IEEE INFOCOM 2021 - IEEE Conference on Computer Communications | 978-1-6654-0325-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/INFOCOM42981.2021.9488845

Bayesian Online Learning for Energy-Aware Resource Orchestration in Virtualized RANs

Jose A. Ayala-Romero*, Andres Garcia-Saavedra†, Xavier Costa-Perez†‡, George Iosifidis§
*School of Computer Science and Statistics, Trinity College Dublin, Ireland
†NEC Laboratories Europe, Germany
‡ i2CAT Foundation and ICREA, Spain
§Delft University of Technology, Netherlands

***Abstract*—Radio Access Network Virtualization (vRAN) will spearhead the quest towards supple radio stacks that adapt to heterogeneous infrastructure: from energy-constrained platforms deploying *cells-on-wheels* (e.g., drones) or battery-powered cells to *green* edge clouds. We perform an in-depth experimental analysis of the energy consumption of virtualized Base Stations (vBSs) and render two conclusions: (i) characterizing performance and power consumption is intricate as it depends on human behavior such as network load or user mobility; and (ii) there are *many* control policies and some of them have *non-linear and monotonic* relations with power and throughput. Driven by our experimental insights, we argue that *machine learning* holds the key for vBS control. We formulate two problems and two algorithms: (i) BP-vRAN, which uses Bayesian online learning to balance performance and energy consumption, and (ii) SBP-vRAN, which augments our Bayesian optimization approach with *safe* controls that maximize performance while respecting hard power constraints. We show that our approaches are *data-efficient* and have provably performance, which is paramount for carrier-grade vRANs. We demonstrate the convergence and flexibility of our approach and assess its performance using an experimental prototype.**

I. Introduction

Virtualization is considered today the most promising approach for bringing cellular networks up to speed with the demanding services they aspire to support [1]. The latest frontier in this endeavor is virtualizing the radio access network (vRAN) by turning the base stations (BSs) into fully-softwarized stacks that can be deployed in diverse platforms such as commodity servers, small embedded devices, or even moving nodes (*cells-on-wheels*) [2]. This paradigm shift is expected to offer the much-needed performance flexibility, facilitate the ongoing network densification, and reduce capital and operating expenses [3]. Hence, not surprisingly, it has spurred numerous industry efforts to build BS stacks [4], fully-open RANs [5], and even launch extensive field tests [6].

However, designing and operating vRANs is far from trivial, since the virtualized base stations (vBSs) differ significantly from their hardware-based counterparts. On the one hand, vBSs are more controllable as one can tune their parameters (transmission power, modulation schemes, etc.) in real time based on the network needs. On the other hand, their softwarization and diverse platforms render less predictable their performance and power consumption. The latter is particularly important both for economic reasons but also because the vBSs, most often, operate under tight energy budgets [7]. Hence, traditional radio control policies run the risk of under-utilizing this new type of BSs, or rendering vRANs economically unsustainable. Clearly, in order to unleash the full potential of vRANs we need to answer two key questions: *(i) what is the performance and energy consumption profile of vBSs?* and *(ii) how can we optimize their operation using an adaptive and platform-oblivious approach?* In this paper we take a hybrid experimental/theoretical approach to address systematically these questions.

We first analyze experimentally the vBSs operation using different platforms and scenarios. Our results shed light on the relationship between performance (throughput), power consumption, and vBS controls such as the modulation and coding schemes (MCS) and spectrum allocation. For instance, we find that the baseband unit (BBU) consumes power comparable to wireless transmissions, and the vBS power and throughput are affected by the configurations in a non-linear/non-monotonic fashion. These results depend also on the hosting platform. Moreover, we observe that uplink (UL)-related computations consume more power and are more sensitive on MCS and SNR variations, than the respective downlink (DL) computations; a finding attributed to the heavier UL decoding. Our analysis is centered on energy, which is the bottleneck vBS resource that affects both its computations and transmissions.

The take-away message from these extensive measurements (details in Sec. III) is that, *unlike legacy BSs*, virtualized BSs have a complex, poly-parametric, and platform-dependent performance and power consumption profile; and this renders traditional control policies inefficient. Therefore, we propose a powerful machine learning framework that learns on-the-fly the vBS profiles and selects their optimal configurations based on the network needs and power availability. We formulate two energy-aware vBS control problems and design novel algorithms to solve them: (i) BP-vRAN, which finds the Pareto-optimal trade-off between performance and power consumption; and (ii) SBP-vRAN, which maximizes performance subject to *hard* constraints on the power source. The former allows operators to balance performance and power expenses, while the latter is crucial for vBS running on power-constrained platforms, e.g., power-over-ethernet (PoE) cells.

Our algorithms are strongly founded on Bayesian online optimization theory [8] and Gaussian Processes [9]. The GPs model the behavior of the vBS in terms of performance and power consumption, using the collected measurements in real

978-1-6654-0325-2/21/$31.00 ©2021 IEEE

time. Accordingly, we use a contextual bandit approach to intelligently *explore* the space of vBS configurations, and *exploit* the best ones for each context, namely UL/DL traffic load and SNR patterns. The result is a non-parametric algorithmic framework that makes minimal assumptions about the system, adapts to user needs and network conditions, and *provably* maximizes performance. Furthermore, drawing ideas from *safe* Bayesian optimization [10], [11], our SBP-vRAN algorithm ensures that the vBS power constraints are not violated even during exploration, hence enabling the vBS deployment on energy-constrained platforms. By its design, this framework outperforms other approaches requiring knowledge of the vBS functions [12] or offline data to approximate them [13], and adaptive techniques that do not offer performance guarantees or have strict assumptions [14], [15] (see Sec. II).

Finally, we performed an extensive evaluation in a customized testbed, virtualizing srsLTE [4] as vBs, and several tools to measure in real time the power consumption. We verified that both algorithms converge and find the optimal vBS configuration in a variety of scenarios. To that end, we also proposed and evaluated practical enhancements that expedite the algorithms' convergence. Using real traffic traces, we show, step-by-step, how our framework explores the configurations, and how it refrains from violating the power constraints when necessary.

To summarize, the main contributions of this paper are:

- We perform a thorough experimental study of the power consumption and performance of vBSs upon different scenarios and platforms, revealing their complex profile/operation.
- We develop a non-parametric fully-adaptive learning framework to optimize on-the-fly the vBS operation; and we propose two algorithms for tackling two important problems: *(i)* BP-vRAN, which balances performance and cost, useful for unconstrained computing platforms; and *(ii)* SBP-vRAN, which maximizes performance subject to a hard power consumption constraints.
- Finally, we assess the performance of our framework using realistic contexts (network loads and channel dynamics). Our findings verify that it constitutes a strong candidate for the next-generation automated/zero-touch vBS control solution.

Our software implementation of BP-vRAN and SBP-vRAN and the dataset used in our paper are publicly available to ease reproducibility and foster future research in this area.

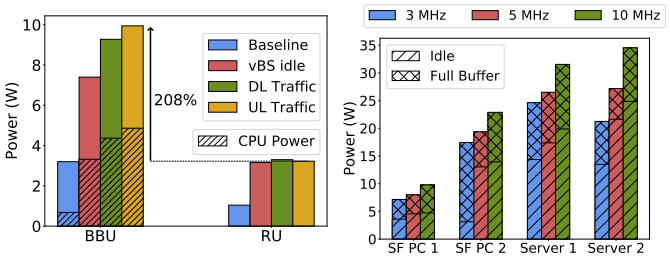
II. Related Work

Resource Orchestration in Networks can be classified to: *(i)* studies that use models which relate control variables to performance; *(ii)* model-free approaches that rely on offline training data; and *(iii)* online learning techniques. Interesting examples in *(i)* include [16] which performs rate control to maximize throughput; [12] that selects also the MCS and airtime; and [17] that adapts to traffic. Nonetheless, such models are often unavailable and platform-dependent. On the other hand, model-free approaches employ machine learning to approximate the performance functions [18] and are used in slicing [19], throughput forecasting [13], etc. Their efficacy is remarkable as long as there are enough training data. Otherwise, we need to employ online learning that has been recently used, for instance, to configure video analytic systems [28] and minimize the power consumption and interference among BSs [22]. Similarly, online convex optimization is used for cloud and IoT resource orchestration [20], [21], but requires convex functions; a condition not satisfied here. Another approach is reinforcement learning (RL), used in spectrum management [14], network diagnostics [23], interference coordination [24], and SDN control [25], among others. However, RL suffers from the curse of dimensionality, and lacks convergence guarantees.

