Optimal Resource Allocation in C-RAN through DSP Computational Load Forecasting

Conference Paper · September 2020

DOI: 10.1109/PIMRC48278.2020.9217268

2 authors:

Armin Okić
Politecnico di Milano
9 PUBLICATIONS **45** CITATIONS

SEE PROFILE

Alessandro Redondi
Politecnico di Milano
104 PUBLICATIONS **1,817** CITATIONS

SEE PROFILE

Optimal Resource Allocation in C-RAN through DSP Computational Load Forecasting

Armin Okic and Alessandro E. C. Redondi
Dip. Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Email: armin.okic@polimi.it, alessandroenrico.redondi@polimi.it

***Abstract*—The Cloud-RAN (C-RAN) paradigm is envisioned to increase the efficiency of future mobile networks by moving the computational resources needed at the Remote Radio Heads (RRH) to the cloud infrastructure. In this work, we provide a framework that optimizes the number of allocated virtual resources by considering both the computational requirements of the RRH and the Quality of Service of users, which could experience loss of service due to reassociations between the RRH and the virtual machines. The provided optimization framework is supported by data coming from a real mobile network of a middle-sized European city, which provides an estimate for the computational loads coming from the RRH. We evaluate the performance of the framework in different scenarios, analyzing the impact of different forecasting algorithms as well as different look-ahead intervals for the predictions (short-term / long-term). The results obtained by our framework can be used to assist network operators in the optimization of C-RAN resources and shed some light on the interplay between forecasting errors and overall performance.**

***Index Terms*—forecasting DSP load, decoding, DSP, OAI, C-RAN**

I. Introduction

The emerging 5G-and-Beyond networks will strongly rely on the concept of Network Function Virtualization (NFV), enabling network operators to move all network functions, pertaining to both Core and Radio Access Network (RAN), into the Cloud. Indeed, the so-called Cloud-RAN (C-RAN) architecture envisions the aggregation of computational resources of all Base Stations (BS) in a centralized baseband unit (BBU) pool, which is connected to a densely distributed set of Remote Radio Heads (RRH) through high capacity links [1]. Such an approach is able to optimally allocate computational resources in a dynamic way to the individual RRH, whose hardware design is much simpler than legacy LTE eNodeBs. Compared to traditional architectures, the C-RAN approach is envisioned to lower overall hardware costs and energy consumption, at the same time increasing spectral efficiency and network resource utilization [2]–[5].

However, the actual realization of the C-RAN vision is not without its challenges: first, the complexity of the front-haul network (the network portion connecting the RRH with the BBU pool) is increasing due to the higher and higher bandwidth requirements. For the same reason, the time requirements of the Digital Signal Processing (DSP) functions (e.g., frame (de)modulation, (de)coding, and IFFT/FFT) to be implemented at the BBU are tighter and tighter. It follows that performing a paradigm shift in which such DSP functions are executed on General Purpose Processing hardware (GPP) in the Cloud, rather than on specialized hardware, is a complex operation which may be successfully completed only by (i) properly characterizing the computational requirements needed at the BBU and (ii) being able to accurately forecast them so that optimal dynamic reallocation (up/down scaling) of the virtual resources can be accomplished.

This paper focuses exactly on such two aspects: first, the computational load required for executing the decoding functions of a real cellular network is characterized, starting from a dataset of measurements related to the radio resources such as Modulation and Coding Scheme (MCS) and used Physical Resource Blocks (PRB). The characterization is performed relying on the Open Air Interface (OAI) simulator, which allows us to accurately evaluate the computational load needed for frame decoding. Then, the obtained computational load requirements are predicted using different forecasting models and prediction look-ahead intervals, in order to thoroughly characterize the prediction error. Finally, the results of these two steps are used to cast a robust optimization problem which minimizes the number of BBU resources needed (e.g., the total cost for the network operator), while ensuring enough computational resources for in-time decoding. The problem is solved over multiple time intervals, also taking into account possible re-associations between RRHs and BBU pools, which could possibly impact the users' Quality of Service. The resulting framework can be used to assist network operators in the optimization of C-RAN resources, also giving insights on the interplay between forecasting look-ahead intervals, prediction errors, and overall performance.

The remainder of this paper is organized as follows: Section II reviews related works in the literature; Section III gives an overview of the system model, while the optimization problem is formulated in Section IV. The characterization of computational load and related forecasting insights are given in Section V and Section VI respectively. Section VII reports on the performance results obtained and Section VIII concludes the paper and highlights future research directions.

