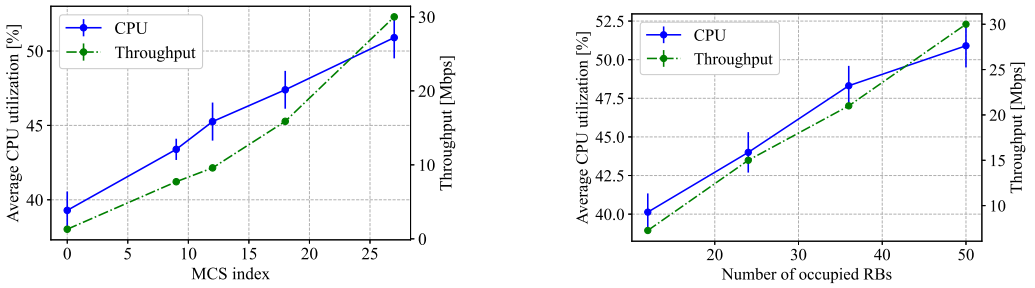


Figure 2: CPU utilization and UDP downlink throughput of the virtual eNB, for different values of the MCS index (a) and occupied RBs (b), and a single connected UE

Figure 2(a) shows the average CPU utilization (left-hand side y-axis) and the throughput (right-hand side y-axis) obtained over 10 iterations of the virtual eNB, as the MCS varies and for a single connected UE. For this experiment, UDP DL traffic is generated at the eNB at 30 Mbps, setting the number of allocated RBs to 50. Also, we set the transmission gain to its maximum value, thus ensuring that the SNR does not drop below 32 dB. From the plot, we can observe that, as also shown in [3], the CPU utilization of the virtual eNB increases as the MCS index grows from 0 to 27. Further, the consumption of computing resources, which is mainly due to the modulation, demodulation, coding and decoding operations, is quite significant in absolute terms: as an example, for MCS= 27, a single user consumes around 51% of a single CPU of the edge node. Using empirical data, we found that the CPU utilization of the virtual eNB can be well approximated as a linear increasing function of the MCS, i.e., $\text{CPU}[\%] = 0.429 \cdot \text{MCS} + 39.58$.

Figure 2(b) shows the CPU utilization (left-hand side y-axis) of the virtual eNB as the number of allocated RBs varies from 12 to 50, for a single UE and MCS= 27. The DL traffic load is set to 9 (for 12 RBs), 15 (for 24 RBs), 21 (for 36 RBs), and 30 (for 50 RBs) Mbps, respectively. We notice that the CPU utilization increases as the number of occupied RBs increases, with a maximum of 51% for a single UE. A higher number of occupied RBs leads to the user transmitting at a higher rate, which results in a higher computational resource consumption. From the experimental data, we found that the CPU utilization of the eNB can be well approximated as a linear increasing function of the number of allocated RBs, a, i.e., $\text{CPU}[\%] = 0.2892 \cdot a + 37.02$. We remark that the provided approximation functions can help interpolate the

average CPU utilization with different radio configurations.

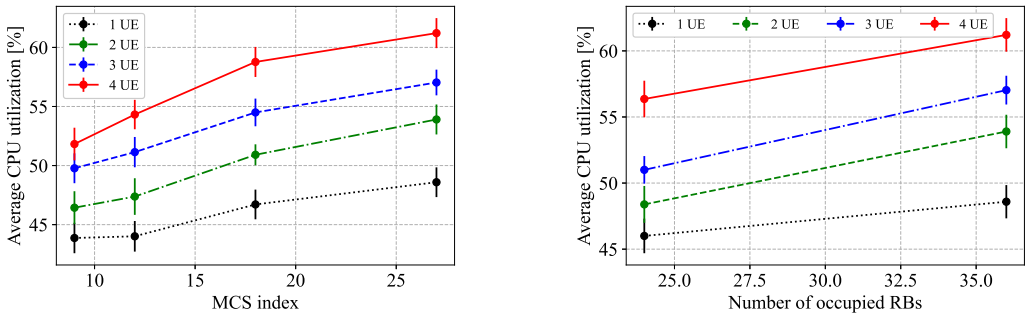
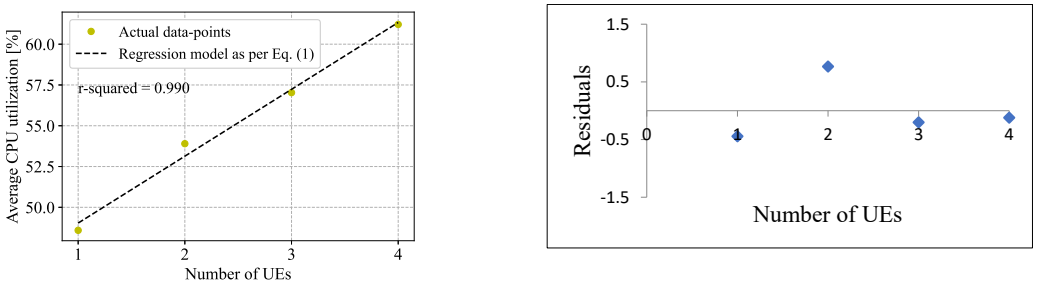


Figure 3: CPU utilization of the virtual eNB for a varying number of connected UEs, versus the MCS index and number of occupied RBs equal to 36 (a), and versus the number of occupied RBs and for MCS= 27 (b)

We are now interested in how the computing resource consumption varies as the number of users connected to the eNB changes. It is indeed a fact that the number of served UEs is rapidly increasing, and that cellular networks will have to support a massive number of users. Figure 3(a) presents the CPU utilization of the virtual eNB as the MCS index varies, for different numbers of users. For this experiment, the overall maximum number of RBs that can be used is set to 36, downlink traffic is generated at 21 Mbps, and the tx_gain is set to its maximum value, so that the SNR is always above 28 dB for all the UEs. In this scenario, an interesting behavior emerges: for a fixed value of the MCS index, the average CPU consumption of the eNB increases significantly as the number of users increases, although the traffic load is kept constant. As an example, for MCS= 27, the average CPU consumption with four UEs is 62% of a single CPU, i.e., about 30% more than with one UE.

