Dynamic Resource Allocation for URLLC and eMBB in MEC-NFV 5G Networks

Caio Souza[*,a], Marcos Falcão[a], Andson Balieiro[a], Elton Alves[b] and Tarik Taleb[c]

[a]Centro de Informatica (CIn) - Universidade Federal de Pernambuco, Recife - PE, Brazil.
[b]Falcudade de Computacao - Universidade do Sul e Sudeste do Para, Maraba, Brazil.
[c]Faculty of Electrical Engineering and Information Technology - Ruhr University Bochum, Bochum, Germany.

ARTICLE INFO

Keywords:
Multi-access Edge Computing
Ultra-Reliable Low-Latency Communications
Continuous-Time Markov Chain
Network Function Virtualization
Enhanced Mobile Broadband
Dynamic Resource Allocation.

ABSTRACT

Supporting the coexistence between enhanced Mobile Broadband (eMBB) and Ultra-Reliable Low Latency Communications (URLLC) is a major challenge in modern communication systems due to their diverse requirements. Multi-access Edge Computing (MEC), Network Function Virtualization (NFV), and Network Slicing (NS) emerge as complementary paradigms to address this challenge, providing fine-grained, on-demand resources closer to the User Equipment (UE) and enabling shared utilization of physical infrastructure. This paper addresses the combination of MEC, NFV, NS, and dynamic virtual resource allocation for overcoming the problem of resource dimensioning at the network edge supporting eMBB and URLLC services. We have proposed a Continuous-Time Markov Chain (CTMC) model to evaluate how requests are managed by the virtualization resources of a single MEC node, primarily focusing on fulfilling the requirements of both eMBB and URLLC services. It characterizes the dynamic virtual resource allocation process and incorporates three key performance metrics, relevant for both URLLC and eMBB services (e.g., availability and response time) as well as for service providers (e.g., power consumption). The model also integrates practical factors such as failures during service processing, service prioritization, and setup (repair) times, enabling insights into how the MEC-NFV-based 5G network handles different service categories by applying service prioritization and dynamic resource allocation. Our key findings reveal that container setup and failure rates play a crucial role in both availability and response times, higher setup rates improve availability and shorten response times. Additionally, the number of containers significantly enhances both metrics, whereas buffer sizes primarily influence response times. Furthermore, higher eMBB arrival rates reduce availability and increase response times, while URLLC availability remains unaffected.

1. Introduction

The Fifth Generation of mobile networks (5G) introduced a robust cloud-native core network with network slicing capabilities, empowering the creation of innovative services such as enhanced Mobile Broadband (eMBB), Ultra Reliable Low Latency Communications (URLLC), and massive Machine Type Communications (mMTC) [1]. While eMBB supports applications requiring higher bandwidth, such as immersive communications and smart offices, URLLC addresses services with strict reliability and latency requirements (e.g., autonomous vehicles and telesurgery). Meanwhile, mMTC is tailored to support a vast number of Internet of Things (IoT) devices, transmitting small amounts of data sporadically.

The synergy between Multi-access Edge Computing (MEC) and Network Function Virtualization (NFV), often referred to as MEC-NFV, plays a crucial role in advancing URLLC. This combination empowers the hosting of virtualized network functions and applications in closer proximity to end-users, resulting in a significant reduction in latency and an enhancement of overall reliability. In addition, MEC-NFV's advantages extend beyond URLLC, benefiting eMBB services as well. NFV facilitates dynamic resource allocation and scaling, aligning network capacity with demand fluctuations and effectively minimizing latency during peak usage periods. Furthermore, content and applications can be cached and processed at the network edge, further ensuring rapid response times.

The eMBB service represents a natural evolution of the mobile broadband provided by LTE networks, being the most widely used service in mobile networks, and also the first category targeted for support by 5G networks in Release 15. On the other hand, URLLC is perhaps the most challenging service to be supported due to its stringent latency and reliability requirements [2], which may be compromised by factors such as failures, virtualization overhead, and coexistence with other services.

