On dimensioning Cloud-RAN systems

Veronica Quintuna
Orange Labs
Lannion
veronicakarina.quintunarodriguez@orange.com

Fabrice Guillemin
Orange Labs
Lannion
fabrice.guillemin@orange.com

ABSTRACT

We investigate in this paper the implementation of a Cloud-RAN architecture, where several software-based Base Band Units (BBUs) are collocated in a cloud data center. We specifically study several scheduling strategies in order to accelerate the runtime of virtualized BBU functions, and thus, to increase the span of a Cloud-RAN system, given that there are fixed deadlines for the execution of these functions. The main goal of this work is to obtain a simple model for dimensioning a Cloud-RAN infrastructure. For this purpose, we introduce the $M^{[X]}/M/C$ model to capture the behavior of a BBU-pool running on a multi-core platform. The theoretical approach is validated by simulation when performing a Cloud-RAN system hosting one hundred of base stations.

CCS CONCEPTS

• **Networks** → **Mobile networks**; Cloud computing; • **Mathematics of computing** → **Queueing theory**; *Markov processes*; • **Theory of computation** → Parallel computing models;

KEYWORDS

Cloud-RAN, NFV, scheduling, dimensioning, $M^{[X]}/M/C$ queue

ACM Reference Format:
Veronica Quintuna and Fabrice Guillemin. 2017. On dimensioning Cloud-RAN systems. In *VALUETOOLS 2017: 11th EAI International Conference on Performance Evaluation Methodologies and Tools, December 5–7, 2017, Venice, Italy*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3150928.3150937

1 INTRODUCTION

The emergence of Network Function Virtualization (NFV) [6] is fundamentally changing the way telecommunication infrastructures are designed and operated. The goal of NFV consists of replacing network functions running on dedicated and proprietary hardware with open software applications running on shared Commercial off-the-shelf (COTS) servers. With this new paradigm, network operators are able to instantiate Virtualized Network Functions (VNFs) on the fly at various network locations as needed by customers.

The NFV approach is actually applicable to any network function in mobile and fixed environments [7] where firewalls, load balancers, authentication or encryption procedures are the simplest examples. More complex network functions are currently under study, notably for the mobile core and radio access network.

In this paper, we focus on the virtualization of the Radio Access Network (RAN) (also known in the technical literature as vRAN, Cloud-RAN or C-RAN), which promises enormous advantages. For instance, it enables multi-site radio cooperation and interference management, which allows both better spectrum efficiency and user experience [4, 16].

The overarching principle of virtualization is to host network functions on one or more Virtual Machines (VMs) or containers. Ideally, VNFs should be located where they are the most efficient in terms of performance and cost. VNFs can be located in data centers, network nodes or even in end user devices depending on the required performance (notably latency) and resources (bandwidth, storage and computing).

In a cloud environment, the performance (in terms of latency) of a VNF is influenced by several factors from the hardware infrastructure to application layers, such as the heterogeneity of data centers in terms of computing architecture (GPU/CPU-based, AMD, x86, etc.), the design of the memory hierarchy [17], the virtualization technology (KVM, OpenStack, Kubernetes, etc.), and other components, in particular the scheduling algorithm that resides in the kernel of the Operating System (OS).

Beyond the behavior of the virtualized infrastructure, the performance of a VNF is also influenced by the software design and the programing model. A VNF can be conceived as a suite of components or sub-functions, which form a forwarding graph [7, 14]. Each component can in turn be split into runnable tasks or jobs, which are affected to processing units according to a given scheduling strategy. When the number of schedulable VNF's jobs outnumbers the computing resources, a significant performance degradation in terms of latency can occur. This degradation is particularly critical when a VNF requires real time processing as in the case of virtualized RAN (vRAN).

Many efforts have already been devoted to the achievement of traditional RAN performances with virtualized functions. Some studies reveal that functions belonging to the physical layer, especially the channel coding function, consume the largest amount of processing time and computing resources; see for instance [10]. Thus, besides the requirement of high-performance processors, parallel programing techniques need to be implemented to ensure the dynamicity and flexibility promised by NFV [13, 14].

However, the main performance problem of Cloud-RAN relies on the non-deterministic behavior of the channel coding function, i.e., the variability of runtime required by the encoding and decoding processes. Much of the variability is due to radio channel conditions of each User Equipment (UE) attached to the eNodeB (eNB), the required data rate per UE, as well as the amount of traffic in

VALUETOOLS 2017, December 5–7, 2017, Venice, Italy

ACM ISBN 978-1-4503-6346-4/17/12…$15.00
https://doi.org/10.1145/3150928.3150937

the cell. The radio scheduler, whose strategy is vendor-proprietary, allocates radio resources to each connected UE requiring transmission/reception. The amount of allocated resources (time and frequency) determines the size of channel coding jobs and consequently their required runtime.

The above observations raise fundamental questions with regard to dimensioning the required computing capacity under non-deterministic conditions. To address this problem, we model a high-performance Cloud-RAN system using queuing theory, namely, an $M^{[X]}/M/C$ system. The goal of this work is to determine the required computing capacity (i.e., the number of processing units) to deploy a Cloud-RAN system when parallelizing coding functions.

This paper is organized as follows: In Section 2, we present the fundamental principles for Cloud-RAN modeling, which subsequently allows us to introduce a queuing system formulation. The theoretical analysis of the Cloud-RAN model is exposed in Section 3. Some numerical experiments are given in Section 4 where we also validate the model by emulating a real Cloud-RAN system. Finally, in Section 5 we depict the main conclusions. Some theoretical developments are performed in Appendix A.

2 CLOUD-RAN SYSTEM

2.1 General assumptions

Cloud-RAN aims at centralizing the base-band processing of radio signals coming from various antennas in a Central Office (CO) or more generally in the cloud. In other words, Cloud-RAN dissociates antennas (Radio Remote Heads (RRHs)) and signal processing units (Base Band Units (BBUs)). Cloud-RAN can be seen as a BBU-pool which handles tens or even hundreds of cells-sites (eNBs). A site is typically composed of 3 sectors, each of them equipped with an RRH. The RRH has two RF paths for down-link and up-link radio signals which are carried by fiber links to the BBU-pool.