In addition, Figure 3(b) shows the CPU consumption of the virtual eNB as the number of allocated RBs varies from 24 to 36, for a different number of connected UEs, MCS= 27, and maximum tx_gain. The DL traffic load for 24 RBs is set to 15 Mbps, while it is 21 Mbps for 36 RBs, so that all the allocated RBs are always occupied. Interestingly, as the number of connected users grows, the CPU consumption of the virtual eNB increases linearly.

Since the MCS index and number of occupied RBs are always finite values that vary over a very specific range, we are mainly interested in understanding the CPU requirements of the virtual eNB as the number of UEs increases.



(a) (b)

Figure 4: (a) Regression plot as the number of UEs varies, for 36 occupied RBs and MCS= 27, (b) Residual plot for the regression model in (1), with 36 occupied RBs and MCS= 27

Then, using the empirical data, we build a regression model that predicts the CPU utilization of the eNB as the number, n, of connected UEs varies. For 36 occupied RBs and MCS = 27, we obtain:

$$CPU[\%] = 4.099 \cdot n + 44.935\,. \tag{1}$$

We remark that similar models can be built for different values of MCS index and number of occupied RBs.

Figure 4(a) shows the CPU utilization under the above settings, along with the curve obtained using the regression model in (1). The Significance F for our model is 0.004, which, being well below 0.05, shows that the model can predict correctly the behavior under study. Further, the regression output (R-squared) indicates that 99% of the variation in CPU consumption is due to the number of UEs. Specifically, every additional UE is expected to entail about 4.1% of increase in CPU usage at the eNB; it follows that 15 users will easily consume up to 100% of a CPU.

Again for 36 occupied RBs and MCS = 27, Figure 4(b) plots the residuals, i.e., the difference in percentage between the actual value of CPU utilization and the one predicted by the regression model, obtained when the number of UEs varies between 0 and 4. We observe that such residuals are always within -0.5% to 1% of CPU usage, which confirms the very good accuracy of the model.

Next, it is important to show that a linear regression model (as in (1)), obtained from experimental data, correctly characterizes the computing requirements of a vRAN as the number of users increases. To this end, we use

all the experimental data obtained for a number of UEs up to 3 (i.e., for a varying number of occupied RBs and MCS indices), and we predict the CPU consumption when four UEs are connected. The corresponding regression model is given by:

$$CPU[\%] = 3.9 \cdot n + 0.369 \cdot \text{MCS} + 35.658 \tag{2}$$

where n is the number of connected UEs and MCS is the adopted MCS index. Similarly, the CPU consumption as a function of the number of connected UEs (n) and number of allocated RBs (a) can be written as:

$$CPU[\%] = 3.9 \cdot n + 0.44 \cdot a + 30\,. \tag{3}$$

Table 1: Actual vs. predicted CPU usage with 4 UEs, a varying number of occupied RBs, and MCS = 27

No. allocated RBs	**Actual CPU** [%]	**Predicted CPU** [%]
24	56.36	55.51
36	61	60.759

Table 2: Actual vs. predicted CPU usage with 4 UEs, different MCS indices, and 36 occupied RBs

MCS index	**Actual CPU** [%]	**Predicted CPU** [%]
12	54.31	55.55
18	58.76	57.71
27	61	61.19

Table 1 and Table 2 report the actual CPU utilization when 4 UEs are connected, and the corresponding value predicted through (2) and (3). Furthermore, Figure 5(a) plots the residuals obtained for the model in (2) and Figure 5(b) those obtained for the model in (3): again, all residual values are within -1% and 1%. These results, along with an F-statistic value of 0.0023, indicates that the linear regression model well describes the behavior of CPU utilization.

The experiments are performed with at least one connected UE. For MCS= 27 and RBs= 36, the predicted CPU utilization from (2) is 45.62%, which is very similar to the value of 45.84% obtained from (3) with no connected UE. Further, in Fig. 2(a) and Fig. 2(b) we validate our models in (2)

(a) (b)

Figure 5: (a) Residuals plot for the regression model in (2), for different values of the MCS index and 36 allocated RBs; (b) Residuals plot for the regression model in (3), for a different number of allocated RBs and MCS = 27

and in (3) with the linear functions provided for one UE, as functions of the MCS (varied from 0 to 27) and the number of allocated RBs (varied from 12 to 50), and considering as minimum values of the MCS and the number of allocated RBs 0 and 12, respectively. We can observe that the CPU usage is almost identical with the same parameter settings for both models, which confirms the accuracy of the prediction models in (2) and (3).

In order to utilize the linear model with diverse parameter settings (different MCS and RB configurations), all our experimental data can be exploited to derive a model as the number of UEs (n), MCS, and RBs vary. The corresponding regression model is given by,

$$CPU[\%] = 3.46 \cdot n + 0.325 \cdot \text{RBs} + 0.28 \cdot \text{MCS} + 26.55 \tag{4}$$

For MCS= 27 and number of occupied $RBs = 36$, the predicted CPU usage from the model in (4) is 59.65% for 4 connected UEs, while the actual CPU usage from the experiments is 61% for the same parameter settings. Figure 6 plots the residuals obtained for the model in (4) showing that all residual values are within -1.5% and $+1.5\%$. These results, along with an F-statistic value lower than 0.05, indicate that the linear regression model well describes the behavior of CPU utilization. It is worth mentioning that the CPU utilization at the eNB can be accurately predicted also when different values of MCS are used for different UEs. Indeed, since the MCS index takes discrete values over a very specific range, using in (4) the average value of the MCS indices allocated over the different users still provides an accurate estimate of the CPU consumption at the eNB.