Although the mMTC is an important service category in 5G networks, it predominantly addresses connection density for low-power devices with infrequent transmission events, where high data rates or low latency are not critical. Thus, this paper focuses on eMBB and URLLC services, where the concept of Network Slicing (NS) is crucial to enable their coexistence [3]. NS plays a fundamental role in enabling the shared utilization of physical infrastructure within dynamic on-demand networking platforms, allowing for the creation of multiple virtual networks. This concept leverages the virtualization of both edge and core network functions, making effective use of well-established virtualization technologies. The coexistence of eMBB and URLLC is expected to encompass a wide range of use cases, however, it also

*Corresponding author
ORCID(s): 000-002-4187-1078 (C. Souza)
[1]E-mail address: cbbs@cin.ufpe.br (C Souza)

poses great challenges such as how to find a balance between their divergent requirements during the dynamic resource allocation in MEC-NFV domain [4].

Multiple works have addressed the coexistence between different service categories within 5G networks. However, they predominantly focus on radio resource allocation within the Radio Access Network (RAN) ([4], [5], [6], [7]). Thus, there exists a notable gap when it comes to considering factors that influence resource provisioning in the MEC-NFV domain. Notably, prior research often presupposes fault-free cloud environments ([4] and [8]) or with instantaneous provisioning times ([5], [6], [9], [10]) which may not align with the remaining components of 5G networks. Furthermore, studies often do not consider that there are service subcategories that differ widely ([8], [11], [12], [13]) and neglect the overhead caused by the virtualization and dynamic resource allocation. For instance, the Virtualized Network Function (VNF) instance boot process plays a key role in cost-performance analysis for both edge and core 5G networks. During installation, energy is consumed, and resources are allocated, yet services remain unattended. This entails repercussions not only in terms of energy efficiency but also in the context of potential Service Level Agreement (SLA) violations.

This paper addresses the combination of MEC, NFV, and the dynamic virtual resource allocation within the context of coexisting 5G service categories: URLLC and eMBB, aiming at the challenge of resource dimensioning in compact MEC-NFV nodes. We propose a Continuous-Time Markov Chain (CTMC) based model to characterize the dynamic virtual resource allocation, incorporating three performance metrics, which will be relevant not only for URLLC and eMBB services (e.g., availability and response time) but also for service providers (e.g., power consumption). In addition, to make the model more practical, we have integrated factors like resource failures, service prioritization, and setup (repair) times into the formulation, as they can incur significant impacts on the 5G applications' requirements. Moreover, the MEC-NFV node model incorporates dynamic scaling capabilities and service prioritization to accommodate the two 5G service categories. Some of our key findings include the observation that higher eMBB arrival rates decrease availability and increase response times, while URLLC availability remains stable. Moreover, the container setup rates and failure rates substantially affect both availability and response times, with higher setup rates enhancing availability and reducing response times. Also, the number of containers emerges as a significant factor, enhancing both availability and response times, while buffer sizes primarily affect response times.

The remainder of this paper is organized as follows. Section 2 discusses relevant works in the field of MEC-NFV resource allocation. Section 3 describes the proposed CTMC-based model for a single node MEC-NFV, assuming a virtual environment featured with containers that are able to process both URLLC and eMBB requests. Section 4 presents the model validation and results obtained by extensive discrete-event simulations. Finally, Section 5 provides our concluding remarks and highlights future work directions.

2. Related Work

This section provides a comprehensive overview of the models proposed in the literature to address MEC-NFV in 5G networks. The focus is particularly placed on the distinct characteristics addressed by each model, including the specific problem(s) they tackle, the types of services involved, and the mathematical tools employed. Additionally, it aims to elucidate the contribution of the present work in comparison to that of the existing literature.

2.1. Addressed Problems and Network Segment

A body of existing literature on radio and computational resource issues related to the MEC-NFV architecture encompasses various classes of problems, including resource scheduling, Dynamic Resource Allocation (DRA), and resource dimensioning [14]. In this section, we provide a summary of the main studies in these fields, focusing on the addressed problems, the network segments involved (RAN or Core functions), and the 5G service categories. Additionally, since all of the following works are analytical in nature, we also extract the mathematical tools utilized to build their models.