Similarly to RL, contextual bandits have been recently employed to adjust video streaming rates [27]; configure BS parameters (e.g., handover thresholds) [29], [30]; assign CPU time to virtualized BSs [15]; and control mmWave networks [31], [32]. Here, instead, we combine Gaussian Processes [9] and contextual bandit algorithms [26] to build a *data-efficient* Bayesian optimization framework [8] with *convergence guarantees*. Our approach captures the non-trivial multimodal correlations of configurations (revealed by our experiments) through Gaussian Processes, and use these perpetually-updated functions to sample the decision space. Our work draws from the seminal CGP-UCB algorithm [26] which is extended to include vRAN-specific context, to optimize throughput and power costs, and to satisfy hard power constraints.

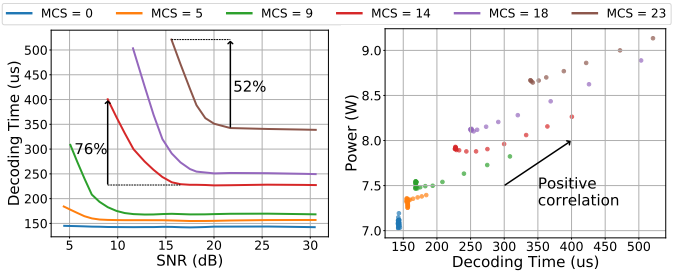
Experimental Studies of vBS. The early work of [34] studied the cost savings when pooling the processing operations of BSs, and [35] proposed a vRAN architecture that reduces processing load by 30%. Other studies considered the effect of MCS, bandwidth, and SNR on Baseband Unit (BBU) computing load [36], [37]. Using an OAI simulator [38] models the processing time for different configurations, and [15] presented measurements with srsLTE for the impact of traffic. Our analysis builds on these important works and further measures the impact of new context parameters and radio schedulers on throughput, the coupling of uplink and downlink, and the vBS power consumption.

Studies of power consumption in legacy BSs focus on the effect of power amplifier, RF output, and baseband processing. The *EARTH* model [39] relates the RF output power to supplied power; and [40] studies the effect of bandwidth. The works [41], [42] proposed models for macro and micro BSs, and [43] studied how the packet length affects the CPU power. Models accounting for various BS components are presented in [44], [45]. Alas, for small vBSs other configuration parameters are equally important. Previous works focusing on vBS include [46] which considered the effect of CPU cores/speed, assuming a linear relation of traffic with computing load. This assumption is not always valid as our measurements and previous studies showed [36]. Importantly, the impact of hardware/software platform on these metrics cannot be captured in predetermined models. We overcome this obstacle by building models on-the-fly using the observed data.



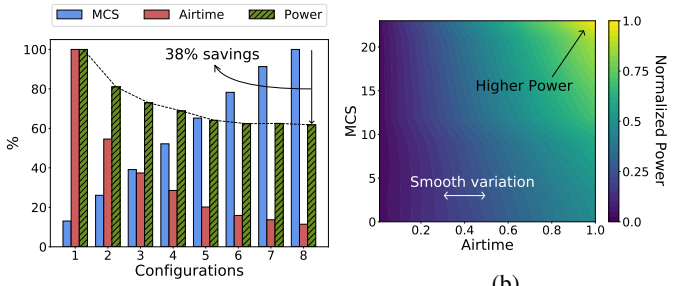
(a) (b)

Fig. 1: **(a)**: Comparison of power consumption at: the BBU (Intel NUC i7-8559U@2.70GHz), the BBU's CPU, and the RU (an USRP SDR), with 20Mbps DL and UL traffic. **(b)**: Consumed power over the baseline for different radio bandwidths and hardware platforms. SF PC 1: Intel NUC i7-8559U@2.70GHz; SF PC 2: Intel NUC i7-8650U@1.90GHz; Server 1: Dell XPS 8900 i7-6700@3.40GHz; Server 2: Dell Aurora R5 i7-9700@3.00GHz.



(a) (b)

Fig. 2: vBS over SF PC 1 at full UL buffer. **(a)**: UL decoding time as a function of SNR and different MCS values. **(b)**: Power consumption as a function of the decoder performance (high correlation).



(a) (b)

Fig. 3: **(a)**: 8 combinations of normalized MCS and airtime providing 2.6Mbps in UL, and its associated power (idle mode power is subtracted). **(b)**: Normalized power consumption at the BBU over baseline for full buffer UL transmissions and high SNR, as a function of MCS and airtime.

III. Preliminary Experimental Analysis

We performed exhaustive experiments using an srsLTE-based vBS testbed [4]; details in Section VI-A. We first present results that motive the problem and our solution approach.

• BBU/CPU Power Cost & Impact of Platform. The first important finding is that the power consumption associated with BBU processing is *comparable* to the RF chain's transmission power. This result is consistent with previous studies; e.g., [39] estimated that 40% of a femtocell's power consumption is due to BBU. Fig. 1a dissects the power consumption of a vBS deployed over a small factor (SF) PC into the share responsible by (i) the BBUs CPUs[1], (ii) the BBUs cloud platform *except the CPUs*, and (iii) the actual radio unit (RU) deployed over an USRP software-defined radio (SDR). We measure the power consumption for four scenarios: (i) the vBS is not deployed (baseline), (ii) the vBS is deployed with an idle user attached (vBS idle), (iii) the vBS is transmitting 20Mbps of downlink (DL) traffic, and (iv) the user is transmitting 20Mbps of uplink (UL) traffic to vBS.

Excluding the baseline scenario, the CPU power cost alone is, on average, 29% larger than that of the RU, while the overall BBU power exceeds it by 175%, on average (208% over full UL load). Interestingly, these numbers depend on the platform which hosts the BBU. Namely, Fig. 1b shows the BBU consumption over the baseline for different platforms.[2] We compare the power consumed by the BBU in idle state and operating at full UL/DL buffer, and subtract the baseline power. Indeed, the power cost changes significantly, and is affected also by the vBS bandwidth.

• Impact of SNR & MCS. The second finding is that the signal-to-noise ratio (SNR) of the wireless channel and the modulation and coding scheme (MCS) in UL affect the BBU computing load and hence its power consumption in a non-linear fashion. This is because the decoder needs more iterations when the received signal becomes noisier. Thus, the decoding time per subframe increases, e.g., by 52% between 20 and 15 dBs for MCS 23, see Fig. 2a; and this induces a commensurate increase in power consumption, see Fig. 2b. Besides, Fig. 2b shows that, even for a given decoding time, higher MCS values induce more power consumption, which is attributed to their more intricate demodulation. Importantly, excessive decoding delays can induce throughput loss since they lead to violations of vBS deadlines [4]. Hence, maximizing throughput does not only have an unpredictable effect on power, but it is indeed highly non-trivial.

• Configuration Options & Impact of Scheduler. The above vBS control challenges are exacerbated by the plenitude of configuration options. Fig. 3a, for instance, presents combinations of MCS and airtime values (percentage of used subframes) achieving the same UL throughput. Configurations with higher MCSs (and therefore lower airtime) reduce power by 38%. However, this relation is *non-monotonic*, as we have also measured higher power when the MCS increases and SNR is relatively low; this is due to the fast increase of computing load (see Fig. 2b). On the other hand, configurations 6 to 8 have the same power consumption, but still differ since config. 8 involves lower airtime and thus can serve more users, while 6 is more resilient to noise. These decisions are made by the vBS radio scheduler which based on the SNR selects the MCS and airtime. Fig. 3b shows the power consumption as a function of MCS and airtime for UL transmissions. We observe that both parameters have a smooth impact on power, but in practice this characterization is not available and needs to be learned.