II. Related work

Many recent works discuss the possibility of executing network-related DSP functionalities in the cloud, according to the C-RAN vision [3]. For what concerns frame decoding in particular, some works focus primarily on obtaining speeded-up implementations through the use of GPU and multicore

CPUs [6] or parallel architectures [7], which can be utilized in Cloud scenarios. At the same time, several works analyze and emphasize the computational complexity of frame decoding, which is directly influenced by several network-related parameters, such as the Moduling and Coding Scheme (MCS), the number of Physical Resource Blocks (PRB) to be processed at the same time, the current Signal-to-Noise Ratio (SNR), as well as Cloud-related parameters such as the CPU frequency and number of cores of the particular virtual machine used for decoding. The work in [8] gives an excellent review of the related work in the area of C-RAN complexity requirements characterization.

In order to perform realistic, repeatable and scalable experiments in such a scenario, Nikaein et. al propose Open Air Interface (OAI), an open-source reference software implementation[1] of 3GPP-compliant LTE/LTE-A systems [9]. The framework has been used in several works related to modeling and analysis of the computational requirements of DSP decoding functionalities in Cloud environments. As an example, [10] proposes the CloudIQ resource management framework, where OAI is used to implement and demonstrate such a solution in a realistic scenario. Authors show that C-RAN architectures can potentially save as much as 22% in computing resources compared to legacy approaches, by exploiting the variations in the processing load across base stations. In [11] virtual DSP functions are implemented on OAI, where different virtual technologies (e.g., virtual machines, containers) are compared in terms of total computing times for different values of MCS and PRB. It is demonstrated that the processing requirements are dominated by uplink decoding and can be estimated accurately as a function of PRB, MCS and the virtualization environment. The OAI framework is also leveraged in [12] to characterize the computation energy consumed by a BBU pool. Consequently, authors cast a resource allocation problem to minimize the number of active virtual machines and therefore obtain energy savings.

Several works are focused on optimizing the association between RRHs and the BBU pool with the goal of either reducing power consumption [13]–[15] or the total system cost [16]. In [13] RRH DSP requirements from two template cells (business/residential areas) of a real mobile network are split in tasks (decoding, modulation, FFT/IFFT) and each task is allocated to a different BBU so that the total power consumption is minimized. The problem is solved through a simulated annealing heuristic, showing power consumption savings between 5% and 20% compared to a static, non-virtualized architecture. In [16], authors focus on minimizing the total system cost, letting each UE to connect to multiple VMs in the BBU pool and considering limited fronthaul capacity. Each VM in the system is modeled as a FIFO queue, and the resulting problem is solved optimally with efficient search algorithms. However, all system parameters are set to arbitrary values, and no realistic datasets are used. Finally, Boulos et al. in [15] focus on RRH-BBU association optimization. The problem is formulated as a bi-objective problem, minimizing both power consumption and, similarly to our work, the total number of RRH re-associations to a new BBU. The problem is solved using a heuristic derived from the bin-packing problem literature. Again, simulation parameters are chosen arbitrarily, without leveraging realistic network datasets. To the best of our knowledge, this is the first work that studies the impact of DSP computational load forecasting on the optimal DSP resource allocation framework performed on a realistic LTE network dataset.

[1]https://www.openairinterface.org/

III. Problem overview

We consider a C-RAN network consisting of N RRHs, connected to a BBU pool formed by M virtual machines (VM). Each VM is responsible for executing the DSP functions of one or more associated RRHs, and we assume that an RRH may be connected to only one VM (e.g., $M \leq N$). At any instant, only a subset of $m \leq M$ VMs is active, providing enough computational resources to serve all RRHs. It is well known that among the different DSP functions, frame decoding is the most intensive one [11]. In this paper, we, therefore, assume that the requirements of each RRH are dominated by the decoding operation, whose computational complexity is determined by the number of used PRBs, the MCS distribution (i.e., the number of bits carried by each PRB) and the SNR. Furthermore, we treat the scheduling algorithm controlling the RRH (i.e., deciding how many PRBs to allocate and the corresponding MCS distribution) as a black box and we assume that our framework is able to observe only the output of such a black box. This scenario closely reflects the knowledge available at the operator side, which generally cannot modify the operational details of the scheduler algorithm, but it is able to observe the network KPIs resulting from its use.

We assume that the network operator relies on a third-party platform for managing and running the VMs, paying a price for the service which depends on how many VMs are active and for how long. The main objective of the operator is to minimize the total cost for the virtualization service, which requires to forecast the computational requirements of all the RRHs in order to activate/deactivate VMs accordingly. The process is subject to two main constraints, both related to the Quality of Service perceived by users of the network:

1) *In-time decoding*: frame decoding for each RRH must be completed by the BBU within stringent time requirements (the typical HARQ loop lasts a few ms in LTE [11], [17]). Any delay in the process due to under provisioning of the virtual resources may lead to the expiration of the corresponding timeouts and triggers of frame retransmissions at the user side, thus decreasing its QoS.
2) *BBU-RRH reassociation*: upon sudden peaks in the computational requirements or any changes in the number of active VMs, some of the RRHs will need to re-associate to a new VM in the BBU pool. Depending on how such a reassociation is handled, the process may cause all users connected to the RRH to experience a loss of QoS. Such an issue should be considered by the optimization framework.