The first five works in Table 1 address the radio resource sharing between 5G service categories. In [4], the authors tackle the challenge of sharing radio resources between eMBB and URLLC, involving a trade-off between latency, reliability, and spectral efficiency, using Combinatorial Programming. Similarly, [5] proposes a dynamic joint scheduling approach for URLLC and eMBB traffic at the sub-frame level, utilizing a queuing mechanism to monitor and control URLLC packet latency in real-time. In [6], the proposal involves an overlapping scheme of puncturing a portion of resources scheduled to an eMBB packet for URLLC packets, impacting eMBB packets. This work extends the method provided by ITU-R to reflect the puncturing of URLLC on eMBB and considers additional delays due to retransmissions, utilizing Queueing Theory. [15] proposes a dynamic resource provisioning scheme that leverages multiple base stations and a shape-based heuristic to optimize resource utilization and enhance QoS across diverse traffic types. Similarly, [16] explores dynamic network slicing for virtualized RANs, managing mixed traffic with varying QoS needs. It employs a coarse provisioning scheme with deep reinforcement learning and a shape-based heuristic to optimize resource utilization and QoS.

The subsequent works in [9], [10], and [7] explore end-to-end characteristics, encompassing both RAN and Core segments. However, they only consider a single service category (URLLC). In particular, [9] develops a DRA algorithm that minimizes end-to-end delay while ensuring a minimum service rate and maximum reliability, considering VNF mapping in both the core and access networks to

minimize end-to-end delay and ensure network slice reliability. Similarly, in [10], the author discusses how to meet the reliability and latency requirements in URLLC using stochastic network calculus (SNC). The paper constructs a tandem model that describes communication in the 5G network and analyzes parameters influencing the delay. Lastly, [7] proposes an NFV-enabled 5G paradigm for industry applications, supporting URLLC through service chain acceleration and dynamic blockchain-based spectrum resource sharing among various applications running on NFV-based equipment.

The remaining works (Table 1) are focused on core network functions and do not consider RAN characteristics. Moreover, it is important to note that none of these works address two or more 5G service categories simultaneously; they are generally dedicated to a single category or agnostic towards a specific category. For instance, [13] proposes an analytical model based on CTMC, along with an optimization problem, to determine the optimal number of virtual resources for maximizing task execution capacity. The paper jointly considers contention-based communications for task offloading, parallel computing, and the occupation of failure-prone MEC processing resources, without focusing on a specific service category. Similarly, [12], [11], and [8] do not specify service categories. In [12], the authors propose a stochastic geometry and CTMC-based spatiotemporal framework to analyze the intertwined communication and computation performance of edge computing systems, focusing on the influence of various parameters on task response delay. In [11], an online task offloading and resource allocation approach for edge-cloud orchestrated computing is proposed, aiming to minimize the average task latency using a mixed-integer optimal decision approach. Lastly, [8] designs a task offloading strategy for MEC systems to enhance user experience quality and increase energy efficiency. The paper establishes a model, formulates a joint optimization problem, and analyzes the influence of parameters on the task offloading strategy to achieve optimal results.

The final set of works includes previous studies on dynamic resource allocation in core networks supporting a single service category (URLLC). For instance, [17] proposes an analytical CTMC framework to evaluate a hybrid virtual MEC environment that combines the strengths of Virtual Machines (VMs) and Containers to meet URLLC constraints while providing cloud-like Virtual Network Function (VNF) elasticity. Similarly, [18] introduces a single MEC-NFV node model that enables resource pre-initialization to mitigate the negative effects of VNF failures and setup times. Finally, [19] presents another analytical model to effectively dimension a MEC-enabled UAV node, considering availability, power consumption, reliability, and latency perspectives.

In this work, we focused on the DRA problem in the core network segment, utilizing CTMC as the main mathematical tool. However, this work introduces a key aspect that distinguishes it from the previously discussed literature: it addresses two 5G service categories, eMBB and URLLC, within a single model. Additionally, it incorporates other critical factors, such as failure possibility, setup and repair times, and the evaluation of system metrics, including availability, power consumption, and response time, as redescribed in the following sections. Table 1 summarizes the related work contributions in terms of addressed problems, network segments, 5G service categories, and approaches adopted for modeling.