A Cloud-RAN's VNF is nothing else but a virtualized BBU (vBBU) which implements in software all network functions belonging to the three lower layers of the LTE protocol-stack, namely "base-band functions". These functions mainly concern IFFT/FFT (I/F), modulation and demodulation (M/D), encoding and decoding (CC), radio scheduling (RS), concatenation/segmentation of Radio Link Control (RLC) protocol, and encryption/decryption procedures of Packet Data Convergence Protocol (PDCP), for the down-link and up-link directions [3]. See Figure 1 for an illustration. The processing of the channel coding function is the most resource consuming and has furthermore a non-deterministic behavior. The challenge is to execute virtualized BBU functions sufficiently fast so as to increase the distance between RRHs and BBU functions (namely BBU-pool) and thus to improve the concentration level of BBUs in the CO for CAPEX and OPEX savings.

When a UE requires either data transmission or reception, a vBBU instance is invoked. As a consequence, various instances of the virtualized BBU run simultaneously in the computing platform. In LTE-based cellular systems, a transmission data unit (namely, a sub-frame) can allocate data of various UEs. The whole base-band processing of a sub-frame must be performed within 2 milliseconds and 1 millisecond in the up-link and down-link direction, respectively. Because a sub-frame is generated every millisecond in both directions, the base-band processing of all cells belonging to a

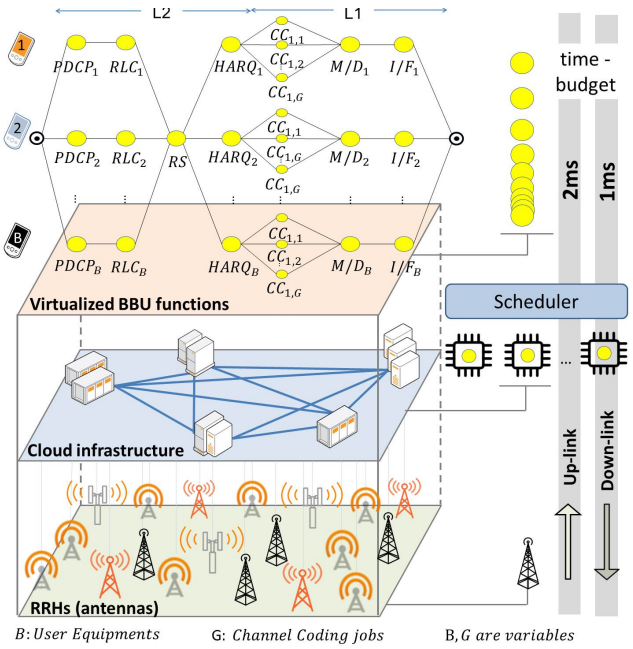

Figure 1: Cloud-RAN architecture

Cloud-RAN system very likely requires high-performance parallel computing.

The general philosophy of parallel computing consists of splitting large tasks into smaller parallel runnable sub-tasks which can be executed on multi-core systems. The parallel execution of sub-tasks allows the runtime of the whole task to be shortened. A vBBU can then be modeled on the top of the cloud infrastructure as a forwarding graph of sub-functions which in turn can be divided into parallel runnable tasks or jobs. See Figure 1 for an illustration. BBU's jobs are executed in a multi-core platform according to a scheduling strategy that resides in the kernel of the OS. In the proposed Cloud-RAN architecture, we take advantage of the performance provided by containers, which contrary to VMs hold a single OS [1].

In parallel computing, each job runs on a single core and only one at any instant [12]. Conversely, concurrent computing enables the simultaneous execution of jobs on a single core by overlapping time-periods; this leads to processor-sharing (PS) models [9]. However, the drawback of processor sharing is in that multitasking on the same core requires context switching and memory splitting, which can notably increase the latency.

2.2 Queuing System Formulation

From an analytical point of view each antenna (RRH) represents a source of jobs in the up-link direction, while for the down-link direction, jobs arrive from the core network which provides connection to external networks (e.g., Internet or other service platforms). There are then two queues of jobs for each cellular sector, one for up-link processing, and other for down-link.

Since the time-budget for processing down-link sub-frames is half of up-link ones, they might be executed separately on dedicated

processing units. However, this solution is not an efficient way of using limited resources [9]. We propose then a single-queuing system with a shared pool of processors, namely, a multi-core system with C cores. A global scheduler allocates computing resources to each runnable encoding (down-link) or decoding (up-link) job.

We assume that vBBUs (notably, virtual encoding/decoding functions) are invoked according to a Poisson process, i.e., inter-arrival times of runnable BBU functions are exponentially distributed. In LTE, frames occur with fixed relative phases but as a worst case, we can assume that they arrive as a Poisson process[1]. This is a reasonable assumption due to RRHs are at different distances of the BBU-pool. Furthermore, when considering no dedicated links, the front-haul delay (inter-arrival time) can strongly vary because of network traffic.

The parallel execution of encoding and decoding tasks on a multi-core system with C cores can be modeled by bulk arrival systems, namely, an $M^{[X]}/G/C$ queuing system[2]. We further consider each task-arrival to be in reality the arrival of x parallel runnable sub-tasks or jobs. Each sub-task requires a single stage of service with a general time distribution. The runtime of each sub-task depends on the workload as well as of the network sub-function that it implements. The number of parallel runnable sub-tasks belonging to a network sub-function is variable. Thus, we consider a non fixed-size bulk to arrive at each request arrival instant. The inter-arrival time is exponential with rate λ. The batch size is independent of the state of the system and is modeled as a random variable B.

In the case of Cloud-RAN, full functional parallelism is not possible since some base-band procedures (i.e., IFFT, modulation, etc) require to be executed in series. However, data parallelism of BBU functions (notably decoding and encoding) promises strong performance improvements. These claims are thoroughly studied in [13, 14]. Results show that the runtime of BBU functions can be significantly reduced when performing data parallelism in a sub-frame, i.e., through the parallel execution either of UEs or even of smaller data units, so-called Code Blocks (CBs). We present below a stochastic service model for each of them in order to evaluate the performance of a Cloud-RAN system.