IV. Problem Formulation

The problem described in the previous section can be formalized as a bin packing problem. Time is divided into discrete epochs of arbitrary length: at each epoch t the computational requirements of all RRHs are fit to the smallest possible number of VMs.

A. Decision variables

Let $x_{i,j,t}$, $1 \leq i \leq N$, $1 \leq j \leq M$, be a binary variable defined as:

$$x_{i,j,t} = \begin{cases} 1, & \text{if the } i\text{-th RRH is associated to the } j\text{-th VM} \\ & \text{during epoch } t, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Since each RRH must be associated to exactly one VM in each epoch, we have that:

$$\sum_{j=1}^{M} x_{i,j,t} = 1, \quad \forall i, t. \tag{2}$$

At each epoch, a VM in the BBU pool can be active (i.e., associated to at least one RRH) or inactive. Let y_j, t be a binary variable tracking the state of each VM in each epoch, defined as:

$$y_{j,t} = \begin{cases} 1, & \text{if } \sum_{i=1}^{N} x_{i,j,t} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

It is easy to show that (3) can be rewritten using the two following linear constraints:

$$\sum_{i=1}^{N} x_{i,j,t} \leq y_{j,t} * N, \quad \forall j, t, \tag{4}$$

$$\sum_{i=1}^{N} x_{i,j,t} \geq y_{j,t}, \quad \forall j, t. \tag{5}$$

B. QoS-related constraints

At the beginning of each epoch t, the operator forecasts the computational load $c_{i,t}$ (expressed in the number of CPU cycles) required for decoding frames coming from the i-th RRH and uses it to allocate VMs properly. As aforementioned, two constraints related to the QoS perceived by users should be considered:

1) *In-time decoding:* frame decoding has strict time requirements, which must be satisfied in order to avoid QoS degradation. Let d be the deadline (in seconds) for decoding frames at the BBU pool, and b the computational budget (i.e., the CPU frequency) of each VM (in Hz). To ensure that all frames are decoded within the deadline by the BBU pool at epoch t, we may write:

$$\sum_{i=1}^{N} x_{i,j,t} * c_{i,t} \leq y_{j,t} * b * d \quad \forall j. \tag{6}$$

For simplicity, here we assume that all VMs have the same computational budget b, although the model can be easily generalized to the case where VMs have different CPU frequencies.

2) *RRH-BBU reassociation:* due to load variations, an RRH may be associated to different VMs between two consecutive epochs. During reassociation, all users connected to the RRH may experience a loss of QoS, e.g., due to forced handovers to other RRHs. Let $r_{i,j,t}$ be a binary variable defined as:

$$r_{i,j,t} = \begin{cases} 1, & \text{if } x_{i,j,t} \neq x_{i,j,t-1}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Again, $r_{i,j,t}$ can be formalized with the help of the following linear constraints:

$$x_{i,j,t} - x_{i,j,t-1} \leq r_{i,j,t}, \tag{8}$$

$$x_{i,j,t-1} - x_{i,j,t} \leq r_{i,j,t}. \tag{9}$$

Rather than expressing a direct constraint, RRH-BBU reassociations are managed as a penalty term in the objective function, as explained next.

C. Objective function

Let p be the per-VM price paid by the operator to the third-party for managing the virtualization service in each epoch (e.g., hourly or every 15 minutes). We assume that the operator is able to forecast future RRH loads in a time window T composed of several epochs (i.e., $t = 1 \dots T$). The operator's goal is to minimize the following objective function:

$$J = \sum_{t=1}^{T} \sum_{j=1}^{M} p * y_{j,t} + \alpha \sum_{t=2}^{T} \frac{1}{2} \sum_{j=1}^{M} \sum_{i=1}^{N} r_{i,j,t}, \tag{10}$$

where the first term captures the total cost for running the VMs, the second term is a penalty introduced to limit the number of RRH-BBU reassociations and the variable α is a scaling factor which could be used for balancing the two terms and prioritizing one over the other. The term $\frac{1}{2}$ is added to avoid counting twice a reassociation of an RRH from one VM to another. Note that here we assume the cost p to be time-invariant, although the model could be easily generalized to the case where p changes with time.