Finally, it is important to underline that the relationship between CPU consumption and factors such as the MCS, the number of allocated RBs, and

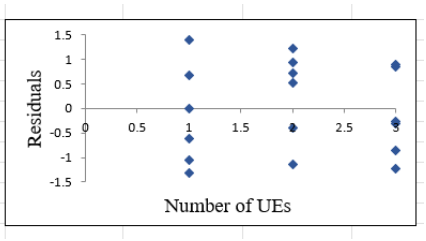


Figure 6: Residual plot for the regression model in (4), for different values of the MCS index and occupied RBs

the number of connected UEs may become non-linear for a certain number of UEs. Although, due to hardware constraints, we had to restrict our analysis to 4 UEs, we have profiled the downlink scheduler when UEs are varied from 1 to 4 and it was found that the processing time increases with the number of users: as an example, the downlink scheduler processing time for 1 UE is 10 μs, while for 4 UEs it is 20 μs. Additionally, some recent work [21] confirms that the processing time of downlink transmission tasks does vary with the number of users (e.g., scheduling becomes more complex). Based on our experiments, we observed that under low values of MCS (as shown in Fig. 3(a)), the primary CPU consumption is not due to the MCS, but rather to the increment in the number of users. For instance, for MCS= 9, the CPU consumption is 43.63% for 1 UE, while it is 52% for 4 UEs. For MCS= 27, the CPU consumption is 48.58% for 1 UE, while it is 61.209% for 4UEs.

To conclude, our proposed linear relationship models hold for the downlink[1] traffic transfer with any value of MCS between 0 and 27, occupied RBs from 0 to 50 (i.e., for a 10 MHz channel), and a number of UEs from 1 to 4. From our experiments, we found that, for higher values of MCS index (i.e., more sophisticated modulation and coding schemes are used), number of allocated RBs (i.e., higher-rate transmissions take place), and number of users, the CPU consumption increases linearly. Moreover, the dominant impact on CPU consumption is due to the number of connected UEs.

[1]It is worth observing that uplink data transfer will unarguably influence the analysis of the CPU utilization: e.g., the processing time for decoding is longer than for encoding, and it increases with the MCS index, as shown in [3, 11, 21].

5. vRAN Slices: Modeling and Optimization

We now leverage the characterization of the vRAN computational requirements given in Section 4, to develop a solution framework for designing and optimizing vRANs slicing. In particular, after modeling the vRAN and the slices supported therein (Section 5.1), we formulate the problem of cost-efficient slice (CES) dimensioning, which maximizes the slice profit while accounting for the CPU cost (Section 5.2).

5.1. System model

We focus on a gNB supporting a group of users ($\mathcal{E}$) requiring eMBB service, and a set of users ($\mathcal{U}$) demanding uRLLC service. For simplicity of notation, we consider a set of slices $\mathcal{M}$ including only a single eMBB and a single uRLLC slice, although the extension to the case of multiple eMBB and uRLLC slices is straightforward. Time is divided into Transmission Time Intervals (TTIs), denoted by $t \in \mathcal{T} = \{1, 2, \ldots, T\}$. Radio resource is divided both in the frequency domain and in the time domain, yielding F RBs, each of bandwidth B. Considering an equal power allocation, the SINR of the generic UE i at time t is given by $\gamma_{i,j,t} = \frac{P \cdot H_{i,j,t}}{B \cdot N_0}$, where P is the transmit power of the gNB, $H_{i,j,t}$ is the channel gain of user i on RB j at time t, and N_0 is the power of additive white Gaussian noise (AWGN).

For the conventional services, such as eMBB with large transmitted packet size, the achievable data rate of UE i for RB $j \in \mathcal{F}$ at the t-th TTI can be directly estimated according to Shannon's capacity as written below in (5):

$$r_{i,j,t} = \begin{cases} \Delta t \cdot B \log_2(1 + \gamma_{i,j,t}) & (5) \\ \Delta t \cdot B \Big[\log_2(1 + \gamma_{i,j,t}) - \sqrt{\frac{C_{i,j,t}}{l_{i,j,t}}} Q^{-1}(\epsilon) \log_2 e) \Big] . & (6) \end{cases}$$

However, for the short-sized packet transmission (ranging from 32 bytes to 200 bytes), such as uRLLC [38], the data rate falls in the finite block-length channel coding regime [39]. Therefore, the data rate are modeled as (6), where

- Δt is the time duration of one TTI, which set to 1 ms,
- ϵ is the transmission error probability,
- $Q^{-1}(\cdot)$ is the inverse of the Gaussian Q-function,

- $l_{i,t}$ represents the length of the codeword block in symbols and can be obtained based on the selected numerology for the uRLLC slice,
- $C_{i,j,t}$ is the channel dispersion, which depicts the stochastic variability of the channel relative to a deterministic channel with the same capacity, given by $C_{i,j,t} = 1 - \frac{1}{(1+\gamma_{i,j,t})^2}$.

Notice that, to guarantee $(1-\epsilon)$ reliability for the transmission of $r_{i,j,t}$ bits per TTI towards a user, it is required to assign sufficient RBs with a large SNR. Thus, we consider that the SNR for every user on each RB never drops below 5 dB [40].