[1]We use the Intels Running Average Power Limit (RAPL) functionality integrated into the Linux kernel to measure the CPU consumed power.

[2]The small factor PCs consume less power than the servers, which however can host more vBSs hence are expected to consume less power per user.

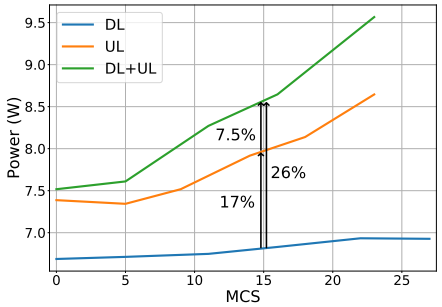


Fig. 4: MCS impact on BBU power consumption with high SNR.

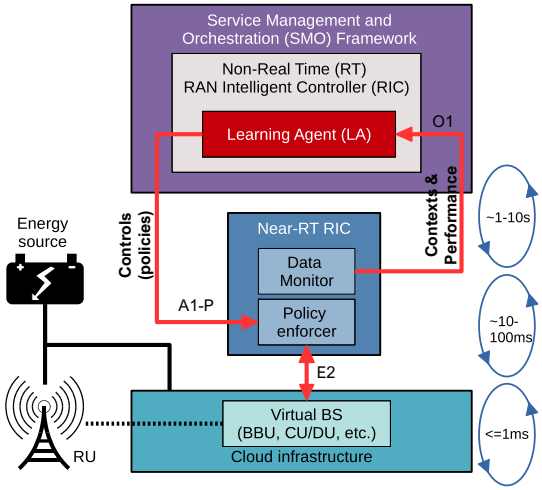


Fig. 5: O-RAN compliant system architecture and workflow.

• Coupling of DL & UL Processing. Finally, Fig. 4 shows the BBU power consumption when DL and UL traffic streams are processed separately and concurrently (UL+DL), for different MCSs and high SNR. We observe that the joint power is not the total sum of the separate components. For instance, for MCS 15, concurrent DL and UL processing consumes just 7.5% more than UL-only processing (and 26% over DL-only). This is because there are common power consumption factors in both links. This, in turn, makes it difficult to predict the overall vBS power consumption, given that the DL and UL can be configured separately. Also, note that UL power costs are higher and more volatile than DL, since decoding is more computationally demanding.

Conclusions: characterizing the vBS power consumption is intricate as it depends on traffic, SNR, MCS and airtime. There are many DL and UL configurations and some of them present *non-linear and non-monotonic* relations with power and throughput. Moreover, the power consumption depends on the BBU platform and radio scheduler. This hinders the derivation of general consumption models. Hence, we propose the use of *online learning* to profile each vBS power cost and performance, and devise goal-driven configuration policies.

IV. System Model and Problem Formulation

A. O-RAN Background and Model

We consider a virtualized Base Station (vBS) comprising a Baseband Unit (BBU), which may correspond to a 4G eNB or 5G gNB[3] hosted in a cloud platform and attached to a Radio Unit (RU), which are fed by a common and possibly constrained energy source. This type of BSs is relevant for low-cost small cells, Power-over-Ethernet (PoE) cells, and so on. Our goal is to use O-RAN's control architecture to select and adapt radio policies to system dynamics satisfying different energy-driven criteria. Fig. 5 shows the high-level system architecture, which is O-RAN compliant [5]. The Learning Agent (LA) runs online algorithms within the Non-Real-Time (Non-RT) RAN Intelligent Controller (RIC) in the system's orchestrator, and sequentially selects efficient *radio policies* every orchestration period t (in the order of seconds) *given the current context*. We hence formulate our problem as a Contextual Multi-armed Bandit or *Contextual Bandit*.

Contexts. We define the DL context at each period t as $\omega_t^{dl} := [\bar{c}_t^{dl}, \tilde{c}_t^{dl}, d_t^{dl}]$, where $\bar{c}_t^{dl}$ and $\tilde{c}_t^{dl}$ are the mean and variance of the DL channel quality indicator (CQI) across all users in the previous period; and d_t^{dl} is the *new* bit arrivals at the vBS DL aggregated across all users. Note that the DL CQI is sent periodically from the UEs to vBS through Uplink Control Information (UCI) carried by 4G/5G's Physical Uplink Shared Channel (PUSCH) or Physical Uplink Control Channel (PUCCH). Conversely, d_t^{dl} is measured by the vBS at the PDCP layer. Also, we define the UL context as $\omega_t^{ul} := [\bar{c}_t^{ul}, \tilde{c}_t^{ul}, d_t^{ul}]$. The UL CQI is measured by the vBS at MAC layer, and the new UL bit arrivals are estimated from the periodic Buffer Status Reports (BSRs) of the users (UEs). All these measurement are collected by the Near-RT RIC's Data Monitor (Fig. 5) from the vBS using the E2 interface at sub-second granularity, and are aggregated at the start of each orchestration period t. We denote the global context vector $\omega_t := [\omega_t^{dl}, \omega_t^{ul}] \in \Omega$, where Ω is the context space.

Controls. We define the DL control $x_t^{dl} := [p_t^{dl}, m_t^{dl}, a_t^{dl}]$ at period t, where $p_t^{dl} \in \mathcal{P}^{dl}$ is a *transmission power control (TPC) policy* for the maximum allowed vBS transmission power, $m_t^{dl} \in \mathcal{M}^{dl}$ is the highest MCS eligible by the vBS (*DL MCS policy*), and $a_t^{dl} \in \mathcal{A}^{dl}$ is the maximum vBS transmission airtime (*DL airtime policy*). We define the UL control $x_t^{ul} := [m_t^{ul}, a_t^{ul}]$, where $m_t^{ul} \in \mathcal{M}^{ul}$ and $a_t^{ul} \in \mathcal{A}^{ul}$ are the UL MCS and airtime policies.[4] We hence formalize each control at decision period t as a *radio policy* $x_t := [x_t^{dl}, x_t^{ul}] \in \mathcal{X}$, where $\mathcal{X} = \mathcal{P}^{dl} \times \mathcal{M}^{dl} \times \mathcal{A}^{dl} \times \mathcal{M}^{ul} \times \mathcal{A}^{ul}$ is the control space. Once computed, the LA sends each radio control policy to the Near-RT RIC via O-RAN's A1-P interface, which is then applied to vBS. The UL policies are applied by configuring each UL scheduling at the vBS MAC layer.

Rewards. We denote $R^{dl}(\omega_t^{dl}, x_t^{dl})$ and $R^{ul}(\omega_t^{ul}, x_t^{ul})$ the DL and UL data transmission rates, and define the *reward* function $r(\omega_t, x_t) :=$

$$\log\left(1+\frac{R^{dl}(\omega_t^{dl}, x_t^{dl})}{d_t^{dl}}\right)+\log\left(1+\frac{R^{ul}(\omega_t^{ul}, x_t^{ul})}{d_t^{ul}}\right) \quad (1)$$

[3]5G decouples BBU in 2 logical functions, i.e., a central unit (CU) and a distributed unit (DU). Our scheme controls the DU, or both when co-located.

[4]We do not define an UL TPC policy for simplicity because the users' transmission power has less impact on the vBS power than the MCS and UL airtime; but our framework can be readily extended to that.

where the logarithms are used to achieve fairness between the DL and UL flows – and to that end, one could use any other α-fair function [47]. The reward is computed at the end of each period by the Near-RT RIC's Data Monitor and sent to LA. It is important to stress that in practice we can only hope to observe **noisy values** of these functions, even when their arguments are fixed, because naturally the system operation is stochastic and also the power measurements are noisy – as we have indeed seen in our experiments. Fortunately, our optimization framework can handle such impairments. Henceforth, we denote $R_t^{dl}(\omega_t^{dl}, x_t^{dl})$, $R_t^{ul}(\omega_t^{ul}, x_t^{ul})$ and $r_t(\omega_t, x_t)$ the sample at period t of the functions.