2.2. Model Assumptions

There is currently no consensus on the size, computational power, or appropriate virtualization technology for the MEC-NFV architecture [20]. Decisions regarding these aspects may depend on technical and business parameters, such as existing suppliers and contracts that may constrain the choice of virtualization technology available to Mobile Network Operators, available site facilities, supported applications, and their requirements, estimated user load, operation, and deployment costs. [21]. While container-based virtualization has gained significant attention, much of the literature remains agnostic towards specific virtualization technologies, reflecting a lack of commitment to the feasibility of their proposals. Containers are software unities that package source code, libraries, and dependencies, offering portable, isolated environments for running applications [22].

More importantly, current virtualization technology may struggle to accommodate the data volume and specific requirements of 5G service categories. While virtualization offers flexibility, it also introduces overhead that can degrade network performance compared to non-virtualized systems, especially for applications with strict low-latency requirements [23]. Therefore, it is crucial to consider events that may hinder the communication process, such as container failures and setup delays. In this section, we further evaluate the previously described works, focusing on the key considerations regarding container usage in the MEC-NFV architecture, aiming to provide more realistic analytical models.

The primary challenge of using containers in MEC-NFV infrastructure is their ability to isolate different VNFs within the same environment, compared to virtual machines, since container-based VNFs share a common operating system [24]. Containerization introduces multiple security risks, as all containers within an OS share a single kernel. Consequently, a breach in the OS kernel can compromise all dependent containers. Furthermore, isolating faults within containers is not trivial, and a fault can be replicated across subsequent instances. In addition to failures, we evaluate two other phenomena: (1) the VNF instantiation process, which represents the delay until a VNF is ready to process a request after being turned off. It may comprise initializing the kernel image and launching the specified function. (2) The repair time denotes the duration taken for a VNF to recover from a failure event. Neglecting these factors can be problematic and result in nodes with underestimated size, for example.

Table 1
Problem, Network Segment, Service Types, and Mathematical Tools.

Work	Problem	Network Segment	5G service Types	Mathematical Tools
[4]	Scheduling and DRA	RAN	URLLC, eMBB	Combinatorial Programming
[5]	Scheduling and DRA	RAN	URLLC, eMBB	Queueing Theory
[6]	Scheduling and DRA	RAN	URLLC, eMBB	Queueing Theory
[15]	DRA	RAN	URLLC, eMBB, mMTC	Queueing Theory
[16]	DRA	RAN	URLLC, eMBB, mMTC	Queueing Theory
[13]	DRA	CORE (MEC)	n/a	CTMC, Stochastic Geometry
[9]	DRA	RAN and CORE	URLLC	Graph Theory
[10]	Delay Bound	RAN and CORE	URLLC	Queueing Theory, SNC
[12]	Offloading	CORE (MEC)	n/a	CTMC, Stochastic Geometry
[11]	Offloading and DRA	CORE (MEC)	n/a	n/a
[8]	DRA	CORE (MEC)	n/a	Queueing Theory
[7]	DRA	RAN, CORE	URLLC	SNC
[17]	DRA	CORE (MEC)	URLLC	CTMC
[18]	DRA	CORE (MEC)	URLLC	CTMC
[19]	DRA	CORE (MEC)/UAV	URLLC	CTMC
This Work	DRA	CORE (MEC)	URLLC, eMBB	CTMC

Moreover, some studies do consider failure events but do not account for repair times, as seen in [11], [7], and in [5]. This omission may impact metrics such as resource availability and power consumption. In our work, we have adopted the considerations from the previously mentioned studies: [12], [17], [19] and [18], as they provide a satisfactory approach to address the evaluated events, encompassing all three phenomena. Table 2 summarizes the assumptions made by each evaluated work.

Table 2
Model Assumptions.