The achievable rate of an eMBB UE, $e \in \mathcal{E}$, in TTI t is thus given by:

$$r_{e,t} = \sum_{j=1}^{F} \alpha_{e,j,t} \cdot r_{e,j,t} \tag{7}$$

where binary variable $\alpha_{e,j,t} = 1$ indicates that the j-th RB is allocated to UE e, and $\alpha_{e,j,t} = 0$ otherwise. The achievable rate of an uRLLC UE, $u \in \mathcal{U}$, in time slot t is instead given by:

$$r_{u,t} = \sum_{j=1}^{F} \beta_{u,j,t} \cdot r_{u,j,t} \tag{8}$$

where binary variable $\beta_{u,j,t} = 1$ indicates that the j-th RB is allocated to UE u and $\beta_{u,j,t} = 0$, otherwise.

Next, we introduce the SLA model, which includes data rate and packet latency as performance metrics. While the former can be derived by aggregating the amount of data that is successfully transmitted over time, a queuing model of UEs' packets is needed to derive the latter.

To this end, we assume that each slice has its DL queue at the gNB, and all packets belonging to a slice share the same queue. We then model the uRLLC slice queue at the gNB as an M/M/1/K queue with service rate μ and traffic arrival rate λ [41]. As μ depends upon the scheduling process at the MAC layer, while λ corresponds to the traffic rate of the users running on top of the slice, we write:

$$\mu_{u,t} = \frac{\sum_{j=1}^{F} \beta_{u,j,t} \cdot r_{u,j,t}}{L} \tag{9}$$

$$\lambda = \frac{|\mathcal{U}| \cdot d_u}{L} \tag{10}$$

where L is the packet size of the uRLLC application, $|\mathcal{U}|$ is the number of UEs belonging to the uRLLC slice, d_u is the traffic arrival rate of uRLLC service per user, and $u \in \mathcal{U}$. The queue length at the t-th TTI can be derived as [42]

$$q_{u,t} = \frac{1-\rho_{u,t}}{1-\rho_{u,t}^{K+1}} \sum_{k=0}^{K} k\rho_{u,t}^{k} \tag{11}$$

where $\rho_{u,t} = \frac{\lambda}{\mu_{u,t}}$. Little's law can then be applied to estimate the latency experienced by uRLLC packets in the corresponding queue:

$$\delta_{u,t} = q_{u,t}/\lambda\,. \tag{12}$$

At the t-th TTI, the delay of a packet arriving at the i-th UE is given by the sum of transmission delay and queuing delay,

$$D_{u,t} = W_{u,t} + \delta_{u,t} \tag{13}$$

where the transmission delay, $W_{i,t}$, is the queue service time, which depends upon the datarate used to transmit towards the UE (see (8)). We then write the average packet delay of the $|\mathcal{U}|$ uRLLC UEs at the t-th TTI as,

$$\overline{D}_t = \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} D_{u,t}\,. \tag{14}$$

5.2. Cost-effective slicing (CES)

In this section, we formulate the problem of ensuring a Cost-Efficient Slicing (CES) of the vRAN, and provide some details of the problem solution.

Our goal is to obtain an optimal vRAN slicing control strategy that maximises the expected long-term profit of all slices. Such profit is defined as the difference between the sum of utility of all eMBB UEs across T TTIs and the normalized cost of computing resource consumption due to the slices supported on the vRAN. Specifically, the objective is twofold: (i) when the CPU capacity is sufficient, the goal is to minimize the operational cost (in terms of CPU usage) as long as the deployed slices meet the desired performance; (ii) when there is a deficit of computing capacity to meet such performance target, the aim is resource efficiency, i.e., to maximize the data rate of eMBB services and minimize the delay experienced by uRLLC users. By taking $\alpha_{e,j,t}$ and $\beta_{u,j,t}$, indicating the RBs allocation for the eMBB and

uRLLC slices, as decision variables, the CES problem formulation is given by:

$$\max_{\{\alpha\},\{\beta\}} \mathbb{E}_{t\in\mathcal{T}}\left[\sum_{e\in\mathcal{E}} U_{e,t} - \sum_{m\in\mathcal{M}} \phi_m^t\right] \tag{15}$$

$$\text{s.t.} \sum_{e\in\mathcal{E}} \alpha_{e,j,t} + \sum_{u\in\mathcal{U}} \beta_{u,j,t} \leq 1, \quad \forall j \in \mathcal{F}, \forall t \in \mathcal{T} \tag{15a}$$

$$\overline{D}_t \leq D_{max} \quad \forall t \in \mathcal{T} \tag{15b}$$

$$\alpha_{e,j,t} \in \{0,1\}, \beta_{u,j,t} \in \{0,1\} \quad \forall u \in \mathcal{U}, \forall e \in \mathcal{E}, j \in \mathcal{F}. \tag{15c}$$

The utility ($U_{e,t}$) of the generic eMBB user e at TTI t is given by the eMBB user data rate on that TTI, i.e.,

$$U_{e,t} = \begin{cases} 1 - \text{erf}(x^{th} - x^o_{e,t}) & \text{if } x^o_{e,t} \geq x_{th} \\ \text{erf}(x^{th} - x^o_{e,t}) & \text{otherwise} \end{cases} \tag{16}$$

where

- x^{th} is the target data rate of an eMBB UE and $x^o_{e,t}$ is the observed data rate of user e on TTI t. Our choice of erf function for estimating individual UEs utility is motivated by its shape, which takes 0 value at the origin, and gradually increases (decreases) and saturates to the maximum (minimum) value in the positive (negative) direction [20]. When the target is met, the utility value is positive and it further increases to its maximum value at +1, as the observed data rate approaches its target value. Likewise, when the target is not met, the value of the utility is negative, which further reduces and saturates to the minimum value 1 as the observed data rate moves away from the target. Moreover, it is essential to keep the observed data rate as close as possible to the respective target for optimum utilization of network resources: substantially better values than the target ones would indeed translate into a waste of resources. Thus, our choice of utility function equally accounts for the aforementioned properties;

- ϕ_m is the cost of computing resource consumption for deploying slice $m \in \mathcal{M}$, which, based on our experimental findings and model in Section 3, is given by

$$\phi_m = 3.9 \cdot n_m + 0.44 \cdot a_m + 30 \quad \forall m \in \mathcal{M} \tag{17}$$

where n_m is the number of users served by slice m and a_m is the number of RBs allocated to the slice.