2.3 Parallelism by UEs

In LTE, several UEs can be served into a sub-frame of 1 millisecond. The maximum and minimum number of UEs scheduled per sub-frame are determined by the eNB bandwidth. LTE supports scalable bandwidth of 1.25, 2.5, 5, 10 and 20 MHz. In a sub-frame, each scheduled UE receives a Transport Block (TB) (namely, a group of radio resources so-called Resource Block (RB)) either for transmission or reception. For example, when considering an eNB of 20 MHz, 100 RBs are available. According to LTE [5, 8], the minimum number of RBs allocated per UE is 6. Hence, the maximum number of connected UEs per sub-frame is given by $b' = \lceil 100/6 \rceil$. The Transport Block Size (TBS) is determined by the radio scheduler in function of the individual radio channel conditions of UEs as well as the traffic in the cell.

From the above observations, the parallel base-band processing (notably channel coding) of LTE sub-frames can be modeled as an $M^{[X]}/G/C$ queuing system. The number of jobs within a batch corresponds to the number of UEs allocated into an LTE sub-frame, e.g., the number of decoding jobs per millisecond in an eNB of 20 MHz range from 1 to 17. The batch size depends on the radio scheduling strategy (e.g., round robin, proportional fair), which takes into account the number of UEs requiring service in a cell and the radio channel conditions of each of them. A sub-frame then comprises a variable number of UEs, which is represented by the random variable B.

We further assume that the processing time of a job (namely that of a TB) is exponential. This assumption is intended to capture the randomness in the processing time of UEs due to non-deterministic behavior of the channel coding function. For instance, the decoding runtime of a single UE can range from few tens of microseconds to almost the entire time-budget, i.e., 2000 microseconds[3] [11]. In practice, this service time encompasses the response time of each component of the cloud computing system, i.e., processing units, RAM memory, internal buses, virtualization engine, data links, etc.

In the following, we precisely assume that the service time of a TB (i.e., a job) is exponentially distributed with mean $1/\mu$. If we further suppose that the number B of UE per sub-frame is geometrically distributed with mean $1/(1-q)$ (that is $\mathbb{P}(B = k) = (1-q)q^{k-1}$ for $k \geq 1$), then the complete service time of a frame is exponentially distributed with mean $1/((1-q)\mu)$.

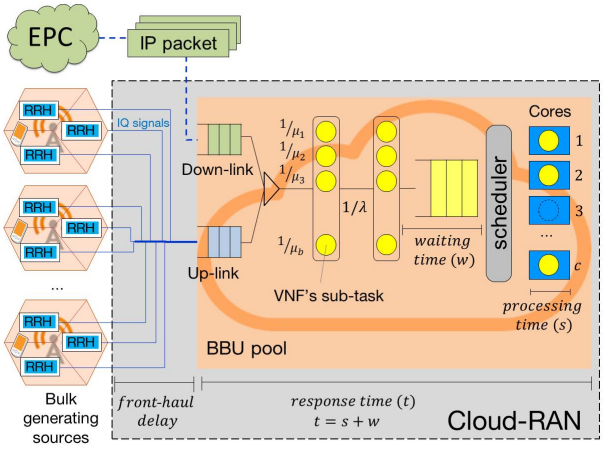

Figure 2: Stochastic service system for Cloud-RAN

With regard to the global Cloud-RAN architecture, the total amount of time t which is required to process BBU functions is given by $t = s + w$, where, s is the job's runtime and w is the waiting time of a job while there are no free processing units. The front-haul delay between RRHs and the BBU-pool is then captured by the arrival distribution. See Figure 2 for an illustration. When

[1] In the same way as an $M/D/1$ queue is "worse" with regard to waiting time than an $\sum_i N_i D_i/D/1$ queue.

[2] By analogy, concurrent computing of VNFs can be formalized by a single server queuing with a processor sharing discipline and batch arrivals, referred to as $M^{[X]}/G/1-PS$, where the single service capacity is done by the addition of individual capacities of all cores in the system. Because switching tasks produces undesirable overhead, this approach is not further considered in the present study.

[3] Runtime values are referential and correspond to the execution of OAI-based channel coding functions on x-86-based General Purpose Processors (GPPs) of 2.6 GHz.

assuming that the computing platform has a non-limited buffer, the stability of the system requires:

$$\rho = \frac{\lambda \mathbb{E}[B]}{\mu C} < 1. \quad (1)$$

In the following, we are interested in the sojourn time of sub-frames (batches) in the system, having in mind that if the sojourn time exceeds some threshold (i.e., 1 millisecond for encoding and 2 milliseconds for decoding) the sub-frame is lost. If we dimension the system so that the probability for the sojourn time to exceed the threshold is small, we can then approximate the sub-frame loss rate by this probability. It is worth noting that in LTE, retransmissions and reception acknowledgments are handled per sub-frame by the Hybrid Automatic Repeat reQuest (HARQ) process. When a TB is lost the whole sub-frame is re-send. It is hence utmost important that the processing time of a sub-frame does not exceed the prescribed threshold.

2.4 Parallelism by CBs

In LTE, when a TB is too big, it is split into smaller data units, referred to as CBs. If we assume that the processing time of a CB is exponential with mean $1/\mu'$, we again obtain an $M^{[X]}/M/C$ model, where the batch size is the number of CBs in a TB. If this number is geometrically distributed, the service time of a TB is exponential, as supposed above. The key difference now is that individual CBs are processed in parallel by the C cores. The scheduler is able to allocate a core to each CB because of more atomic decomposition of sub-frames and TBs.

2.5 No parallelism

If the processing of TBs or CBs is not parallel, the scheduling is carried out per sub-frame. Still assuming a multi-core system where sub-frames arrive according to a Poisson process, we are lead to consider an $M/G/C$ queuing system. By making the exponential assumptions for service times of CBs and TBs as well as supposing a geometric number of CBs per TB, we obtain an $M/M/C$ queue, which is well known in the literature.