Work	Setup Time	Failure	Repair Time
[4]	✗	✗	✗
[5]	✗	✓	✗
[6]	✗	✓	✓
[15]	✗	✗	✗
[16]	✗	✗	✗
[13]	✗	✓	✓
[9]	✗	✓	✓
[10]	✗	✓	✓
[12]	✗	✓	✗
[11]	✗	✓	✗
[8]	✗	✗	✗
[7]	✗	✓	✗
[17]	✓	✓	✓
[18]	✓	✓	✓
[19]	✓	✓	✓
This Work	✓	✓	✓

2.3. Performance Metrics

Since the introduction of 3GPP Release 16 [1], significant attention has been given to potential architecture enhancements aimed at supporting URLLC services through MEC and NFV. In addition to the fundamental performance metrics of latency and reliability, the literature explores other metrics, such as resource availability, which is essential for resource provisioning and dimensioning schemes, as well as energy-related metrics, which are particularly important for infrastructure providers. Furthermore, it is important to note that the interpretation of performance metrics may vary depending on the network segment being analyzed. This section provides an overview of the main performance metrics examined in some of the previously described works. Specifically, we focus on three performance metrics: Availability, Power Consumption, and Response Time, although some works may address additional metrics.

In studies characterizing the RAN ([4], [5], [6], [15], and [16]), the definitions of latency and reliability differ from those used in core or edge networks. Among these, [4] concentrates solely on latency, while [5] exclusively examines reliability. Notably, [6] is the sole study to simultaneously consider both reliability and latency, which aligns with the requirements of URLLC. In contrast, [15] and [16] do not address reliability, latency, or energy consumption in their analysis. The former evaluates queue length, resource utilization, and satisfaction ratios at both the slice and system levels, while the latter examines queue length, resource usage, slice satisfaction, and isolation.

The remaining works primarily focus on the backhaul, leading to differences in the interpretation of certain performance metrics compared to the RAN.. Latency-related metrics are commonly evaluated in some of these works, as in [12]. However, it is common to find evaluations involving two or more metrics simultaneously. For instance, in [13] explores availability and reliability while imposing an energy constraint per device. Since no dedicated formulation for the energy metric is provided, it is treated as a constraint rather than a distinct performance metric. In contrast, [10], [9], and [7] focus exclusively on reliability and latency. Furthermore, [11] and [8] both evaluate energy-related metrics alongside latency.

The subsequent set of works represents a more comprehensive approach to performance metrics, as they address

Table 3
Performance Metrics.

Work	Availability	Reliability	Energy	Latency
[4]	✗	✗	✗	✓
[5]	✗	✓	✗	✗
[6]	✗	✓	✗	✓
[15]	✗	✗	✗	✗
[16]	✗	✗	✗	✗
[13]	✓	✓	✗	✗
[9]	✗	✓	✗	✓
[10]	✗	✓	✗	✓
[12]	✗	✗	✗	✓
[11]	✗	✗	✓	✓
[8]	✗	✗	✓	✓
[7]	✗	✓	✗	✓
[17]	✓	✓	✓	✓
[18]	✓	✗	✓	✓
[19]	✓	✓	✓	✓
This Work	✓	✗	✓	✓

three or more metrics. For example, [18] evaluates availability, energy, and latency. Similarly, [17] and [19] consider all these three metrics. In our work, we evaluate three performance metrics, excluding reliability, which is adopted as an input parameter (failure rate). Its value is consequently reflected in the system performance when homogeneous virtualization technology is employed. Therefore, our focus is on availability, power consumption, and response time, considering both eMBB and URLLC service types. Table 3 summarizes the related works based on their performance metrics.

3. System Model

Analytical models can be valuable tools for efficiently evaluating large-scale distributed MEC infrastructure projects since simulation and testbeds may not always be feasible. This study proposes an analytical model to analyze resource allocation, dimensioning, and configuration of edge computing systems based on MEC and NFV technologies hosting eMBB and URLLC services. The model enables evaluation of the impact of service and node parameters, as well as the overhead introduced by virtualization, on both services and system performance. Fig. 1 illustrates the modeled system, where both eMBB (blue flow) and URLLC (red flow) packets originating from UEs are processed by the RAN, passed on to the MEC node and are handled by containerized VNFs, which are scaled accordingly. This model was designed in isolation from the RAN, Core, and Central Cloud, hence, the only uncertainty is related to the virtual components themselves, i.e., setup, failure, and repair events.