V. Load Characterization

A. Dataset

In order to characterize the loads $c_{i,t}$, we leverage a dataset containing measurements from 443 LTE base stations deployed in a middle-sized European city, all working at a frequency of 2100 MHz and with a channel bandwidth of 10 MHz. For each base station, two weeks of data sampled in 15 minutes are available. We focus on two specific measurements, namely the MCS and PRB utilization uplink distribution. The former reports the distribution of uplink MCS assigned by the scheduler in each 15-minute interval, while the latter tracks the distribution of used uplink radio resources (i.e., how many of the 50 PRBs available in the 10 MHz channel are allocated to users) in each 15-minute interval. Such distributions are

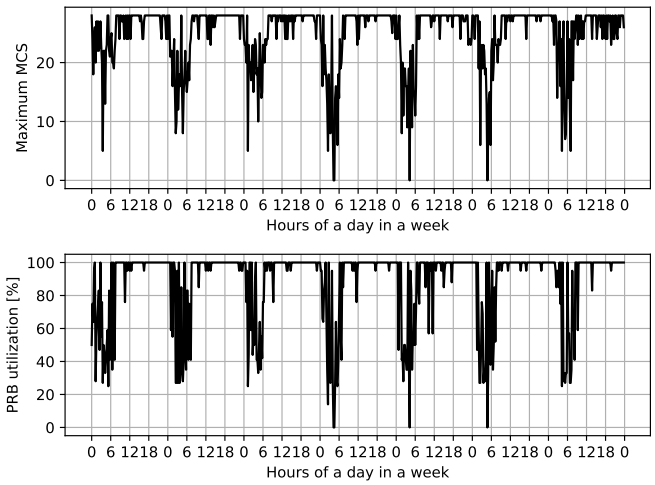

Fig. 1: Maximum MCS (top) and PRB (bottom) utilization profiles of one base station in the dataset over a full week.

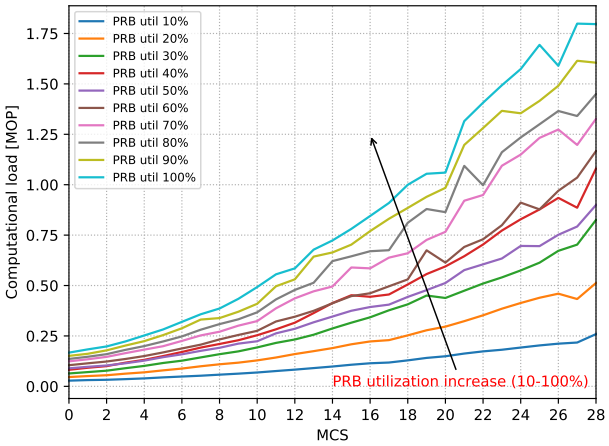

Fig. 2: Computational load generated with OAI simulation for channel bandwidth 10 MHz over all MCS and for different PRB utilization.

pre-processed in order to obtain two single-valued time series, by picking the maximum MCS and PRB utilization values that occurred within each time interval. We prefer such a conservative approach, rather than using the distribution mean, in order to ensure that the worst-case scenario in terms of the computational load is considered. Fig. 1 shows the MCS and PRB traces for one base station in the dataset relative to one week.

B. Emulation with OAI

The MCS and PRB traces are input to the Open Air Interface (OAI) platform, which allows emulating the full 3GPP LTE protocol stack, including highly optimized baseband processing functionalities such as turbo decoding. In particular, we leverage the *ulsim* tool of OAI, which emulates the Physical Uplink Shared Channel (PUSCH) decoding pipeline at the eNodeB. The tool allows to specify several input parameters, including the sub-frame MCS, the current PRB load (how many resource blocks are allocated to the user), channel parameters such as bandwidth and SNR, as well as other options (e.g., the number of iterations of the turbo decoder). For a given input configuration, the tool emulates the decoding process and keeps track of the corresponding CPU time t. Knowing the CPU frequency f, it is possible to express the computational load for decoding as $c = f * t$. Fig. 2 shows the computation load in Million Operations (MOP), obtained with OAI at different MCS and PRB working points, averaged over 1000 sub-frames. Due to the lack of fine-grained SNR measurements in our dataset, simulations were run setting the lowest possible SNR value allowed for each MCS configuration, which corresponds to the working point with the highest computational load, as demonstrated in [8], [18]. All tests were performed on a single-core Intel(R) Xeon(R) CPU E5-1660 v3 @ 3.00GHz, 16 GB RAM, with Ubuntu 16.04 OS, using an AWGN channel with a bandwidth of 10 MHz (as in our dataset), setting the number of iterations of the decoder to 4.

C. Estimation of RRH computational load

The curves in Fig. 2 are used to compute a load profile for each base station in the dataset, starting from the MCS and PRB traces described in Section V-A. We observe that different base stations are characterized by different load profiles. Therefore we perform k-mean clustering over the traces, choosing the best k using both Silhouette and Davies-Bouldin cluster quality indexes. We reveal the existence of 3 distinct clusters, shown in Fig. 3, where each cluster centroid is represented by the red bold line. As it can be observed, all clusters are characterized by the day/night fluctuation typical of mobile traffic. However, the first cluster is mainly composed of base stations characterized by a constant, high load throughout the whole week, with a very small decrease during nights. In the second cluster, the load during nights is approximately halved, while the third cluster is composed of cells that are completely offloaded during nights. Among the 443 base stations in the dataset, 48% belong to the first cluster, 9% to the second, and 43% to the third. We refer to these three clusters as High Load (H), Moderate Load (M) and Low Load (L), respectively.