B. Case 1: Balancing performance and cost

We consider first the case where the power supply is scarce, or the operator simply wants to reduce the power costs. This can be achieved with a scalarized *objective function*:

$$u(\omega_t, x_t) := r(\omega_t, x_t) - \delta B\big(P(\omega_t, x_t)\big), \tag{2}$$

where $P(\omega_t, x_t)$ is the vBS power consumption associated with the pair context-control (ω_t, x_t), $B(\cdot)$ corresponds to a smooth function that models the monetary cost associated with power consumption, and parameter δ prioritizes one over the other criterion based on the operator's preferences. Considering the noisy samples at period t $P_t(\omega_t, x_t)$ and $r_t(\omega_t, x_t)$, we define $u_t(\omega_t, x_t)$ as the objective function observation. In order to penalize power-consuming policies, we use a sigmoid function with sharpness and tipping parameters a and b:

$$B(k) := \frac{1 + e^{ab}}{e^{ab}} \left(\frac{1}{1 + e^{-a(k-b)}} - \frac{1}{1 + e^{ab}} \right). \tag{3}$$

When $a \to 0$, $B(\cdot)$ approximates a linear function, and the step function when a grows [48].

Following the standard approach in Bayesian bandit optimization [11], [26], we use the cumulative contextual regret to assess the performance of our algorithm. Namely, we define the average T-period contextual regret:

$$R_T := \sum_{t=1}^{T} \left(\max_{x' \in \mathcal{X}} u(\omega_t, x') - u(\omega_t, x_t) \right),$$

where $\max_{x' \in \mathcal{X}} u(\omega_t, x')$ yields the best decision for the current period, which we cannot calculate in practice since the objective function is unknown. Our goal, therefore, is to find a sequence of decisions $\langle x_t \rangle_{t=1}^{T}$ from set $\mathcal{X}$ which ensure asymptotically sublinear average regret, i.e., $\lim_{T \to \infty} R_T/T = 0$.

C. Case 2: Hard power budget

When the vBS operates under a hard power budget $P_{\max}$, e.g., when powered over Ethernet (PoE), the LA has to find the maximum-throughput configuration that respects this budget. Importantly, the LA needs to achieve that with a *safe* exploration of the configuration space $\mathcal{X}$ in order to satisfy the $P_{\max}$ threshold at any period, i.e., not only at the final optimal-operation stage. We define the respective regret:

$$R_T^s := \sum_{t=1}^{T} \left(\max_{x' \in S_t(\omega_t)} r(\omega_t, x') - r(\omega_t, x_t) \right), \tag{4}$$

where in this case the decisions are selected from set

$$S_t(\omega_t) = \{x \in \mathcal{X} \mid P(\omega_t, x) \leq P_{\max}\}. \tag{5}$$

Our goal is to find a sequence $\langle x_t \rangle_{t=1}^{T}$, $x_t \in S_t(\omega_t)$, such that $\lim_{T \to \infty} R_T^s/T = 0$. Note that sets $S_t(\omega_t), \forall \omega_t$, are unknown since $P(\omega, x)$ is unknown, and thus need to be learned from the measurements $P_t(\omega_t, x_t)$. And, similarly, we only have access to r_t and u_t, i.e., the t-period noisy measurements, instead of the actual functions r and u. To solve this problem, we propose a non-parametric learning approach in the sequel.

V. Bayesian Online Learning solutions

Next, we propose two online algorithms for solving the problems stated in Sections IV-B and IV-C.

A. BP-vRAN: Balancing performance and cost

Many algorithms for solving contextual bandit problems assume that there is a feature vector associated with each action, and the objective function is linear in that vector [49], [50]. However, this assumption does not hold here for the following reasons. Firstly, our objective function is not linear, as we can observe in eqs. (1)-(3). Secondly, the function values associated with different actions (i.e., vBS control policies) are correlated. Intuitively, we can think that a small change in some configuration parameter (e.g., airtime) will induce a small change in the vBS consumed power. This is actually evaluated experimentally in Fig. 3b. This means that we can obtain information about unobserved context-control pairs by observing nearby actions, thus reducing the exploration time.

Based on these observations, we propose a Bayesian optimization method where we model the objective function as a sample from a Gaussian Process (GP) over the joint context-control space. This non-parametric estimator captures the aforementioned non-linearities and correlations, and provides predictive uncertainty on the function estimation. Hence, addresses effectively the exploration - exploitation trade-off.

Function estimator. We use a GP as a function estimator, which is a collection of random variables following joint Gaussian distributions [9]. Let $z \in \mathcal{Z} = \Omega \times \mathcal{X}$ denote a context-control pair. We model the unknown objective function in eq. (2) as a sample from a $GP(\mu(z), k(z, z'))$, where $\mu(z)$ is its mean function and $k(z, z')$ is its covariance function or kernel. Without loss of generality, we assume $\mu := 0$ and a bounded variance $k(z, z) < 1$, which we refer to as the *prior distribution*, not conditioned on data. Given this prior and a set of observations, the mean and covariance of the *posterior distribution* can be computed using closed form formulas. Let $y_T = [u_1, \ldots, u_T]$ be a vector of noisy samples (assuming *i.i.d.* Gaussian noise $\sim N(0, \sigma^2)$) at points $Z_T = [z_1, \ldots, z_T]$. Then, the posterior distribution of the objective function follows a GP distribution with mean $\mu_T(z)$ and covariance $k_T(z, z')$:

$$\mu_T(z) = k_T(z^\top)(K_T + \zeta^2 I_T)^{-1} y_T \tag{6}$$

$$k_T(z, z') = k(z, z') - k_T(z^\top)(K_T + \zeta^2 I_T)^{-1} k_T(z') \tag{7}$$

where $k_T(z) = [k(z_1, z), \ldots, k(z_T, z)]^\top$, $K_T(z)$ is a kernel matrix defined as $[k(z, z')]_{z, z' \in Z_T}$, and I_T is the T-dimension

identity matrix. These equations allow us to estimate the distribution of unobserved values of $z \in \mathcal{Z}$ based on the prior distribution, the vector Z_T, and the function observations y_T.

Kernel function. The selection of the kernel is crucial because it shapes the prior and posterior GP distributions by encoding the correlation between the values of the objective function of every pair of points. Namely, $k(z, z')$ indicates the similarity between $u_t(z)$ and $u_t(z')$. In other words, the kernel characterizes the smoothness of the function [51]. Based on our experiments, we select the kernel for this problem satisfying two properties: *stationarity* and *anisotropicity*. On the one hand, a kernel $k(z, z')$ is stationary if it depends only on the distance between z and z', which means that it is invariant to translations in $\mathcal{Z}$. On the other hand, a kernel is anisotropic when the encoded smoothness is different among the different dimensions of $\mathcal{Z}$. That means that an anisotropic kernel is not invariant to rotations in $\mathcal{Z}$. We encode the smoothness of the objective function u with a length-scale vector $\mathcal{L} = [l_1, \ldots, l_N]$, where N indicates the number of dimensions of $\mathcal{Z}$. Thus, the distance between two points based on the length-scale vector can be written as

$$d(z, z') = \sqrt{(z - z')^\top L^{-2}(z - z')}, \tag{8}$$

where $L = \text{diag}(\mathcal{L})$ is a diagonal matrix of the length-scale values. We select the anisotropic version of the Matérn kernel, which satisfies the properties discussed above [9]. To this end, following standard practice, we particularize the Matérn kernel with parameter $\nu = \frac{3}{2}$ (details in [9]) to devise a simple expression that guarantees that the objective function is at least once differentiable, which yields:

$$k(z, z') = (1 + \sqrt{3}d(z, z')) \exp(-\sqrt{3}d(z, z')). \tag{9}$$

To improve performance, we can optimize the hyperparameters $\mathcal{L}$ and the noise variance ζ^2 (eq. (6)-(7)) before running the algorithm by maximizing the likelihood estimation over prior data and we keep these values constant over time.