3 BATCH MODEL

From the analysis carried out in the previous section, the $M^{[X]}/M/C$ model can reasonably be used to evaluate the processing time of a sub-frame in a Cloud-RAN architecture based on a multi-core platform. While the sojourn time of an arbitrary job of a batch has been analyzed in [2], the sojourn time of a whole batch seems to have received less attention in the technical literature. In this section, we derive the Laplace transform of this last quantity; this eventually allows us to derive an asymptotic estimate of the probability of exceeding a large threshold.

Let us consider an $M^{[X]}/M/C$ queue with batches of size B arriving according to a Poisson process with rate λ. The service time of a job within a batch is exponential with mean $1/\mu$. We assume that the load of the system $\rho < 1$ so that a stationary regime exists. The number N of jobs in the system in the stationary regime is such that [2]

$$\phi(z) \stackrel{def}{=} \mathbb{E}\left(z^N\right) = \frac{\sum_{k=0}^{C-1}(C-k)p_k z^k}{C - \frac{\lambda}{\mu} z\left(\frac{1-B(z)}{1-z}\right)},$$

where $p_k = \mathbb{P}(N = k)$ and $B(z)$ is the generating function of the batch size B, i.e., $B(z) = \sum_{k=0}^{\infty} \mathbb{P}(B = k) z^k$. As explained in [2], the probabilities p_k for $k = 1, \ldots C-1$ depend on p_0 which can eventually be computed by using the normalizing condition $\sum_{k=0}^{C-1}(C-k)p_k = C - \rho$.

We consider a batch of size b arriving at time t_0 and finding n jobs in the queue. We consider the following sub-cases.

First case: $n \geq C$. In that case, the first job of the tagged batch has to wait before entering service.

Second Case: $n < C$. In that case, $b \wedge (C-n) \stackrel{def}{=} \min(b, C-N)$ jobs of the tagged batch immediately enter service, the $0 \vee (b+n-C) \stackrel{def}{=} \max(0, b+n-C)$ jobs have to wait before entering service.

3.1 Analysis of the first case

In the case $n \geq C$, the tagged batch will have to wait for a certain time before the first job begins to be served. Let t_1 denote the time at which the first job of the tagged batch begins its service.

We obviously have that $T_1 = t_1 - t_0$ is equal to the sum of $n-C+1$ independent random variables exponentially distributed with mean $1/(\mu C)$. The Laplace transform of T_1 is defined for $\Re(s) \geq 0$ by

$$\mathbb{E}_b\left(e^{-sT_1}\right) = \left(\frac{\mu C}{s + \mu C}\right)^{n-C+1},$$

where $\mathbb{E}_b$ is the expectation conditionally on the batch size b.

Let t_2 denote the time at which the last job of the batch enters its service. The difference $T_2 = t_2 - t_1$ is clearly the sum of $b-1$ independent exponential random variables with mean $1/(\mu C)$; the Laplace transform of this difference is

$$\mathbb{E}_b\left(e^{-sT_2}\right) = \left(\frac{\mu C}{s + \mu C}\right)^{b-1}.$$

To completely determine the sojourn time of the tagged batch, it is necessary to know the number y_b of jobs which belong to this batch and which are in the queue when the last job of the batch begins its service. Let $t_1 = \tau_1 < \tau_2 < \ldots < \tau_b = t_2$ denote the service completion times of jobs (not necessarily belonging to the tagged batch) in the interval $[t_1, t_2]$. (Note that the point t_1 corresponding to the time at which the first job of the tagged batch enters service is itself a service completion time of one customer which was in the queue upon arrival of the tagged batch.) By definition τ_n is the time at which the n-th job of the tagged batch enters service.

Let us denote by y_n the number of jobs belonging to the tagged batch at time τ_n^+. Then, the sequence (y_n) is a Markov chain studied in Appendix A, where the conditional transition probabilities are expressed in terms of Stirling numbers of the second kind $S(n,k)$ [15] defined for $0 \leq k \leq n$ by

$$S(n,k) = \sum_{j=0}^{k} \frac{(-1)^{k-j}}{(k-j)!j!} j^n.$$

Stirling numbers are such that $S(n,n)=1$ for $n\geq 0$, $S(n,1)=1$ and $S(n,0)=0$ for $n\geq 1$, and satisfy the recursion for $n\geq 0$ and $k\geq 1$

$$S(n+1,k)=kS(n,k)+S(n,k-1).$$

With the above notation, when the b-th job of the tagged batch enters service, there are y_b jobs of this batch in the queue. The time T_3 to serve these jobs is

$$T_3=\mathcal{E}(y_b\mu)+\mathcal{E}((y_b-1)\mu)+\ldots+\mathcal{E}(\mu)$$

where $\mathcal{E}(k\mu)$ for $k=1,\ldots,y_b$ are independent random variables with mean $1/(k\mu)$. The Laplace transform of T_3 knowing y_b is

$$\mathbb{E}_b\left(e^{-sT_3}\mid y_b=k\right)=\frac{k!}{\prod_{\ell=1}^{k}\left(\frac{s}{\mu}+\ell\right)}=\frac{k!}{\left(\frac{s}{\mu}+1\right)_k},\qquad(2)$$

where $(x)_k$ is the Pochammer symbol (a.k.a. rising factorial) defined by $(x)_k=x(x+1)\ldots(x+k-1)$. By using Lemma A.1, it follows that the Laplace transform of the sojourn time T of a batch of size b in the system when there are $n\geq C$ customers in the queue upon arrival is

$$\mathbb{E}_b\left(e^{-sT}\mid N=n\geq C\right)=\frac{C!}{C^b}\left(\frac{\mu C}{s+\mu C}\right)^{n+b-C}\sum_{k=0}^{C}\frac{S(b,k)}{(C-k)!}\frac{k!}{\left(\frac{s}{\mu}+1\right)_k}.\qquad(3)$$