VI. Load Forecasting

A. Look-ahead interval

The optimization problem proposed in Section IV needs as input the loads $c_{i,t}$ for the entire time window T. We assume that load forecasting is performed up to a certain prediction horizon, which we refer to as look-ahead interval L. As an example, if each epoch t lasts 15 minutes and $L = 4$, forecasting is performed to obtain load values up to 1 hour in the future. Note that L controls how many times the optimization problem needs to be solved. When $L = T$ the forecast algorithms predict all values $c_{i,t}$ and only one instance of the problem is solved. When $L < T$, the optimization problem is solved $\lceil T/L \rceil$ times, each time considering only the L available predicted loads. In this latter case, we substitute the

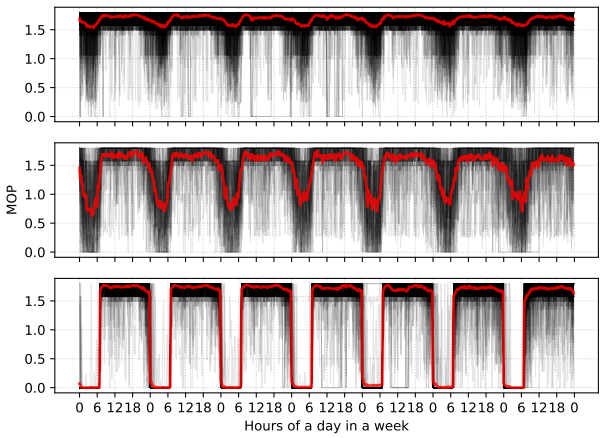

Fig. 3: Clustered computational load profiles of base-stations. The red dashed line illustrates cluster centroid.

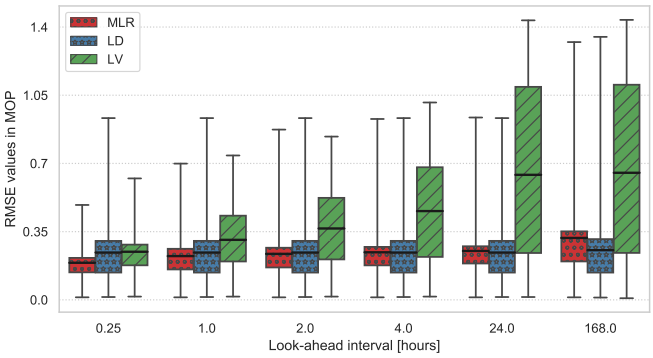

Fig. 4: Distributions of RMSE values for different forecasting methods over different look-ahead intervals.

term T in (10) with L, and the variables $r_{i,j,0}$ at the beginning of each look-ahead interval except for the very first are stored from the previous interval and passed as input parameters.

B. Forecasting algorithms

Let t be the epoch at which forecasts need to be computed. We consider three different options for computing the predicted load samples $c_{i,l},\ l = t+1, \ldots, t+L$:

- *Last value (LV)*: predictions are obtained copying the last available load sample $c_{i,t}$, that is $c_{i,l} = c_{i,t}$.
- *Last day (LD)*: the future samples are obtained from the samples available from the last 24 hours, that is $c_{i,l} = c_{i,l-96}$ when each epoch lasts 15 minutes.
- *Multiple Linear Regression (MLR)*: the forecasts are obtained using a machine learning model trained for predicting the next sample starting from W past samples, that is $c_{i,l} = \beta_0 + \sum_{k=1}^{W} \beta_k \cdot c_{i,l-k}$. Note that only the sample $c_{i,t+1}$ is predicted from past samples, while all other forecasts up to L are predicted starting from already predicted values. The model parameters β_k and the value of W are chosen following a standard machine learning approach: the entire set of computational loads $c_{i,t}$ is divided into training (first week of data) and test set (second week). The model is trained and the parameters are chosen so as to minimize the training Root Mean Squared Error (RMSE). The value of W resulting from such a process is equal to 96 samples, corresponding to one day. Increasing values of W resulted in negligible improvements with a considerably higher amount of memory used, and were hence not considered.