The system consists of a finite amount of containers and buffer positions that can be allocated to each type of service, eMBB or URLLC. In our model, each VNF runs equally and independently on a single container, and a centralized control unit determines if requests are admitted. A request admission occurs if there are enough resources, i.e., if either containers or buffer positions are available, thus, if admitted, each request may be processed or queued.

With regards to the auto-scaling mechanism, a dynamic VNF auto-scaling strategy was embedded into our formulation to cope with the sudden load increase caused by the intensive request periods. Thus, before the proper processing phase, the containerized VNF must be initialized, which incurs a delay (setup time). In addition, the possibility of failure during service and its respective repair time is also embedded in our formulation. In this case, the containerized VNF is restarted, and the request is either reallocated to another available container or, if there are no available resources, it is placed back in its respective service queue with higher priority than new requests. In both cases, the service processing is restarted.

In terms of service prioritization, the following policy has been adopted: (1) if there are both URLLC and eMBB services to be served, URLLC services have higher priority, thus, the containers that are being released or activated are allocated first to URLLC services. (2) In the case where there is a URLLC service waiting in queue for available resources and an eMBB service has been completed, the released container is restarted to be used by the URLLC service. However, if there are other available containers, the current one will be allocated to a sequential eMBB service or deactivated if the eMBB queue is empty. (3) The preemption of the lower-priority service (eMBB) that is being processed is not allowed.

Since MEC-NFV systems are inherently stochastic, involving random events like failures, setup and repair times, dynamic workloads, and resource allocation variations, we have adopted CTMCs to model the system. CTMCs are particularly well-suited for capturing these random processes, as they model state transitions based on probabilistic rules, helping to analyze their impact on availability, latency, and power consumption. Unlike other formalisms, like Petri Nets, which typically rely on simulations for performance evaluation, the elegant structure of CTMCs allows for closed-form expressions, providing precise and analytical insights into useful metrics. Furthermore, CTMCs effectively represent the dynamic state transitions of containers (e.g., idle, setup, and busy) in dynamic resource allocation-based MEC-NFV systems. In our scenario, which involves the coexistence of uRLLC and eMBB services within the same MEC-NFV node, CTMCs offer a robust framework to analyze the impact of resource sharing, service prioritization, and various configurations on both system-level metrics (e.g., power consumption) and service-level metrics (e.g., response time and availability) [25].

Following the above description, the system is modeled using an M/N/c/k+K queue with two types of users, prioritization, failure, initialization time, First-Come-First-Served (FCFS) service discipline, and a limited buffer for each user type, k for URLLC and K for eMBB. The model states are represented by the tuple (i, j, l, m), where $i, j, l, m \in N$. Here, i and j denote the number of URLLC and eMBB services, respectively, and l and m represent the number

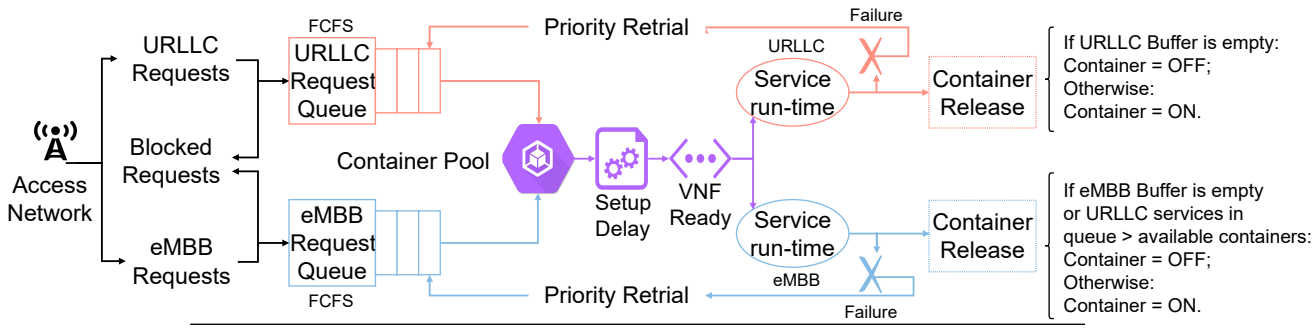


Figure 1: Edge Node Resource Allocation Flow.