3.2 Analysis of the second case

When the number n of jobs in the queue is less than C upon the arrival of the tagged batch of size b, then $b\wedge(C-n)$ customers immediately enter service. Let us assume first that $b+n>C$. Taking the tagged batch arrival as time origin, the last job of the tagged batch enters service at random time T_2' with Laplace transform

$$\mathbb{E}(e^{-sT_2'})=\left(\frac{\mu C}{s+\mu C}\right)^{n+b-C}.$$

The number of jobs of the tagged batch present in the system when the last job enters service is Y_n such that

$$\mathbb{P}(Y_n=k)=\mathbb{P}(y_{b+n-C}=k\mid y_1=C-n),$$

where (y_n) is the Markov chain analyzed in Appendix A. For a given value $Y_n=k$, the time T_3 needed to serve all jobs of the tagged batch has Laplace transform given by Equation (2). By using Lemma A.1, we conclude that under the assumption $n<C$ and $b+n>C$, the sojourn time T of the tagged batch has Laplace transform

$$\mathbb{E}_b\left(e^{-sT}\mid N=n,b+N>C,N<C\right)=\left(\frac{\mu C}{z+\mu C}\right)^{n+b-C}\sum_{k=C-n}^{C}\mathbb{P}(Y_n=k)\frac{k!}{\left(\frac{s}{\mu}+1\right)_k}.\qquad(4)$$

The probability $\mathbb{P}(Y_n=k)$, $k\geq C-n$, can be expressed in terms of polynomials $\mathcal{A}_{n,p}(x)$ introduced in Appendix A as

$$\mathbb{P}(Y_n=k)=\frac{1}{C^{n+b-C-1}}\binom{n}{k+n-C}\mathcal{A}_{n+b-C-1,k+n-C}(C-n).$$

It follows that

$$\mathbb{E}_b\left(e^{-sT}\mid N=n<C,b+N>C\right)=\left(\frac{\mu C}{z+\mu C}\right)^{n+b-C}f(n,b;s),\qquad(5)$$

where

$$f(n,b;s)=\frac{1}{C^{n+b-C-1}}\sum_{k=C-n}^{C}\binom{n}{k+n-C}\frac{k!\mathcal{A}_{n+b-C-1,k+n-C}(C-n)}{\left(\frac{s}{\mu}+1\right)_k}.\qquad(6)$$

When $b+n\leq C$, all jobs of the tagged batch enter service just after arrival and the Laplace transform of the sojourn time is

$$\mathbb{E}_b\left(e^{-sT}\mid N=n,b+N\leq C\right)=\frac{b!}{\left(\frac{s}{\mu}+1\right)_b}.\qquad(7)$$

3.3 Main result

By using the results of the previous sections, we determine the Laplace transform $\Phi(s)=\mathbb{E}(e^{-sT})$ of the sojourn time of a batch in the $M^{[X]}/M/C$ queue; the proof of the following result is straightforward and is then omitted for the sake of conciseness.

Theorem 3.1. *The Laplace transform $\Phi(s)$ is given by*

$$\Phi(s)=\beta(s)\left(\phi\left(\frac{\mu C}{s+\mu C}\right)-\phi_C\left(\frac{\mu C}{s+\mu C}\right)\right)+\mathbb{E}\left(\frac{B!}{\left(\frac{s}{\mu}+1\right)_B}\mathbb{P}(N\leq C-B)\right)+\sum_{n=0}^{C-1}p_n\mathbb{E}\left(f(n,B;s)\left(\frac{\mu C}{s+\mu C}\right)^{n+B-C}\right),\qquad(8)$$

where

$$\beta(s)=\mathbb{E}\left(\frac{1}{C^{B-1}}\left(\frac{\mu C}{s+\mu C}\right)^{B-C}\sum_{k=0}^{C}\binom{C-1}{k-1}\frac{\mathcal{A}_{B,k-1}(1)}{\left(\frac{s}{\mu}+1\right)_k}\right),\qquad(9)$$

$$\phi_C(z)=\sum_{n=0}^{C-1}p_nz^n,$$

and $f(n,b;s)$ defined by Equation (6).

Following [2], let us define z_1 the root with smallest module to the equation

$$V(z)\stackrel{def}{=}C-\frac{\lambda}{\mu}z\left(\frac{1-B(z)}{1-z}\right)=0;$$

the root z_1 is real and greater than 1. The negative real number

$$s_1=-\mu C\left(1-\frac{1}{z_1}\right)$$

is the singularity with the smallest module of the Laplace transform $\Phi(s)$ if $s_1>-\mu$ (namely, $z_1<\frac{C}{C-1}$ or $V\left(\frac{C}{C-1}\right)>0$).

Corollary 3.2. *If $s_1>-\mu$, then when t tends to infinity*

$$\mathbb{P}(T>t)\sim\frac{\mu CU(z_1)\beta(s_1)}{s_1z_1^2V'(z_1)}e^{s_1t},$$

where $U(z) = \sum_{k=0}^{C-1}(C-k)p_k z^k$. *If* $s_1 < -\mu$, *then the decay rate of the distribution of* T *is* $-\mu$.[4]

It is worth noting that when the batch size is geometrically distributed with mean $1/(1-q)$, (i.e., $\mathbb{P}(B=k) = (1-q)q^{k-1}$), we have $s_1 = -(1-q)\mu C(1-\rho)$ and

$$z_1 = \frac{C}{qC + \frac{\lambda}{\mu}} > 1 \text{ for } C > \frac{\lambda}{(1-q)\mu}. \tag{10}$$

It is worth noting that $z_1 < \frac{C}{C-1}$ if and only if $\rho > 1 - \frac{1}{C(1-q)}$.

4 NUMERICAL EXPERIMENTS

In this section, we evaluate by simulation the behavior a Cloud-RAN system hosting the base band processing of one hundred of base stations. The goal is to test the relevance of the $M^{[X]}/M/C$ model for dimensioning purposes.

Cloud-RAN sizing refers to determining the minimum number of servers (cores) which are required to ensure the processing of LTE sub-frames within deadlines for a given number of base stations (eNBs). As a consequence, the maximum front-haul distance between antennas and the BBU-pool can also be estimated.