C. Performance comparison

Fig. 4 shows the distributions of RMSE values for different forecasting approaches and look-ahead intervals. Each boxplot reports the average RMSE over all base stations in the dataset (the bold line), as well as the 25th and 75th percentile. Whiskers correspond to the minimum and maximum RMSE values. The following considerations can be done: (i) in general, as expected the RMSE increases as look-ahead interval increases; (ii) among all algorithms, MLR outperforms both LD and LV for small look-ahead intervals, not only in terms of average error but also in terms of maximum RMSE; (iii) for larger look-aheads (e.g. 4 hours) the performance of MLR and LD are similar on average, with MLR showing much lower variance at small look-aheads; (iv) the LV algorithm shows the worst performance among all methods. In the following section, we use such an error characterization to analyze the impact of using different algorithms and look-ahead intervals in the C-RAN optimization problem.

VII. Experiments

A. Setup

We evaluate the optimization framework considering 60 out of the 443 available base stations, randomly selected from the dataset according to the same cluster distribution reported in Section V-C. The time window T is set to one day, coinciding with the first day of the test week, and four look-ahead intervals are tested: 15 minutes, 1 hour, 2 hours, and 4 hours. The problem formulated in Section IV is implemented with Python using the Gurobi interface and solved on an Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz, 40 cores, 126 GBs of RAM, running Ubuntu 16.04 OS. In order to provide realistic values for our scenario, we use parameters derived from the popular Amazon EC2 cloud service. In particular, we set the budget value b to 10 MOP, which corresponds to a price p of 0.2016 dollars for every 15-minute occupation of the resources.

B. Benchmarks

As a first experiment we run the optimization problem setting $\alpha = 0$, that is without considering RRH-BBU reassociations, and assuming the availability of an *oracle* algorithm which is able to forecast the L future load values perfectly. This serves as a benchmark to evaluate the minimum VMs' cost paid by the operator in case of dynamic resource allocation. Fig. 5 compares such cost (in the number of allocated 15-minute instances) with two other benchmarks: (i) *static C-RAN*, where VMs are allocated in a static fashion through over-provisioning,

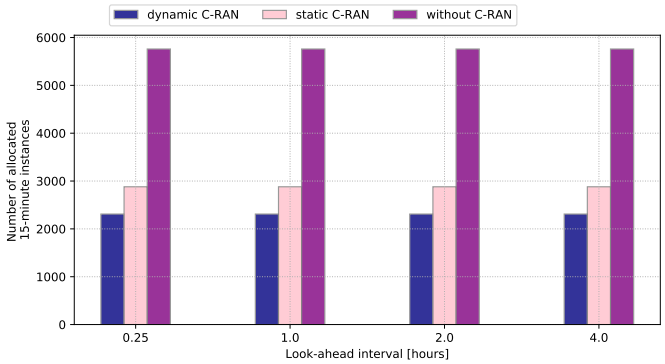

Fig. 5: Number of allocated 15-minute virtual instances for three benchmarks over different look-ahead intervals, without considering moving cost penalties.

considering the peak load of each RRH and letting RRHs to share the same VM instance; (ii) *without C-RAN*, representing a scenario in which each RRH is associated with one full VM instance, without the possibility of sharing resources among RRHs. As one can see from Fig. 5, dynamic C-RAN allocation is able to provide considerable cost savings, in the order of 25% compared to the static C-RAN case and 2.5 times better than without C-RAN. Note also that the look-ahead interval does not have any effect in this case, since i) the prediction returned by the oracle are not affected by the look-ahead interval and ii) reassociations are not contemplated, making the temporal dependency between adjacent epochs (i.e., the second term of (10)) irrelevant.

C. Numerical results

Next, we run the optimization problem with different values of α and considering several pairs of forecast algorithms and look-ahead intervals. Note that, as illustrated in Fig. 4, each pair is characterized by a different RMSE distribution. Since the forecasts are used in (6) as input to the optimization problem, it is important to leverage the knowledge of the RMSE distribution to avoid that the predicted load are underestimated, which would result in insufficient provisioning of VMs. We take here a worst-case robust optimization approach in which the forecasted loads $c_{i,t}$ are augmented with a value ϵ, which is set equal to the maximum RMSE observed for the particular algorithm/look-ahead interval pair under consideration. In this way, we ensure that the active VMs are always enough to satisfy the RRHs' requirements, even when the forecasts are affected by the maximum error. This satisfies the first QoS constraint introduced in Section III. The four graphs in Fig. 6 show the results obtained for different values of α in the set $\{0.1, 0.5, 1, 100\}$, that is gradually increasing the importance of minimizing the RRH-BBU reassociations. Each graph shows on the left side with solid bars the total VM cost (expressed in percentage increase over the dynamic C-RAN benchmark), while on the right side the number of reassociations (striped bars). We note that:

- A clear trade-off is visible between the two terms of the objective functions: small values of α keep the cost