of active containers for each user type, with $l + m$ being smaller or equal to the maximum number of containers (c). The service request arrivals follow a Poisson process with rate λ_u for URLLC services and λ_e for eMBB. The service processing is provided by the c available containers, with exponentially distributed service times, having rates μ_u for URLLC and μ_e for eMBB. Similarly, both failure occurrences and container initialization times follow exponential distributions with rates γ and α, respectively. The setup rate (α) denotes the number of container instantiations completed per unit of time. A higher setup rate indicates a faster VNF readiness and is the inverse of the setup time. Similarly, the failure rate (γ) represents the number of failures processing per unit of time. A higher failure rate indicates a less reliable system. Fig. 2 illustrates all possible CTMC transitions and states of the proposed system, along with its respective parameters. These states can be categorized into seven major types, as outlined below.

- A state that represents the empty system;
- States that indicate at least one user (URLLC or eMBB), no active containers, and the number of URLLC and eMBB users within the limits (k and K);
- States that denote at least one URLLC user being served and no eMBB users in the system;
- States that represent the system with at least one eMBB service being processed and no URLLC users in the system;
- States that represent the system with at least one URLLC service being processed and no eMBB users occupying containers;
- States modeling the system with users of both types, but only eMBB services being processed;
- States that describe the system with both and at least one service of each type being processed.

Table 4 summarizes these major state types, which encompass various subgroups and their associated conditions for the balance equations.

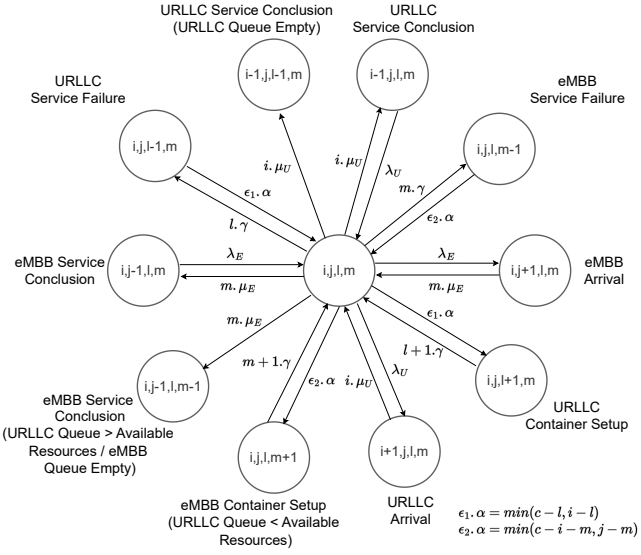


Figure 2: Generic CTMC State Diagram.

3.1. Performance Metrics

This section describes the steady-state analysis of the adopted CTMC, followed by the derivation of two performance metrics for each user type (eMBB and URLLC), namely Availability (A) and Mean Response Time (T) and also the mean Power Consumption (PC). In the following sections, $\pi_{i,j,l,m}$ denotes the steady-state probability of the state (i, j, l, m).

3.1.1. Availability (A)

The adoption of MEC and NFV environment near the UE is widely acknowledged for its potential to reduce latency and enhance reliability. However, the limited resources of edge nodes impose constraints on their service capacity, which is typically referred to as system availability. Consequently, when the maximum capacity is reached, two primary alternatives arise: forwarding the flow to a neighboring MEC node or redirecting it to the central cloud [26]. Both alternatives involve establishing a new route with multiple intermediate hops, introducing significant uncertainty concerning latency and reliability. Therefore, analyzing the availability of edge nodes becomes essential. This issue

Table 4
States Description.