In LTE, deadlines are applied to the whole sub-frame. For instance, when the runtime of the base-band processing of a sub-frame in the up-link direction exceeds 2 milliseconds, the whole sub-frame is lost and therefore retransmitted. In order to bring new perspectives for the radio channel efficiency, we also evaluate the loss of single users, thereby, RAN systems might hold less redundant data. The loss of sub-frames as well as UEs are captured in the $M^{[X]}/M/C$ model by the impatience of batches and customers respectively.

4.1 Simulation settings

We evaluate a Cloud-RAN system hosting 100 eNBs where each of them has a bandwidth of 20 MHz. All eNBs have a single antenna (i.e., work under Single Input Single Output (SISO) configuration) and use Frequency Division Duplex (FDD) transmission mode. Antennas (eNBs) are distributed around the computing center within a 100 km radius.

In the following, we focus our analysis on the reception process carried out in the up-link direction due to the non-deterministic behavior of the decoding function, as well as because it is the greatest computing resource consumer of all BBU functions [11, 13].

To assess the runtime of the decoding function, we use Open Air Interface (OAI)'s code, which implements all BBU functions in open-source software[11].

4.2 Model analysis

In order to reflect the behavior of a Cloud-RAN system using the $M^{[X]}/M/C$ model, we feed the queuing system with statistical parameters captured from the Cloud-RAN emulation during the busy-hour. We capture the behavior of the decoding function in a multi-core system performing parallelism by UEs. The obtained parameters are as follows:

- The mean service time of decoding jobs, $\mathbb{E}[S]$, is set to 281 microseconds. Each decoding job corresponds to the data of a single UE.
- The mean number of decoding jobs requiring service at the same time, i.e, the mean batch size, is given by $\mathbb{E}[B] = 5$. As explained in Section 2, the number of UEs scheduled per sub-frame can vary between 1 and 17 for an eNB of 20 MHz. This asset is in accordance with values following a geometric distribution of parameter $q = 0.8$ ($q = 1 - \frac{1}{\mathbb{E}[B]}$), where batch-sizes are in such interval with a probability of 0.98.
- The mean inter-arrival time of batches is 10 microseconds. Each eNB generates a bulk of decoding jobs (sub-frame) every millisecond. Hence, the mean inter-arrival time come up of dividing the periodicity of sub-frames by the number of eNBs.
- The time-budget (deadline) for the up-link processing is given by $\delta = 2000$ microseconds.

We can then evaluate the $M^{[X]}/M/C$ model with the following parameters: $\mu = 1/281$ and $\lambda = 1/10$. We easily obtain from Equation (1); for $C = 150$, $\rho = 0.9367$. The CDFs of the sojourn time of jobs and batches are shown in Figure 3. By using Corollary 3.2, we verify that if D is the sojourn time of a job in the $M^{[X]}/M/C$ queue, then $\mathbb{P}(T > t)/\mathbb{P}(D > t)$ tends to a constant when $t \to \infty$. It can also be checked that the slopes of the curves $-\log(P(D > t))/t$ and $-\log(P(T > t))/t$ for large t are both equal to μ.

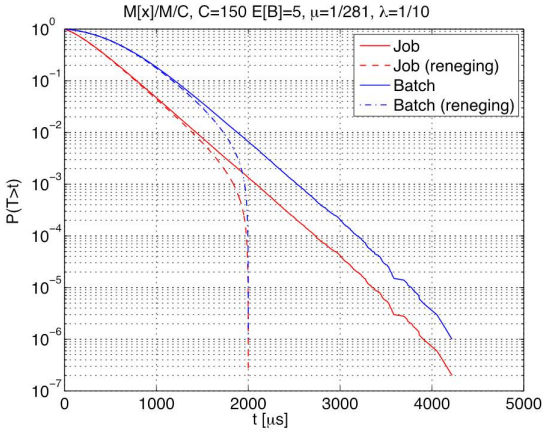

Figure 3: Sojourn time of jobs and batches

In practice, aborting the execution of sub-frames which overtake deadlines is highly desirable to save computing resources. We are then interested in the behavior of the $M^{[X]}/M/C$ with reneging of both customers and batches. A job (customer) leaves the system (even during service) when its sojourn time reaches a given deadline δ. In case of batch's reneging, the sojourn time of a batch is calculated from the arrival until the instant in which the last job composing the batch is served. Results performing impatient customers and batches are included in Figure 3.

When performing impatience, the loss rate of jobs and batches, is respectively 0.0013 and 0.0065. We observe that the gap between the two rates (i.e., 0.0065/0.0013) is close to the mean batch size, $\mathbb{E}[B]$. This is true when loss rates are at least of order 10^{-3}.

[4] Contrary to what is stated in [2], the same result holds of the decay rate of the sojourn time of a job in the system.

Due to the complexity of the theoretical analysis of impatience-based models, we choose to use the performance of a $M^{[X]}/M/C$ system without reneging for sizing a Cloud-RAN infrastructure. Since this model stochastically dominates the system with reneging, we obtain conservative bounds. As illustrated in Figure 4, we verify for both jobs and batches, that the probability of deadline exceedance is always greater in a system without reneging, and moreover, these two probabilities are close to each other when C increases.

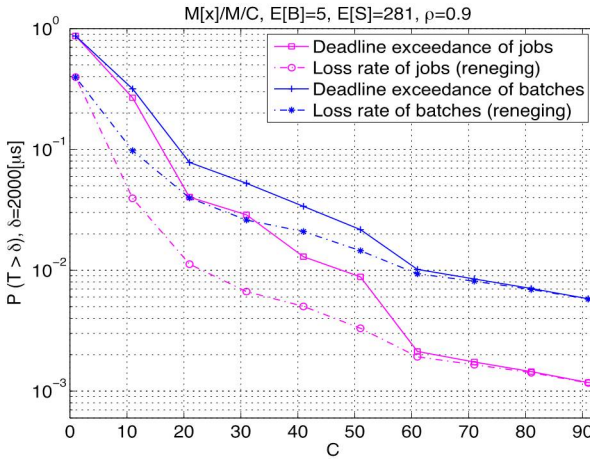

Figure 4: Deadline exceedance of jobs and batches

4.3 Cloud-RAN dimensioning

The final goal of Cloud-RAN sizing is to determine the amount of computing resources needed in the cloud (or a data center) to guarantee the base-band processing of a given number of eNBs within deadlines. For this purpose, we evaluate the $M^{[X]}/M/C$ model (without reneging) while increasing C, until an acceptable probability of deadline exceedance (say, ε). Consequently, the required number of cores is the first value that achieves $P(T > \delta) < \varepsilon$.