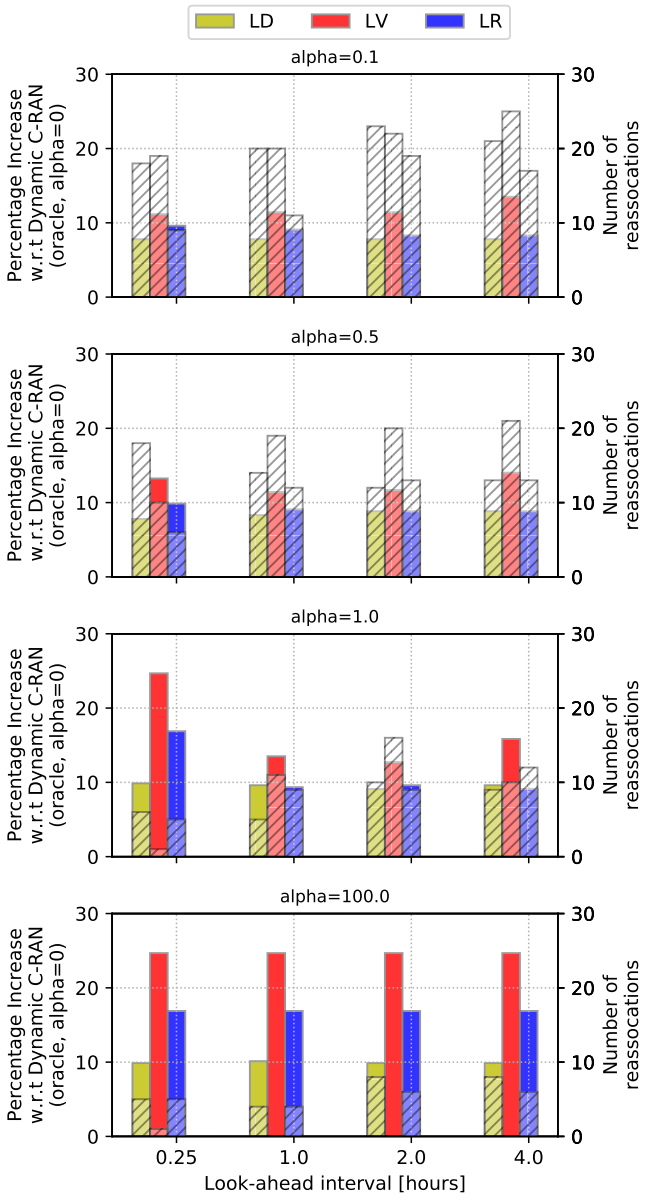

Fig. 6: Percentage increase of allocated 15-minute instances w.r.t. dynamic C-RAN benchmark, with number of re-associations for different moving costs, over different prediction methods and look-ahead intervals.

increase around 10% for all algorithm/look-ahead interval pairs, with an associated number of reassociations higher than 10 in most cases. Conversely, higher values of α are able to reduce considerably the number of reassociations at higher VM cost. An interesting case is given by the LV predictor (red bars) for $\alpha = 100$: the high error associated with the LV algorithm has the effect of greatly overestimating the RRH load requirements, making the solution similar to the static C-RAN scenario. In this case, no reassociations are performed.
- For low values of α, the optimal look-ahead interval changes according to the algorithm used. As an example,

Look-ahead interval (h)	No. of problem solved	No. of variables per problem	Total solving time (s)
0.25	96	544	22
1	24	2176	363
2	12	4352	1855
4	6	8704	25016

TABLE I: Solving times for different look-ahead intervals.

for $\alpha = 0.1$ and 0.5, the VM cost decreases as the look-ahead interval increases in the LD and LR cases. This does not hold for the LV case, for which the RMSE increases drastically as the look-ahead interval expands.

- For high values of α, the number of reassociations increases as the look-ahead interval increases for the LD and LR case, suggesting that short-term predictions should be preferred in place of long-term ones.
- In general, and as expected, no solution is able to decrease simultaneously both the number of reassociations and the VM cost. It is therefore the operator's duty to tune the α parameter according to the preferred scenario.

D. Problem complexity

As explained in Section VI-A, the look-ahead interval controls the number of optimization problems run in the entire interval T. Note that each problem is essentially a bin-packing problem, which is known to be NP-hard. Table I shows the solving times for $\alpha = 0.1$, in the dynamic C-RAN with oracle scenario and with 60 BS. As one can see, a higher look-ahead interval means higher solving time. This is promising for cases where short-term predictions provide better results than long-term ones, as solving multiple small instances of the problem is much more efficient than solving less larger. Tackling large instances of the problem, entailing hundreds of RRHs and with time intervals spanning over weeks, requires the development of specific heuristic algorithms, which are left as future work.