Constraint (15a) limits the RB resources, while (15b) guarantees that the average uRLLC users' packet delay will not exceed the target value D_{max} at any TTI. Constraint (15c) ensures binary-valued $\alpha_{e,j,t}$ and $\beta_{u,j,t}$.

5G NR, adhering to the principles of OFDMA technology, supports multiple waveform configurations, which results in scalable numerology. A numerology represents a set of parameters such as subcarrier spacing (SCS), PRB bandwidth, time-slot duration, and OFDM symbol duration. While LTE supports carrier bandwidths of up to 20 MHz with a mainly fixed OFDM numerology (15 kHz SCS), 5G NR offers scalable OFDM numerologies by scaling the basic LTE SCS by 2^{μ}, where μ is an integer between 0 and 4. The numerology is selected independently from the frequency band, with possible SCS of 15 kHz to 240 kHz. Regardless of the numerology, the length of a radio frame and a subframe are always 10 ms and 1 ms, respectively, while the difference is represented by the number of time slots within a subframe. The coefficients in our model in (3), which represents the rate of change in CPU utilization as the number of RBs and users increase, are independent of numerology. Moreover, as already discussed in Sec. 5.2, we use the normalized cost of CPU resource consumption while designing our slicing solution. As a result, using the flexible frame structure of 5G NR, our models can significantly help design a slicing solution for 5G vRAN.

The problem formulation, along with the above constraints, results in a mixed integer quadratically constrained programming (MIQCP) problem. Moreover, the problem includes non-positive semi-definite quadratic equality constraints. To find the CES solution, we used Gurobi [43] where the non-linear functions are approximated as as piece-wise linear functions. When solving the model, the objective bounds section provides information on the best known objective value for a feasible solution (i.e., the objective value of the current incumbent), and the current objective bound provided by leaf nodes of the search tree. A new feasible solution is found, either by a MIP heuristic or by branching. When the gap between the best feasible solution and the best bound is smaller than the default MIPGap parameter (set to 10^{-4}), Gurobi produces an optimal termination status. Although a MIP problem is in general known to be an NP-complete problem, we were able to solve the model with an optimality gap that is at maximum just 0.01%.

6. CES Performance Evaluation

In this section, we demonstrate the effectiveness of our proposed network slice dimensioning method, CES, while also considering isolation guarantees.

We set the system parameters as presented in Table 3, and we compare the slice profit of CES to static resource slicing (SRS), where slice requests are processed without considering the CPU cost of the gNB due to slicing. Clearly, the fewer the RBs assigned to a slice, the higher the profit of the slice. Recall that two different slices are considered in our analysis, namely, eMBB and uRLLC; also, the results are obtained considering two UEs (Figure 7) and four UEs (Figure 8) connected to a gNB for each of the slices.

Table 3: Parameter settings

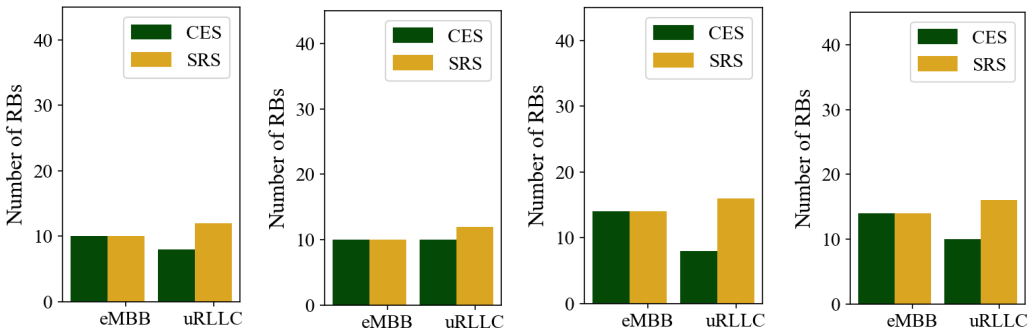
Parameter	**Value**
Number of RBs (F)/RB Bandwidth	50/180 kHz
gNB transmit gain	80 dB
D_{max}	5 ms
Service type	eMBB and uRLLC
Packet size	800 bytes (eMBB); 200 bytes (uRLLC)

Comparison of slice profit. The plots in Figure 7 and Figure 8 for the two considered scenarios present the number of RBs allocated to the slices, respectively, under our proposed scheme (CES) and under the considered benchmark (SRS).