Acquisition function. The acquisition function selects one control x_t at each period t based on the posterior distribution of the objective function over the context-control pairs. To this aim, we use the Upper Confidence Bound (UCB) method:

$$x_t = \underset{x \in \mathcal{X}}{\text{argmax}} \ \mu_{t-1}(\omega_t, x) + \sqrt{\beta_t}\sigma_{t-1}(\omega_t, x). \tag{10}$$

where ω_t is the observed context at time t, β_t is a weighting parameter and $\sigma_t(z) = k_t(z, z)$. To sum up, we formalize our approach, which we refer to as BP-vRAN (Bayesian optimization for Power consumption in vRANs), in Algorithm 1. At the beginning of each decision period t a context ω_t is observed (line 3). Then control x_t is decided based on the GP posterior and the acquisition function (line 5). At the end of t the throughput and consumed power are observed and the value of the objective function is computed (lines 6-7). The new measurements are included in Z_t and y_t to improve the posterior distribution in $t + 1$ (lines 8-10).

Theoretical results. The choice of a value for β_t in eq. (10) is very important since it controls the trade-off between exploration and exploitation. Larger values of β_t lead the acquisition function to select controls with higher uncertainty while, conversely, controls already known to be high-performing (though not necessarily *highest-performing*) are selected when β_t takes smaller values. Following [26], we select

$$\beta_t = 2B^2 + 300\gamma_t \ln^3(t/\epsilon) \tag{11}$$

where $\epsilon \in (0, 1)$, $B \geq \|u\|_k$ is an upper bound on the Reproductive Kernel Hilbert Space (RKHS) norm of u, and γ_t is the maximum mutual information gain obtained from u after t observations.

Algorithm 1 BP-vRAN: Performance and cost balancing

1: **Inputs:** Control Space $\mathcal{X}$, kernel k, β
2: **Initialize:** $y_0 = \emptyset$, $Z_0 = \emptyset$
3: **for** $t = 1, \ldots, T$ **do**
4: Observe the context ω_t
5: $x_t = \text{argmax}_{x \in \mathcal{X}} \ \mu_{t-1}(\omega_t, x) + \sqrt{\beta_t}\sigma_{t-1}(\omega_t, x)$
6: Measure $R_t^{dl}(\omega_t^{dl}, x_t^{dl})$, $R_t^{ul}(\omega_t^{ul}, x_t^{ul})$ and $P_t(\omega_t, x_t)$ at the end of the decision period t
7: Compute $u_t(\omega_t, x_t)$ using (1), (2) and (3)
8: Update $Z_t \leftarrow Z_{t-1} \cup [\omega_t, x_t]$
9: Update $y_t \leftarrow y_{t-1} \cup u_t(\omega_t, x_t)$
10: Perform Bayesian update to obtain μ_t and σ_t
11: **end for**

Lemma 1. *The contextual regret R_T of BP-vRAN satisfies*

$$P\left(R_T \leq \sqrt{C_1 T \beta_T \gamma_T} + 2 \ \forall T \geq 1\right) \geq 1 - \zeta \tag{12}$$

at stage T, where $C_1 = \frac{8}{\log(1+\sigma^{-2})}$ and $\gamma_t = \mathcal{O}(t^{44/45}\log(t))$.

Proof. For the derivation of the bound of the information gain γ_t, we consider a Matérn kernel with $\nu = \frac{3}{2}$ and $N = 11$ dimensions in $\mathcal{Z}$ (corresponding to a 6- and a 5-dimensional context and control space, respectively, as described in Sec. IV). For this setting, we particularize the expression provided in Theorem 5 of [33] to obtain the bound $\gamma_t = \mathcal{O}(t^{44/45}\log(t))$. For further details please see [26]. ■

B. SBP-vRAN: Safe Bayesian Optimization

Imposing hard constraints as proposed in Sec. IV-C, compounds the problem. Prior works, e.g., in robotics and other areas [10], [11], [52], [53], have proposed Bayesian optimization algorithms with *safety constraints*. Their main idea lays upon the definition: every t we define a subset of *safe* controls $S_t \subseteq \mathcal{X}$ that satisfy the constraints with certainty. Then, it is needed to interleave an exploration process so as to expand the safe set, while seeking a safe action with high performance. Unfortunately, these works do not consider contextual information, which clearly affects the safe set, i.e., $S_t(\omega_t) \subseteq \mathcal{X}$. To the best of our knowledge, only SafeOpt [53] proposes a contextual safe learning algorithm. However, although that algorithm provides theoretical guarantees, its acquisition function selects the control with the highest uncertainty among all candidates that can expand the safe set and also the potential maximizers. We found in our experiments that this approach has overly slow convergence. This practical issue has been reported in other works as well, e.g. [54]. Hence, we improve

Algorithm 2 SBP-vRAN: Safe online optimization

Inputs: Control Space $\mathcal{X}$, Initial safe set S_0, kernel k, β, $P_{\max}$
Initialize: $y_0^f = \emptyset$, $y_0^c = \emptyset$, $Z_0 = \emptyset$
for $t = 1, \ldots, T$ **do**
  Observe the context ω_t
  $S_t = S_0 \cup \{x \in \mathcal{X} \mid \mu_{t-1}^c(\omega_t, x) + \beta_t \sigma_{t-1}^c(\omega_t, x) \leq P_{\max}\}$
  $x_t = \text{argmax}_{x \in S_t} \ \mu_{t-1}(\omega_t, x) + \sqrt{\beta_t}\sigma_{t-1}(\omega_t, x)$
  Measure $R_t^{dl}(\omega_t^{dl}, x_t^{dl})$, $R_t^{ul}(\omega_t^{ul}, x_t^{ul})$ and $P_t(\omega_t, x_t)$ at the end of the decision period t
  Compute $r_t(\omega_t, x_t)$ using (1)
  Update $Z_t \leftarrow Z_{t-1} \cup [\omega_t, x_t]$
  Update $y_t^f \leftarrow y_{t-1}^f \cup r_t(\omega_t, x_t)$
  Update $y_t^c \leftarrow y_{t-1}^c \cup P_t(\omega_t, x_t)$
  Perform Bayesian update to obtain μ_t^f, σ_t^f, μ_t^c and σ_t^c
end for

this methodology by employing the acquisition function of CGP-UCB [26], but *constrained to the safe set.*

We denote $y_T^f = [r_1, \ldots, r_T]$ the vector of reward samples at T and $y_T^c = [P_1, \ldots, P_T]$ the power consumption samples. We use one GP for the reward and one for the power constraint. Both GPs have the same prior distribution and kernel but different hyperparameters. The posterior distribution can be computed using (6)-(7), and replacing y_T by y_T^f or y_T^c, for each GP. We denote the posterior mean and covariance of the reward at T as $\mu_T^f(z)$ and $k_T^f(z, z')$, and $\mu_T^c(z)$ and $k_T^c(z, z')$ for the power, respectively. The initial safe set $S_0 \subseteq \mathcal{X}$ is common for all contexts, and includes low power consumption configurations (vBS close to idle). This is worst-case S_0 can be expanded using prior data.

At each period, S_t is computed based on the posterior distribution of the power consumption provided by the GP. We assume the true value of the power consumption at time t is within the interval $[\mu_t^c(z) \pm \beta_t \sigma_t^c(z)]$, where $\sigma_t^c(z) = k_t^c(z, z)$. Using the posterior distribution, we define the safe set a time t and for a given context ω_t as:

$$S_t = \{x \in \mathcal{X} \mid \mu_{t-1}^c(\omega_t, x) + \beta_t \sigma_{t-1}^c \omega_t, x) \leq P_{\max}\}. \quad (13)$$

The controls are selected at each decision period t using the CGP-UCB policy constrained to the safe set:

$$x_t = \underset{x \in S_t}{\text{argmax}} \ \mu_{t-1}^f(\omega_t, x) + \sqrt{\beta_t}\sigma_{t-1}^f(\omega_t, x), \quad (14)$$

where $\sigma_t^f(z) = k_t^f(z, z)$.

We summarize our approach, named SBP-vRAN (Safe Bayesian optimization for Power consumption in vRANs), in Algorithm 2. Note that it does not expand explicitly the safe set, like it is done in SafeOpt [53]. Instead, a new observation is included at each t, which updates the posterior distribution of the power consumption and therefore alters the safe set indirectly. In our preliminary experimental campaign in Sec. III, we observed that, roughly speaking, controls with higher throughput are associated with higher power consumption. This leads Algorithm 2 to explore controls *at the boundary of the selected constraint*. As a result, the uncertainty around their neighborhood decreases, which allows Algorithm 2 to include more controls in the safe set. That is, SBP-vRAN's acquisition function exploits the structure of our problem to effectively expand the safe set, as our experimental evaluation presented in the next section demonstrates.