VIII. Conclusion and Future Works

In this paper we introduce a framework that exploits computational loads forecasting to optimally allocate VM instances in a C-RAN architecture, additionally taking into account the number of service interruptions for the final users due to RRH-BBU reassociations. The outcomes of this work can be used for various cases of 5G-and-Beyond networks in which DSP resources of RAN are virtualized and combined with dynamical algorithms for re-allocation of virtual instances according to the current network loads. Furthermore, in combination with slicing technologies, our approach could be applied differently for various slices since it allows different treatments of service interruptions, which could be defined by a network operator. Future works include (i) improving the prediction algorithms to obtain oracle-like forecasts and (ii) developing a heuristic algorithm to tackle larger instances of the problem, considering hundreds of base stations and larger look-ahead intervals.

Acknowledgment

This research has been partially supported by the H2020-MSCA-ITN-2016 framework under grant agreement number 722788 (SPOTLIGHT).

References

[1] Z. Zhu, P. Gupta, Q. Wang, S. Kalyanaraman, Y. Lin, H. Franke, and S. Sarangi, "Virtual base station pool: towards a wireless network cloud for radio access networks," in *Proceedings of the 8th ACM Intl. Conf. on computing frontiers*. ACM, 2011, p. 34.

[2] A. Checko, H. L. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. S. Berger, and L. Dittmann, "Cloud RAN for mobile networks—A technology overview," *IEEE Comm. surveys & tutorials*, vol. 17, no. 1, pp. 405–426, 2014.

[3] D. Wübben, P. Rost, J. Bartelt, M. Lalam, V. Savin, M. Gorgoglione, A. Dekorsy, and G. Fettweis, "Benefits and impact of cloud computing on 5G signal processing," *IEEE Signal Processing Mag.*, vol. 31, no. 6, pp. 35–44, 2014.

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

[5] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang, "Recent advances in cloud radio access networks: System architectures, key techniques, and open issues," *IEEE Comm. Surveys & Tutorials*, vol. 18, no. 3, pp. 2282–2308, 2016.

[6] M. Wu, G. Wang, B. Yin, C. Studer, and J. R. Cavallaro, "HSPA;/LTE-A turbo decoder on GPU and multicore CPU," in *2013 Asilomar Conf. on Signals, Systems and Computers*. IEEE, 2013, pp. 824–828.

[7] V. Q. Rodriguez and F. Guillemin, "Cloud-RAN modeling based on parallel processing," *IEEE Journal on Selected Areas in Comm.*, vol. 36, no. 3, pp. 457–468, 2018.

[8] P. Rost, S. Talarico, and M. C. Valenti, "The complexity–rate tradeoff of centralized radio access networks," *IEEE Trans. on Wireless Comm.*, vol. 14, no. 11, pp. 6164–6176, 2015.

[9] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Comm. Review*, vol. 44, no. 5, pp. 33–38, 2014.

[10] S. Bhaumik, S. P. Chandrabose, M. K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, and T. Woo, "CloudIQ: A framework for processing base stations in a data center," in *Proceedings of the 18th annual Intl. Conf. on Mobile computing and networking*. ACM, 2012, pp. 125–136.

[11] N. Nikaein, "Processing radio access network functions in the cloud: Critical issues and modeling," in *Proceedings of the 6th Intl. Workshop on Mobile Cloud Computing and Services*. ACM, 2015, pp. 36–43.

[12] A. Younis, T. X. Tran, and D. Pompili, "Bandwidth and energy-aware resource allocation for cloud radio access networks," *IEEE Trans. on Wireless Comm.*, vol. 17, no. 10, pp. 6487–6500, 2018.

[13] M. Qian, W. Hardjawana, J. Shi, and B. Vucetic, "Baseband processing units virtualization for cloud radio access networks," *IEEE Wireless Comm. Letters*, vol. 4, no. 2, pp. 189–192, 2015.

[14] K. Boulos, M. El Helou, and S. Lahoud, "RRH clustering in cloud radio access networks," in *2015 Intl. Conf. on Applied Research in Computer Science and Engineering (ICAR)*. IEEE, 2015, pp. 1–6.

[15] K. Boulos, M. El Helou, K. Khawam, M. Ibrahim, S. Martin, and H. Sawaya, "RRH clustering in cloud radio access networks with re-association consideration," in *2018 IEEE Wireless Comm. and Networking Conf. (WCNC)*. IEEE, 2018, pp. 1–6.

[16] J. Tang, W. P. Tay, T. Q. Quek, and B. Liang, "System cost minimization in cloud RAN with limited fronthaul capacity," *IEEE Trans. on Wireless Comm.*, vol. 16, no. 5, pp. 3371–3384, 2017.

[17] S. Khatibi, K. Shah, and M. Roshdi, "Modelling of Computational Resources for 5G RAN," in *2018 European Conf. on Networks and Comm. (EuCNC)*. IEEE, 2018, pp. 1–5.

[18] P. Rost, A. Maeder, M. C. Valenti, and S. Talarico, "Computationally aware sum-rate optimal scheduling for centralized radio access networks," in *2015 IEEE Global Comm. Conf. (GLOBECOM)*. IEEE, 2015, pp. 1–6.