We validate by simulation the effectiveness of the $M^{[X]}/M/C$ model with the behavior of the real Cloud-RAN system during the reception process (up-link) of LTE sub-frames. See Figure 5 for an illustration. Results show that for a given $\varepsilon = 0.00615$, the required number of cores is $C_r = 151$, which is in accordance with the real Cloud-RAN performance where the probability of deadline exceedance is barely 0.00018.

When C takes values lower than a certain threshold C_s, the Cloud-RAN system is overloaded, i.e., the number of cores is not enough to process the vBBUs' workload; the system is then unstable. The threshold C_s can be easily obtained from Equation (1); for $\rho = 1$, $C_s = \lceil \lambda * E[B]/\mu \rceil = 141$ cores.

5 CONCLUSION

We have proposed a stochastic multi-server system with batch arrivals of BBU-jobs, namely the $M^{[X]}/M/C$ model, for Cloud-RAN performance modeling. Both arrival and service distributions respectively capture the variability of the front-haul delay and jobs' runtime. The batch-size varies with a geometric distribution to reflect the changing number of UEs in an LTE sub-frame. We have evaluated by simulation the sojourn time of both BBU jobs (customers) and LTE sub-frames (batches), as well as, a Cloud-RAN system during the busy-hour. Due to strict LTE deadlines for processing BBU jobs, we have also performed the $M^{[X]}/M/C$ model with impatient jobs.

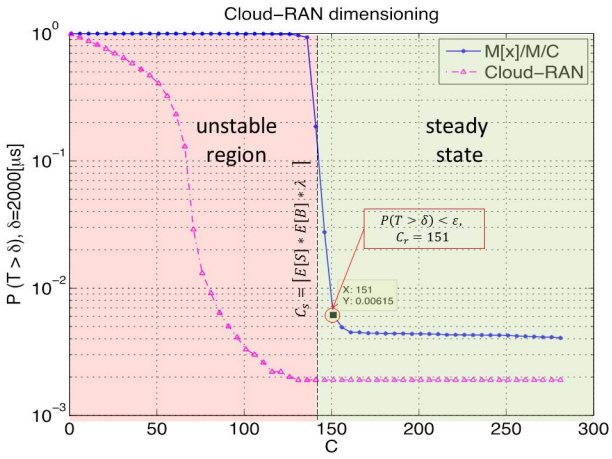

Figure 5: Cloud-RAN sizing when using the $M^{[X]}/M/C$ model

Results show that for Cloud-RAN dimensioning purposes, we are able to use the simplest model (without reneging), given that, when the probability of deadline exceedance is sufficiently low (of order 10^{-3}), the incidence of the impatience criterion is negligible. The main conclusion is that the theoretical model is reasonably accurate to reflect the Cloud-RAN performance.

A ANALYSIS OF THE MARKOV CHAIN

The Markov chain (y_n) describing the number of jobs of the tagged batch at times (τ_n^+) for $n = 1, \ldots, b$ is such that

$$y_n = \begin{cases} y_{n-1} & \text{with probability } y_{n-1}/C, \\ y_{n-1} + 1 & \text{with probability } (C - y_{n-1})/C. \end{cases}$$

By definition $y_n \in \{1, \ldots, C\}$ and the Markov chain is absorbed at state C. We show the following result.

Lemma A.1. *The conditional transition probabilities of the Markov chain (y_n) are given for $k \geq \ell$ by*

$$\mathbb{P}(y_n = k \mid y_1 = \ell) = \frac{(C-\ell)!}{(C-k)!C^{n-1}} \sum_{m=0}^{n-1} \binom{n-1}{m} S(m, k-\ell) \ell^{n-1-m}. \quad (11)$$

Proof. The transition matrix of the Markov chain (y_n) is

$$P = \begin{pmatrix} \frac{1}{C} & \frac{C-1}{C} & 0 & \cdot & \\ 0 & \frac{2}{C} & \frac{C-2}{C} & 0 & \cdot \\ & & \ddots & & \\ & & & 0 & 1 \end{pmatrix}.$$

Let us first consider the case $y_1 = 1$, we have

$$\mathbb{P}(y_n = k \mid y_1 = 1) = e_1^t P^{n-1} e_k,$$

where e_n is the column vector with all entries equal to 0 except the n-th one equal to 1, and e_n^t is the transpose of vector e_n. To compute the matrix P^n, we diagonalize the matrix P.

The eigenvalues of matrix P are obviously the positive reals $x_j = \frac{k}{C}$ for $j = 1, \ldots, C$. Simple computations show that the eigenvector associated with the eigenvalue x_j is the vector v_j with entries

$$v_j(\ell) = \frac{(j-1)\ldots(j-\ell+1)}{(C-1)\ldots(C-\ell+1)}$$

for $\ell = 1, \ldots, j$ (with $v_1 = 1$) and $v_j(\ell) = 0$ for $\ell = j+1, \ldots, C$. Note that we have for $\ell = 1, \ldots, j$

$$v_j(\ell) = \frac{(j-1)!(C-\ell)!}{(j-\ell)!(C-1)!}.$$

The vectors v_j for $j = 1, \ldots, C$ are obviously linearly independent and form a basis of $\mathbb{R}^C$. To compute $P^n e_k$, we determine the representation of e_k on the basis (v_j). By setting

$$e_k = \sum_{j=1}^{C} u_j^{(k)} v_j.$$

We easily check that $u_j^{(k)} = 0$ for $j = k+1, \ldots, C$ and

$$u_k^{(k)} = \frac{(C-1)!}{(C-k)!(k-1)!}. \tag{12}$$

Simple manipulations then show that the other coefficients can be obtained by solving the matrix equation $U_k \widetilde{u}^{(k)} = \beta^{(k)}$, where the matrix U_k with coefficients $\binom{j}{\ell}$ for $\ell = 1, \ldots, k-1$, $\ell \le j \le k-1$ and other coefficients equal to 0, is an upper triangular Pascal matrix and β_k is the vector with entries

$$\beta_k(\ell) = -\frac{(C-1)!}{(C-k)!(k-\ell)!\ell!}$$

for $\ell = 1, \ldots, k-1$ and $\widetilde{u}^{(k)}$ is the vector with entries equal to $u_j^{(k)}/j$ for $j = 1, \ldots, k-1$.