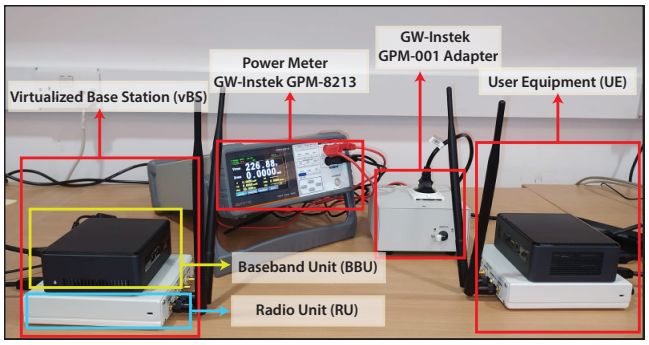


Fig. 6: Experimental vBS and UE testbed.

To conclude, it is important to remark that one inherent problem in GP-based approaches is that they need $\mathcal{O}(N^3)$ computation complexity (for matrix inversion) each orchestration period, where N is the number of data points. We observed in our experiments however that the unprecedented convergence speed of our methods pays off in a very short time. Moreover, we found that these computations do not induce a delay since, according to O-RAN specifications, we have a wide enough time window to update the policy.

VI. Experimental Evaluation

A. Experimental setup

Our testbed is shown in Fig. 6 and comprises a vBS, the user equipment (UE)[5], and a digital power meter. Both the vBS and the UE consist of an Ettus Research USRP B210 as RU, srseNB/srsUE (from srsLTE suite [4]) as BBU for both the eNB and UE, and two small factor general-purpose PCs (Intel NUCs with CPU i7-8559U@2.70GHz) deploying each respective BBU and the near-RT RIC of Fig. 5. The vBS and the UE are connected using SMA cables with 20dB attenuators, and we adjust the gain of the RU's RF chains to attain different SNR values. Without loss in generality, we select a 10-MHz band that renders a maximum capacity of roughly 32 and 23 Mbps in DL and UL, respectively. We use the power meter GW-Instek GPM-8213 to measure the power consumption of the BBU and the RU by plugging their power supply cable to a GW-Instek Measuring adapter GPM-001. Finally, we have integrated E2's interface and the ability to enforce control policies *on-the-fly* (see Section IV) in srseNB.

We use three auxiliary PCs (not shown in the picture) hosting the non-RT RIC and the network traffic end hosts, which use *mgen*[6]. We have finally implemented O1 interface (Fig. 5) using the USB-based power meter SCPI (Standard Commands for Programmable Instruments) interface concerning power consumption measurements and a REST interface for the remainder. A final remark is that our RU (USRP B210) does not integrate a variable power amplifier. Instead, it uses a fixed power amplifier consuming 3W and a variable attenuator for power calibration (see Fig. 1a). To compensate for this, we post-process the power measurements to include a variable RU

[5]We use one UE emulating the load of multiple users (see Section VI-C).
[6]https://www.nrl.navy.mil/itd/ncs/products/mgen.

consumption according to a linear model based on previous works [39], [41] and a 3W cap.

For the elaboration of the dataset used in Sec. III, we configure the vBS and UE in order to fix the conditions in the uplink and the downlink in terms of traffic load, channel quality, MCS, and airtime. Then, we fix each configuration for approximately one minute while the system takes measurements that later are processed to obtain its statistics. We assess the power behavior of the vBS by measuring the power consumption of its CPU and the whole BBU, the achieved performance in terms of throughput and goodput, details about the decoder at the vBS such as the subframe decoding time and the number of turbo decoder iterations per subframe, and some MAC and PHY indicators such as the Buffer Status Report (BSR), Block Error Rate (BSR), and the used MCS and airtime. Moreover, we detect and identify unfeasible configurations in the dataset. This mainly occurs when an MCS value is forced but the channel quality is not good enough to decode its data. Finally, we release our dataset[7] online allowing the community to realistically emulate the behavior of a vBS in terms of power consumption and performance as a function of its configuration and conditions (user traffic load and channel qualities) for future research.

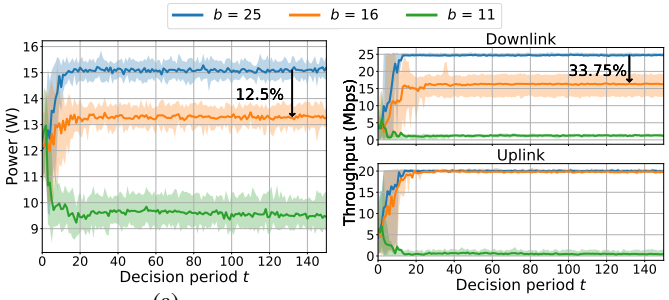
For our evaluation, we consider $|\mathcal{P}^{dl}| = 20$, $|\mathcal{M}^{dl}| = 28$, $|\mathcal{M}^{ul}| = 24$, and $|\mathcal{A}^{dl}| = |\mathcal{A}^{ul}| = 11$. In consequence, the size of the control set is $|\mathcal{X}| \approx 1.6 \cdot 10^6$. Note that, for a decision period of 10s, we would need 185 days to explore every control policy in $\mathcal{X}$ once, which highlights the need for a data-efficient learning strategy. Although Lemma 1 guarantees convergence and sublinear regret in general, faster convergence can be achieved with problem-specific information. Hence, and in line with previous works [53], [54], we select $\beta^{1/2} = 2.5$, which shows good performance in our setup. In the case of BP-vRAN, we configure $\delta = 20$ and set the parameters a and b in the penalty function, eq. (3), to severely penalize the power consumption values close to b or higher. Namely, we set $a = 2.5$ and evaluate different values of b. To visualize our results, we use lines representing the average along 10 independent runs and a colored shadow representing the area within the 10^{th} and 90^{th} percentiles.

The implementation of the algorithms BP-vRAN[8] and SBP-vRAN[9] used for this evaluation can be found online.

B. Convergence Evaluation

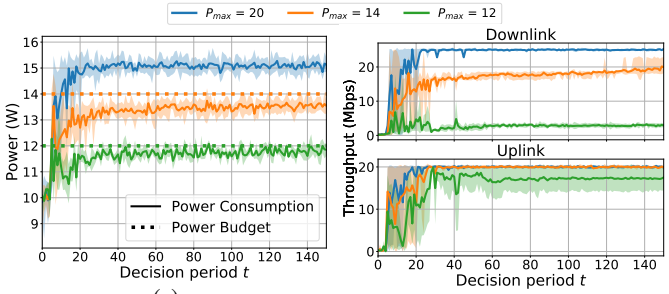
We start off by evaluating the convergence of BP-vRAN and SBP-vRAN. To this end, we consider the special case of a single context and observe their performance over time *with no prior training up till they converge to yield optimal policies*. We select a context with high SNR = 35 dB (CQI = 15) in DL and UL, and high traffic demands (relative to our testbed's capacity) equal to 25 and 20 Mbps for DL and UL, respectively. Fig. 7-8 show the temporal evolution of different metrics for both algorithms during 150 orchestration periods.

[7]https://github.com/jaayala/power_dlul_dataset
[8]https://github.com/jaayala/contextual_bayesian_optimization
[9]https://github.com/jaayala/constrained_bayes_opt



(a) (b)

Fig. 7: Convergence rate evaluation of BP-vRAN for different configurations of the objective function.



(a) (b)

Fig. 8: Convergence rate evaluation of SBP-vRAN for different values of the power budget P_{max}.

Let us discuss first the results of BP-vRAN in Fig. 7. We observe that the power consumption and, consequently, throughput, are lower for lower values of b, e.g., 12.5% power drop and 33.75% throughput drop between $b = 25$ and $b = 16$. This is intuitive because lowering b induces more stringent power requirements. Note that $b = 16$ only penalizes DL throughput. This is because it imposes a mild power requirement, and hence BP-vRAN *only* sacrifices transmission power, which reduces DL SNR and thus DL throughput. Lower values of b force BP-vRAN to sacrifice UL throughput too.