For the first scenario, in the first pair of plots (Figures 7(a) and 7(b)) and in the second pair of plots (Figures 7(c) and 7(d)), the target eMBB data rate for every UE is set to 2 Mbps and 3 Mbps, respectively, while two different values of uRLLC traffic demand are considered, namely, 0.4 packets/TTI in Figures 7(a) and 7(c), and 0.8 packets/TTI in Figures 7(b) and 7(d). The results highlight how the number of RBs allocated to the eMBB and uRLLC slices is lower under CES compared to SRS. Also, notice that the requirements in terms of delay for uRLLC traffic and data rate for eMBB traffic are always fulfilled, under both CES and SRS, as reported in Table 4

For the second scenario, in the first pair of plots (Figures 8(a) and 8(b)) and in the second pair of plots (Figures 8(c) and 8(d)), the target eMBB data rate for every UE is set to 2 Mbps and 3 Mbps, respectively, while two different values of uRLLC traffic demand are considered, namely, 0.1 packets/TTI

in Figures 8(a) and 8(c), and 0.4 packets/TTI in Figures 8(b) and 8(d). The results highlight how the number of RBs allocated to the eMBB and uRLLC slices is less under CES compared to SRS, and it equals that of SRS only for high traffic demand of the eMBB slice. Additionally, we remark that the target delay for uRLLC traffic is always fulfilled, under both CES and SRS.



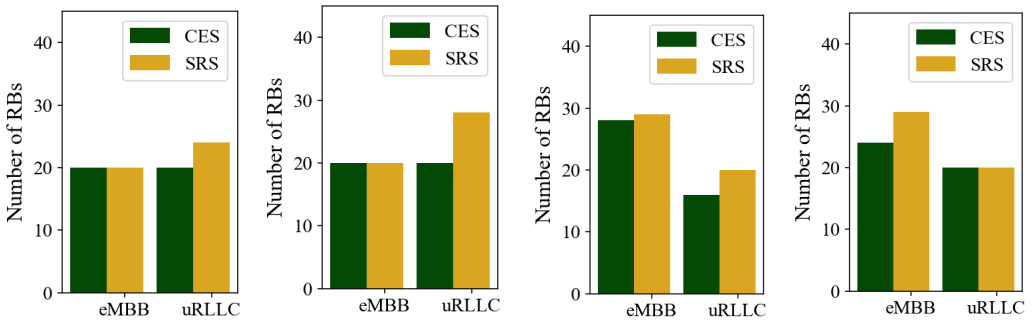
(a) eMBB=2Mbps uRLLC=0.40pkts/TTI (b) eMBB=2 Mbps uRLLC=0.8 pkts/TTI (c) eMBB=3 Mbps uRLLC=0.4 pkts/TTI (d) eMBB=3 Mbps uRLLC=0.8 pkts/TTI

Figure 7: Comparison of the number of RBs allocated to the slices at each TTI, under CES and SRS. The traffic demand of each eMBB UE is set to 2 Mbps in (a) and (b), and to 3 Mbps in (c) and (d)

The plots confirm that CES is more efficient than SRS in the support of both the uRLLC and eMBB slice: CES is able to reduce the radio resource consumption even in the presence of high eMBB traffic demand. Additionally, by looking at both Figure 8(d) and Table 5, we notice that CES can reduce the cost due to slices support also when both the slices exhibit high traffic demand, by compromising at maximum 20% of the target eMBB data rate.

In conclusion, in all of the considered scenarios, the radio resource consumption is lower under CES than under SRS, which confirms the validity of our approach. In summary, CES performs a dual role of reducing the CPU cost and at the same time fulfilling the SLA requirements (i.e., increasing the utility of eMBB users to meet the target rate and maintaining uRLLC target delay). This strategy drives CES to efficiently allocate radio resources at the edge devices where computing resources are constrained.

Slice isolation. Isolation performance in network slicing can be evaluated by measuring the impact that changes (e.g., in traffic demand) occurring in certain slices have on another slice. To correctly measure and evaluate the isolation performance of the CES scheme, we consider two scenarios:



(a) eMBB=2Mbps uRLLC=0.10pkts/TTI (b) eMBB=2 Mbps uRLLC=0.4 pkts/TTI (c) eMBB=3 Mbps uRLLC=0.1 pkts/TTI (d) eMBB=3 Mbps uRLLC=0.4 pkts/TTI

Figure 8: Comparison of the number of RBs allocated to the slices at each TTI, under CES and SRS. The traffic demand of each eMBB UE is set to 2 Mbps in (a) and (b), and to 3 Mbps in (c) and (d)

(i) Given the eMBB traffic demand, the traffic demand of the uRLLC slice is varied and the subsequent effect on the data rate of eMBB users is evaluated;

(ii) The delay experienced by uRLLC traffic is assessed, as the traffic demand of the eMBB slice varies while keeping that of the uRLLC slice fixed.

Table 4 and Table 5 illustrate the observed data rate of eMBB UEs ($x^o_{e,t}$) and experienced delay of uRLLC UEs ($\overline{D}_t$), for different values of uRLLC and eMBB traffic demand. Looking at the difference between the target and achieved data rate, it can be noted that the performance of the eMBB slice under CES is not affected much by the variation of uRLLC traffic and this holds also for different values of eMBB demand. Table 4 and Table 5 also present the observed uRLLC delay for different values of eMBB traffic. Importantly, the variation of the eMBB demand does not affect the delay of the uRLLC slice, which remains always below the max tolerable delay value (set to 5 ms) regardless of the value of uRLLC traffic, thus highlighting again a very good level of isolation between the two slices.