It is classical that the inverse of the matrix U_k is the upper triangular matrix with coefficients $(-1)^{j+\ell}\binom{j}{\ell}$ for $\ell = 1, \ldots, k-1$, $\ell \le j \le k-1$ and other coefficients equal to 0.

We eventually come up with the representation

$$e_k = \sum_{j=1}^{k} (-1)^{k+\ell} \frac{(C-1)!}{(C-k)!(j-1)!(k-j)!} v_j$$

for $k = 1, \ldots, C$. It follows that

$$P^{n-1} e_k = \sum_{j=1}^{k} (-1)^{k+j} \frac{(C-1)!}{(C-k)!(j-1)!(k-j)!} \left(\frac{j}{C}\right)^{n-1} v_j$$

and then for $n \ge 1$

$$\mathbb{P}(y_n = k \mid y_1 = 1) = \sum_{j=0}^{k} (-1)^{k+j} \frac{C!}{(C-k)!j!(k-j)!} \left(\frac{j}{C}\right)^{n}.$$

The above expression can be rewritten as

$$\mathbb{P}(y_n = k \mid y_1 = 1) = \frac{C!}{(C-k)!} \frac{S(n,k)}{C^n},$$

where $S(n,k)$ denotes the Stirling number of the second kind [15].

In the same way we have for $\ell \le k$

$$\begin{aligned}\mathbb{P}(y_n = k \mid y_1 = \ell) &= e_\ell^t P^{n-1} e_k \\ &= \frac{(C-\ell)!}{(C-k)!C^{n-1}} \sum_{m=0}^{n-1} \binom{n-1}{m} S(m, k-\ell) \ell^{n-1-m}\end{aligned}$$

and Equation (11) follows. Note that by the identity

$$\sum_{m=0}^{n} \binom{n}{m} S(m,k) = S(n+1, k+1),$$

the above expression is valid for $\ell = 1$. □

To conclude this section, let us note that by introducing the family of polynomials $\mathcal{A}_{n,p}(x) = p! \sum_{j=0}^{n} \binom{n}{j} S(j,p) x^{n-j}$, we have

$$\mathbb{P}(y_n = k \mid y_1 = \ell) = \frac{1}{C^{n-1}} \binom{C-\ell}{k-\ell} \mathcal{A}_{n-1,k-\ell}(\ell). \tag{13}$$

REFERENCES

[1] Rabindra K Barik, Rakesh K Lenka, K Rahul Rao, and Devam Ghose. 2016. Performance analysis of virtual machines and containers in cloud computing. In *Computing, Communication and Automation (ICCCA), 2016 International Conference on*. IEEE, 1204–1210.

[2] M.V. Cromie, M.L. Chaudhry, and W.K. Grassman. 1979. Further results for the queueing systems $M^X/M/c$. *J. Opl Res. Soc.* 30, 8 (1979), 755–763.

[3] Erik Dahlman, Stefan Parkvall, and Johan Skold. 2013. *4G: LTE/LTE-advanced for mobile broadband*. Academic press.

[4] Ericsson. 2015. Cloud-RAN. *White Paper* (2015).

[5] Safa Essassi, Ridha Hamila, Sofiane Cherif, Mohamed Siala, and Mazen Omar Hasna. 2016. RB allocation based on genetic algorithm in cloud radio access networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2016 International*. IEEE, 1002–1005.

[6] GSNFV ETSI. 2013. 001:" Network Functions Virtualisation (NFV). *Architectural Framework* (2013).

[7] GSNFV ETSI. 2013. Network functions virtualisation (NFV); use cases. *V1* 1 (2013), 2013–10.

[8] ETSI TS 136 213 v12.4.0 2015. *LTE, Evolved Universal Terrestrial Radio Access, Physical layer procedures (3GPP TS 36.213 version 12.4.0 Release 12)*. Standard. European Telecommunications Standards Institute.

[9] Leonard Kleinrock. 1976. *Queueing Systems*. Vol. II: Computer Applications. Wiley Interscience.

[10] Navid Nikaein. 2015. 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*. ACM, 36–43.

[11] Navid Nikaein, Mahesh K Marina, Saravana Manickam, Alex Dawson, Raymond Knopp, and Christian Bonnet. 2014. OpenAirInterface: A flexible platform for 5G research. *ACM SIGCOMM Computer Communication Review* 44, 5 (2014), 33–38.

[12] R Pike and A Gerrand. 2012. Concurrency is not parallelism. *Heroku Waza* (2012).

[13] Veronica Quintuna-Rodriguez and Fabrice Guillemin. 2017. Towards the deployment of a fully centralized Cloud-RAN architecture. In *International Wireless Communications and Mobile Computing Conference (IWCMC 2017)*. IEEE.

[14] Veronica Quintuna-Rodriguez and Fabrice Guillemin. 2017. VNF modeling towards the cloud-RAN implementation. In *Networked Systems (NetSys), 2017 International Conference on*. IEEE, 1–8.

[15] John Riordan. 1968. *Combinatorial identities*. Wiley, New York.

[16] Ericsson Telefonica. 2015. Cloud-RAN Architecture for 5G. *White Paper* (2015).

[17] Chengwei Wang, Oliver Spatscheck, Vijay Gopalakrishnan, Yang Xu, and David Applegate. 2016. Toward High-Performance and Scalable Network Functions Virtualization. *IEEE Internet Computing* 20, 6 (2016), 10–20.