Concerning SBP-vRAN, we evaluate different values of P_{max} up to $P_{\text{max}} = 20$, which is an upper bound for the power consumption irrespective of the policy and the context. The results, in Fig. 8, depict how SBP-vRAN learns to use configurations within the power budget *with high probability*, sacrificing throughput when so required. Note that, in all the cases, SBP-vRAN always selects policies very close to P_{max}. This is because the optimal policy, i.e., the one that maximizes throughput, usually requires consuming all the P_{max} budget. To this end, SBP-vRAN gradually expands its safe set close to P_{max} and therefore an explicit strategy to expand the safe set is not needed. Specifically, Fig. 9 shows that all the controls are safe for $P_{\text{max}} = 20$, with 15.4% and 53.2% less safe policies for $P_{\text{max}} = 14$ and $P_{\text{max}} = 12$, respectively. As expected, lower values of P_{max} incur a smaller safe policy set.

We conclude this evaluation with the observation that, despite using a large set of policies $\mathcal{X}$, both algorithms converge in, at most, 30 orchestration periods. This result highlights the *data-efficiency* nature of our solutions, which are able to find optimal policies by observing only a very small subset of $\mathcal{X}$.

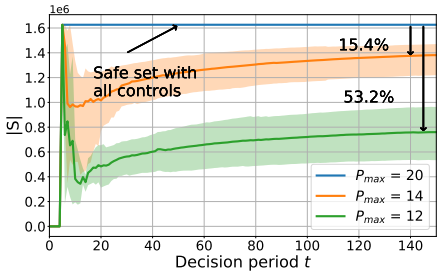


Fig. 9: Time evolution of safe set size of SBP-vRAN for different power budgets $P_{\max}$.

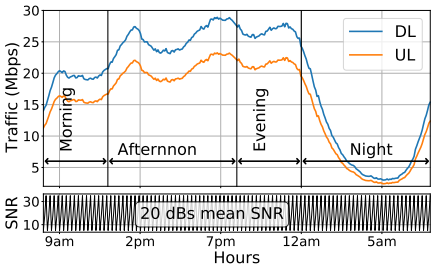
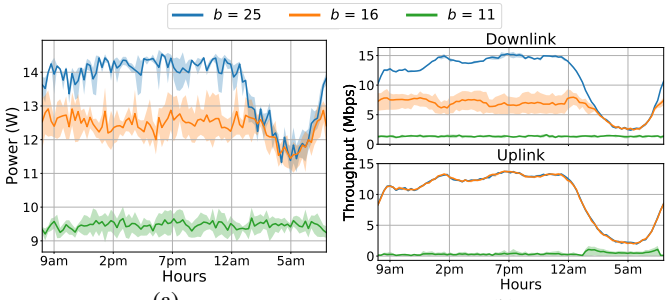


Fig. 10: One-day traffic pattern (top) and worst case channel quality pattern (bottom).

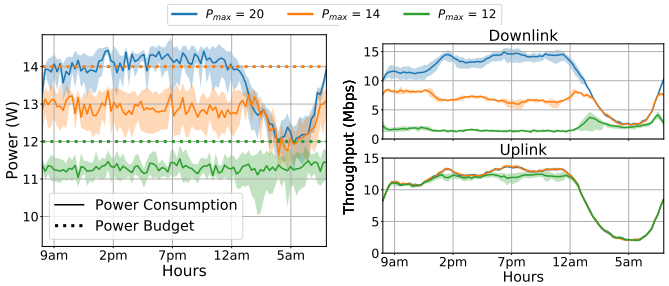
C. Performance in real network contexts

Next, we evaluate the performance of BP-vRAN and SBP-vRAN using a realistic one-day traffic pattern from [55] (Fig. 10, top). Concerning channel quality, we consider a worst-case pattern emulating UEs with high mobility (Fig. 10, bottom), which compromises network capacity (well below the demand). Due to the granularity of our traffic pattern dataset, we set the orchestration period length to 5 minutes in these experiments (note there is no loss in generality). We run our algorithms for two days and present results of the second day to focus on the attained system performance. Their convergence, evaluated in the previous subsection, takes just a few periods (well within a day). This is possible because the knowledge acquired by our algorithms for one context is *transferred to other similar contexts*. That is, after a few periods, the algorithms select efficient policies even for *unseen* contexts. To remove the clutter introduced by the high SNR variability under evaluation, each point in Figs. 11 and 12 corresponds to the average across all the points of a SNR cycle (see Fig. 10, bottom). Fig. 11 shows the total power consumption (a) and the evolution of throughput along the day (b) using BP-vRAN and different configurations of the objective function. We observe that the power consumption evolves with the traffic demand and with the selected value of b. For instance, when $b = 16$, the achieved throughput is penalized in favor of better power consumption during daylight but no performance degradation is required during the night (between 2am and 7am). Similarly, Fig. 12 shows the performance of SBP-vRAN under the same scenarios. Specifically, SBP-vRAN manages to satisfy the power budget constraint with probabilities 0.99 and 0.93 when $P_{\max}$ equals 14 and 12, respectively, while maximizing throughput (which we calculated through exhaustive search).



(a) (b)

Fig. 11: Performance evaluation of BP-vRAN throughout one day.



(a) (b)

Fig. 12: Performance evaluation of SBP-vRAN throughout one day.

VII. Conclusions

We have presented an in-depth experimental study of the energy behavior of virtualized base stations (vBSs). Our results made evident the complex relationship between performance, power consumption, and different vBS control policies. In light of these, we argued that such complexity can only be tamed with data-driven machine-learning solutions. To that end, we have proposed an online learning framework, compliant with O-RAN, to achieve two goals: (i) balance performance and power consumption in unconstrained platforms such as data centers; and (ii) maximize performance subject to power constraints vBS, e.g., solar-powered platforms or cells-on-wheels. We have followed a judicious design approach by resorting to Bayesian learning theory. This methodology allowed us to derive two algorithms, BP-vRAN and SBP-vRAN, that achieve the aforementioned goals (i) with theoretical performance guarantees, (ii) with high data-efficiency and convergence speed, and (iii) respecting power constraints even during learning. We have finally presented a thorough experimental evaluation of our algorithms using real-life traffic load and signal quality patterns. Our results demonstrated the ability of our approach to converge quickly to optimal policies. We have released the source code of BP-vRAN and SBP-vRAN along with the dataset used in this work to foster future research in this area.

Acknowledgments

This work was supported by the European Commission through Grant No. 856709 (5Growth) and Grant No. 101017109 (DAEMON); and by SFI through Grant No. SFI 17/CDA/4760.

REFERENCES

[1] "AT&T and Nokia Accelerate the Deployment of RAN Open Source," AT&T. Press Release, 2019. [Online]. Available: https://about.att.com/story/2019/open_source.html

[2] "Virtualized Radio Access Network: Architecture, Key Technologies and Benefits," Samsung. Technical Report, 2019.

[3] "Open & Virtualized – The Future of Radio Access Network," NEC. White Paper, 2020.

[4] I. Gomez-Miguelez *et al.*, "srsLTE: an Open-source Platform for LTE Evolution and Experimentation," in *in Proc. of ACM WinTech*, 2016.

[5] O-RAN Alliance, "O-RAN-WG1-O-RAN Architecture Description - v01.00.00." Technical Specification, Februry 2020.

[6] "Reimagining the End-To-End Mobile Network in the 5G Era," Cisco, Rakuten, Altiostar. White Paper, 2019.

[7] "5G network energy efficiency," Nokia Corporation. White Paper, 2016.

[8] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. d. Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[9] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[10] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with Gaussian Processes," in *Proc. of ICML*, 2015, pp. 997–1005.

[11] Y. Sui *et al.*, "Stagewise Safe Bayesian Optimization with Gaussian Processes," *arXiv preprint arXiv:1806.07555*, 2018.