We evaluated the performance of CES only in terms of the number of allocated radio resources, since, as it can be noted by comparing (2) and (3), the dominant impact on the CPU consumption is represented by the number of connected UEs, rather than by the number of allocated RBs. In addition, we would like to highlight that further considerations about the

Table 4: Achieved data rate of eMBB UEs [Mbps] and average delay for uRLLC UEs [ms] for CES and SRS, as uRLLC traffic demand [Mpbs] and target eMBB data rate [Mbps] vary for 4 connected UEs.

uRLLC traffic	x^{th}	**CES** $x^o_{e,t}$	**SRS** $x^o_{e,t}$	**CES** $\overline{D}_t$	**SRS** $\overline{D}_t$
0.4	2	2	2	0.22	0.14
0.8	2	2	2	0.20	0.16
0.4	3	2.96	2.96	0.22	0.10
0.8	3	2.96	2.96	0.43	0.17

Table 5: Achieved data rate of eMBB UEs [Mbps] and average delay for uRLLC UEs [ms] for CES and SRS, as uRLLC traffic demand [Mpbs] and target eMBB data rate [Mbps] vary for 8 UE case.

uRLLC traffic	x^{th}	**CES** $x^o_{e,t}$	**SRS** $x^o_{e,t}$	**CES** $\overline{D}_t$	**SRS** $\overline{D}_t$
0.1	1	0.96	0.96	0.25	0.13
0.4	1	0.96	0.96	0.2	0.11
0.1	2	2.0	2.0	0.25	0.19
0.4	2	2.0	2.0	0.20	0.13
0.1	3	2.8	2.96	0.36	0.25
0.4	3	2.4	2.96	0.2	0.2

CPU consumption can be made starting from the plots in Figure 7 and Figure 8, which show how CES allocates a lower number of RBs to the eMBB and uRLLC slices, with respect to SRS. The smaller the number of radio resources allocated, the lower the CPU utilization of the virtual gNB according to (3).

7. Conclusions and Future Work

Virtualized radio access networks (vRANs) are the basis of next-generation base stations design. To provide real-world insights and key inputs to design optimized resource management in vRANs, we investigated and characterized the computational requirements of vRANs by developing an srsRAN-based test-bed. Through extensive experiments, we profiled the CPU utilization of the vRAN. Our results shed light on the vRAN behavior across different scenarios, showing that, remarkably, the CPU utilization of the eNB increases substantially with the number of users. It is worth underlining that the re-

sults have been obtained under a constant value of traffic load and number of occupied resource blocks. Based on these empirical results, we also built linear regression models for the prediction of CPU utilization as the number of users varies.

Then, leveraging our experimental findings, we formulated the problem of cost-efficient network slicing (CES). The numerical results confirmed that our solution leads to a cost-efficient resource slicing with 10-15% reduction in radio resource consumption, while also accomplishing performance isolation and meeting the data rate and delay specified in the service level agreements of, respectively, eMBB and uRLLC slices.

Future work will consider a wider range of services and applications, and it will investigate how radio resources allocation can be further improved by exploiting 5G numerology.

Acknowledgement

The work has been partially supported by POLITO/EURECOM joint Ph.D. program under Zero-touch 5G networks management project, and by the NPRP-S 13th Cycle grant No. NPRP13S-0205-200265 from the Qatar National Research Fund (a member of Qatar Foundation). The findings herein reflect the work, and are solely the responsibility, of the authors.

References

[1] O. Alliance, O-ran: Towards an open and smart ran, white paper (October 2018).

[2] M. Polese, L. Bonati, S. D'Oro, S. Basagni, T. Melodia, Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges (2022).

[3] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, J. J. Alcaraz, vrain: Deep learning based orchestration for computing and radio resources in vrans, IEEE Transactions on Mobile Computing (2020).

[4] D. Bega, A. Banchs, M. Gramaglia, X. Costa-Pérez, P. Rost, Cares: Computation-aware scheduling in virtualized radio access networks, IEEE Transactions on Wireless Communications 17 (12) (2018).

[5] P. Rost, I. Berberana, A. Maeder, H. Paul, V. Suryaprakash, M. Valenti, D. Wübben, A. Dekorsy, G. Fettweis, Benefits and challenges of virtualization in 5g radio access networks, IEEE Communications Magazine 53 (12) (2015) 75–82.

[6] S. Zhang, An overview of network slicing for 5g, IEEE Wireless Communications 26 (3) (2019) 111–117.

[7] R. Schmidt, N. Nikaein, Radio access network slicing system, Chapter book of Wiley 5G Ref, 2020.

[8] J. Huang, L. Gao, Wireless Network Pricing, Vol. 6, 2013.

[9] I. Jang, D. Suh, S. Pack, G. Dán, Joint optimization of service function placement and flow distribution for service function chaining, IEEE Journal on Selected Areas in Communications 35 (11) (2017) 2532–2541.

[10] N. Zhang, Y.-F. Liu, H. Farmanbar, T.-H. Chang, M. Hong, Z.-Q. Luo, Network slicing for service-oriented networks under resource constraints, IEEE Journal on Selected Areas in Communications 35 (11) (2017).

[11] S. Bhaumik, S. P. Chandrabose, M. K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, T. Woo, Cloudiq: A framework for processing base stations in a data center, Mobicom '12, p. 125–136.

[12] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, D. J. Leith, Srslte: An open-source platform for lte evolution and experimentation, in: WiNTECH, Association for Computing Machinery, 2016, p. 25–32.

[13] K. C. Garikipati, K. Fawaz, K. G. Shin, Rt-opex: Flexible scheduling for cloud-ran processing, CoNEXT '16, Association for Computing Machinery, p. 267–280.

[14] H. Khedher, S. Hoteit, P. Brown, R. Krishnaswamy, W. Diego, V. Vèque, Processing time evaluation and prediction in cloud-ran, in: IEEE International Conference on Communications (ICC), 2019, pp. 1–6.

[15] P.-C. Lin, S.-L. Huang, Performance profiling of cloud radio access networks using openairinterface, in: IEEE Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2018, pp. 454–458.

[16] T. X. Tran, A. Younis, D. Pompili, Understanding the computational requirements of virtualized baseband units using a programmable cloud radio access network testbed, in: IEEE International Conference on Autonomic Computing (ICAC), IEEE, 2017, pp. 221–226.

[17] S. Khatibi, K. Shah, M. Roshdi, Modelling of computational resources for 5g ran, in: 2018 European Conference on Networks and Communications (EuCNC), 2018. doi:10.1109/EuCNC.2018.8442563.

[18] M. C. Valenti, S. Talarico, P. Rost, The role of computational outage in dense cloud-based centralized radio access networks, in: 2014 IEEE Global Communications Conference, pp. 1466–1472.

[19] A. Garcia-Saavedra, X. Costa-Perez, D. J. Leith, G. Iosifidis, Fluidran: Optimized vran/mec orchestration, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, 2018, pp. 2366–2374.

[20] S. Tripathi, C. Puligheddu, C. F. Chiasserini, F. Mungari, A context-aware radio resource management in heterogeneous virtual rans, IEEE Transactions on Cognitive Communications and Networking (2021).

[21] G. Garcia-Aviles, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, P. Serrano, A. Banchs, Nuberu: Reliable ran virtualization in shared platforms, in: Proceedings of the 27th Annual International Conference on Mobile Computing and Networking, MobiCom '21, 2021.

[22] P. Rost, A. Maeder, M. C. Valenti, S. Talarico, Computationally aware sum-rate optimal scheduling for centralized radio access networks, in: 2015 IEEE Global Communications Conference (GLOBECOM), 2015.

[23] P. Rost, S. Talarico, M. C. Valenti, The complexity–rate tradeoff of centralized radio access networks, IEEE Transactions on Wireless Communications 14 (11) (2015) 6164–6176.

[24] K. Wang, X. Yu, W. Lin, Z. Deng, X. Liu, Computing aware scheduling in mobile edge computing system, Wireless Networks.

[25] H. Halabian, Distributed resource allocation optimization in 5g virtualized networks, IEEE Journal on Selected Areas in Communications 37 (3) (2019) 627–642.

[26] R. Wen, G. Feng, J. Tang, T. Q. S. Quek, G. Wang, W. Tan, S. Qin, On robustness of network slicing for next-generation mobile networks, IEEE Transactions on Communications 67 (1) (2019) 430–444.

[27] M. Richart, J. Baliosian, J. Serrat, J.-L. Gorricho, Resource slicing in virtual wireless networks: A survey, IEEE Transactions on Network and Service Management 13 (3) (2016) 462–476.

[28] R. Su, D. Zhang, R. Venkatesan, Z. Gong, C. Li, F. Ding, F. Jiang, Z. Zhu, Resource allocation for network slicing in 5g telecommunication networks: A survey of principles and models, IEEE Network 33 (6) (2019) 172–179. doi:10.1109/MNET.2019.1900024.

[29] P. Caballero, A. Banchs, G. de Veciana, X. Costa-Pérez, Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads, IEEE/ACM Transactions on Networking 25 (5) (2017) 3044–3058.

[30] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, N. Sastry, O. Holland, S. Tayade, B. Han, D. Bega, D. Aziz, H. Bakker, Network slicing to enable scalability and flexibility in 5g mobile networks, IEEE Communications Magazine 55 (5) (2017).

[31] G. Wang, G. Feng, W. Tan, S. Qin, R. Wen, S. Sun, Resource allocation for network slices in 5g with network resource pricing, in: GLOBECOM 2017 - 2017 IEEE Global Communications Conference, 2017.

[32] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, M.-J. Tsai, Deploying chains of virtual network functions: On the relation between link and server usage, in: IEEE INFOCOM 2016.

[33] A. K. Bairagi, M. S. Munir, M. Alsenwi, N. H. Tran, S. S. Alshamrani, M. Masud, Z. Han, C. S. Hong, Coexistence mechanism between embb and urllc in 5g wireless networks, IEEE Transactions on Communications 69 (3) (2021).

[34] K. Boutiba, A. Ksentini, B. Brik, Y. Challal, A. Balla, Nrflex: Enforcing network slicing in 5g new radio, Computer Communications (181) (2022).

[35] P. Yang, X. Xi, T. Q. S. Quek, J. Chen, X. Cao, D. Wu, How should i orchestrate resources of my slices for bursty urllc service provision?, IEEE Transactions on Communications 69 (2) (2021).

[36] B. Ojaghi, F. Adelantado, A. Antonopoulos, C. Verikoukis, Slicedran: Service-aware network slicing framework for 5g radio access networks, IEEE Systems Journal 16 (2) (2022).

[37] S. Pramanik, A. Ksentini, F. Chiasserini, C. Characterizing the computational and memory requirements of virtual rans, in: 2022 17th Wireless On-Demand Network Systems and Services Conference (WONS), 2022, pp. 1–8.

[38] 3GPP, Study on new radio access technology physical layer aspects.

[39] H. Yang, K. Zheng, K. Zhang, J. Mei, Y. Qian, Ultra-reliable and low-latency communications for connected vehicles: Challenges and solutions, IEEE Network 34 (3) (2020) 92–100.

[40] M. Setayesh, S. Bahrami, V. W. Wong, Resource slicing for embb and urllc services in radio access network using hierarchical deep learning, IEEE Transactions on Wireless Communications (2022).

[41] S. Bakri, P. A. Frangoudis, A. Ksentini, M. Bouaziz, Data-driven ran slicing mechanisms for 5g and beyond, IEEE Transactions on Network and Service Management 18 (4) (2021) 4654–4668.

[42] L. Kleinrock, Theory, Volume 1, Queueing Systems, Wiley-Interscience, USA, 1975.

[43] https://www.gurobi.com/.