[12] D. Bega *et al.*, "CARES: Computation-aware Scheduling in Virtualized Radio Access Networks," *IEEE Trans. on Wireless Communications*, vol. 17, no. 12, pp. 7993–8006, 2018.

[13] D. Raca *et al.*, "On Leveraging Machine and Deep Learning for Throughput Prediction in Cellular Networks: Design, Performance, and Challenges," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 11–17, 2020.

[14] N. Zhao *et al.*, "Deep Reinforcement Learning for User Association and Resource Allocation in Heterogeneous Cellular Networks," *IEEE Trans. on Wireless Communications*, vol. 18, no. 11, pp. 5141–5152, 2019.

[15] J. A. Ayala-Romero *et al.*, "vrAIn: A Deep Learning Approach Tailoring Computing and Radio Resources in Virtualized RANs," in *Proc. of ACM MOBICOM*, 2019.

[16] P. Rost *et al.*, "Computationally Aware Sum-Rate Optimal Scheduling for Centralized Radio Access Networks," in *Proc. of IEEE GLOBECOM*, 2015.

[17] K. Wang *et al.*, "Computing Aware Scheduling in Mobile Edge Computing System," *Springer Wireless Networks*, pp. 1–17, 2019.

[18] C. Zhang, P. Patras, and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 3, pp. 2224–2287, 2019.

[19] D. Bega *et al.*, "DeepCog: Optimizing Resource Provisioning in Network Slicing With AI-Based Capacity Forecasting," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 361–376, 2020.

[20] N. Liakopoulos *et al.*, "No Regret in Cloud Resources Reservation with Violation Guarantees," in *Proc. of IEEE INFOCOM*, 2019.

[21] V. Valls, G. Iosifidis, G. de Mel, and L. Tassiulas, "Online network flow optimization for multi-grade service chains," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1329–1338.

[22] J. A. Ayala-Romero, J. J. Alcaraz, A. Zanella, and M. Zorzi, "Online learning for energy saving and interference coordination in hetnets," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1374–1388, 2019.

[23] F. Mismar, J. Choi, and B. L. Evans, "A Framework for Automated Cellular Network Tuning With Reinforcement Learning," *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7152–7167, 2019.

[24] J. J. Alcaraz, J. A. Ayala-Romero, J. Vales-Alonso, and F. Losilla-López, "Online reinforcement learning for adaptive interference coordination," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 10, p. e4087, 2020.

[25] Z. Zhang, L. Ma, K. Poularakis, K. K. Leung, and W. Lingfei, "Dq scheduler: Deep reinforcement learning based controller synchronization in distributed sdn," in *Proceedings of IEEE ICC*, 2019.

[26] A. Krause and C. S. Ong, "Contextual Gaussian Process Bandit Optimization," in *Proc. of NIPS*, 2011, pp. 2447–2455.

[27] A. Bastian *et al.*, "CBA: Contextual Quality Adaptation for Adaptive Bitrate Video Streaming," in *Proc. of IEEE INFOCOM*, 2019.

[28] A. Galanopoulos, J. A. Ayala-Romero, G. Iosifidis, and D. Leith, "Bayesian online learning for mec object recognition systems," in *2020 IEEE Global Communications Conference (Globecom)*. IEEE, 2020.

[29] X. Wang, X. Guo, J. Chuai, Z. Chen, and X. Liu, "Kernel-based Multi-Task Contextual Bandits in Cellular Network Configuration," in *Proc. of IEEE Conf. on Big Data*, 2019.

[30] J. Chuai, Z. Chen, G. Liu, X. Guo, X. Wang, X. Liu, C. Zhu, and F. Shen, "A collaborative learning based approach for parameter configuration of cellular networks," in *Proceedings of IEEE INFOCOM*, 2019.

[31] M. Anjum Qureshi and C. Tekin, "Fast learning for dynamic resource allocation in ai-enabled radio networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 1, pp. 95–110, 2020.

[32] M. Hashemi, A. Sabharwal, C. E. Koksal, and N. B. Shroff, "Efficient Beam Alignment in Millimeter Wave Systems Using Contextual Bandits," in *Proc. of IEEE INFOCOM*, 2018.

[33] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design," in *Proceedings of ICML*, 2010.

[34] S. Bhaumik *et al.*, "CloudIQ: A Framework for Processing Base stations in a Data Center," in *Proc. of ACM Mobicom*, 2012.

[35] W. Wu *et al.*, "PRAN: Programmable Radio Access Networks," in *Proc. of ACM HotNets*, 2014.

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

[37] T. X. Tran *et al.*, "Understanding the Computational Requirements of Virtualized Baseband Units Using a Programmable Cloud Radio Access Network Testbed," in *Proc. of IEEE ICAC*, 2017.

[38] N. Nikaein, "Processing Radio Access Network Functions in the Cloud: Critical Issues and Modeling," in *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services*, 2015, pp. 36–43.

[39] G. Auer *et al.*, "How Much Energy is Needed to Run a Wireless Network?" *IEEE Wireless Communications*, vol. 18, no. 5, pp. 40–49, 2011.

[40] H. Holtkamp *et al.*, "A parameterized base station power model," *IEEE Communications Letters*, vol. 17, no. 11, pp. 2033–2035, 2013.

[41] O. Arnold, F. Richter, G. Fettweis, and O. Blume, "Power Consumption Modeling of Different Base Station Types in Heterogeneous Cellular Networks," in *IEEE Future Netw. & Mob. Summit*, 2010.

[42] M. Deruyck, W. Joseph, and L. Martens, "Power Consumption Model for Macrocell and Microcell Base Stations," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 3, pp. 320–333, 2014.

[43] B. H. Jung, H. Leem, and D. K. Sung, "Modeling of Power Consumption for Macro-, Micro-, and RRH-based Base Station Architectures," in *Proc. of IEEE VTC Spring*, 2014, pp. 1–5.

[44] C. Desset *et al.*, "Flexible power modeling of lte base stations," in *Proc. of IEEE WCNC*, 2012.

[45] B. Debaillie *et al.*, "A Flexible and Future-proof Power Model for Cellular Base Stations," in *Proc. of IEE VTC Spring*, 2015.

[46] T. Zhao, J. Wu, S. Zhou, and Z. Niu, "Energy-delay tradeoffs of virtual base stations with a computational-resource-aware energy consumption model," in *Proc. of IEEE ICCS*, 2014.

[47] J. Mo, and J. Walrand, "Fair End-to-End Window-Based Congestion Control," *IEEE/ACM Trans. on Netw.*, vol. 8, no. 5, pp. 556–567, 2000.

[48] L. Diez, A. Garcia-Saavedra, V. Valls, X. Li, X. Costa-Perez, and R. Aguero, "LaSR: A supple multi-connectivity scheduler for multi-RAT OFDMA systems," *IEEE Transactions on Mobile Computing*, 2018.

[49] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A Contextual-bandit Approach to Personalized News Article Recommendation," in *Proc. of ACM WWW*, 2010, pp. 661–670.

[50] P. Rusmevichientong and J. N. Tsitsiklis, "Linearly parameterized bandits," *Math. of Oper. Research*, vol. 35, no. 2, pp. 395–411, 2010.

[51] D. Duvenaud, "Automatic model construction with gaussian processes," Ph.D. dissertation, University of Cambridge, 2014.

[52] S. Amani *et al.*, "Regret Bounds for Safe Gaussian Process Bandit Optimization," *arXiv preprint arXiv:2005.01936*, 2020.

[53] F. Berkenkamp, A. Krause, and A. P. Schoellig, "Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics," *arXiv preprint arXiv:1602.04450*, 2016.

[54] M. Fiducioso, S. Curi, B. Schumacher, M. Gwerder, and A. Krause, "Safe oCntextual Bayesian Optimization for Sustainable Room Temperature PID Control Tuning," *arXiv preprint arXiv:1906.12086*, 2019.

[55] C. Márquez, M. Gramaglia, M. Fiore, A. Banchs, and Z. Smoreda, "Identifying Common Periodicities in Mobile Service Demands with Spectral Analysis," in *Proc. of MedComNet*, 2020.