State(s)	Condition(s)	Meaning
$(0,0,0)$	n/a	Empty system.
$(i,j,0,0)$	$(0 \leq i \leq k)$, $(0 \leq j \leq K)$ and $(i+j>0)$	System with at least one user and no active containers.
$(i,0,l,0)$	$(0 \leq i \leq k)$, $(0 < l \leq c)$ and $(k > c)$	System with at least one URLLC user being served and no eMBB user.
$(0,j,0,m)$	$(0 \leq j \leq K)$, $(0 < m \leq c)$ and $(K > c)$	System with at least one eMBB user being served and no URLLC user.
$(i,j,l,0)$	$(0 < i \leq k)$, $(0 < j \leq K)$ and $(0 < l \leq c)$	System with users of both types and at least one URLLC and no eMBB being served.
$(i,j,0,m)$	$(0 < i \leq k)$, $(0 < j \leq K)$ and $(0 < m \leq c)$	System with users of both types with at least one eMBB and no URLLC being served.
(i,j,l,m)	$(0 < i \leq k)$, $(0 < j \leq K)$ and $(0 < l,m \leq c)$	System with at least one user of each type being served.

is particularly crucial in the context of URLLC services, as compared to eMBB. While MEC availability remains important for eMBB applications, which focus more on delivering high data rates, the URLLC category places a stronger emphasis on meeting stringent latency and reliability requirements.

In our model, the MEC availability refers to the system's ability to offer the minimum amount of functional and accessible VNFs or buffer positions. Additionally, due to the service prioritization, the MEC node availability is segmented based on each service category, i.e., URLLC (A_U) and eMBB (A_E). These availabilities are described in Eqs. 1 and 2, which are derived by summing the probabilities of all states except those representing full capacity for each type of service.

$$A_U = 1 - \sum_{j=0}^{K} \sum_{l=0}^{c} \sum_{m=0}^{min(c-l,j)} \pi_{k,j,l,m} \tag{1}$$

$$A_E = 1 - \sum_{i=0}^{k} \sum_{m=0}^{c} \sum_{l=0}^{min(c-m,i)} \pi_{i,K,l,m} \tag{2}$$

3.1.2. Power Consumption (PC)

The computational power consumption is an important component of operational costs and must be factored into the service provider's resource planning to optimize the cost-performance trade-off. In our model, the mean power consumption ($\overline{PC}$) is determined by combining the mean number of virtual resources and energy consumption constants for each container's operating state: Setup and Busy. The power consumption (in Watts) of a single container in the setup state is denoted as P^{CT}_{setup} while in the busy state, it is represented as P^{CT}_{busy}. It is important to note that this power consumption metric is computed for the combined set of service categories.

The mean number of containers $\overline{CT}$ in each state (Busy and Setup) is defined by Eqs. (3) and (4) and are detailed in the next few lines. Eq. (3) captures the mean amount of containers in the busy state by iterating over each system state with service loads, and varying the combination of the number of each container type (URLLC and eMBB), from 0 up to the number of services of a particular category, or until the maximum available resources in the system. Moreover, Eq. (4) calculates the mean number of containerized VNFs in the setup state by iterating over system states where the number of online services exceeds the total number of available resources (containers) for each service category. Finally, the total mean power consumption ($\overline{PC}$) is given by Eq. (5).

$$\overline{CT}_{busy} = \sum_{i=0}^{k} \sum_{j=0}^{K} \sum_{l=0}^{min(c,i)} \sum_{m=0}^{min(c-l,j)} (l+m)\pi_{i,j,l,m} \tag{3}$$

$$\overline{CT}_{setup} = \sum_{i=0}^{k} \sum_{j=0}^{K} \sum_{l=0}^{min(c,i)} \sum_{m=0}^{min(c-l,j)} min[(c-l-m), (i+j-l-m)]\pi_{i,j,l,m} \tag{4}$$

$$\overline{PC} = P^{CT}_{setup} \overline{CT}_{setup} + P^{CT}_{busy} \overline{CT}_{busy} \tag{5}$$

3.1.3. Response Time (T)

The response time plays a vital role in URLLC applications, while also maintaining relevance for eMBB applications. Recognizing that the significance may vary depending on the service category, we have opted to analyze them separately, as denoted by Eqs. 8 and 9. The Response Time for each category is defined as the interval between the service arrival (at the edge node) and its completion, which includes any setup/restart times if these events are triggered during service attendance. The response time is obtained by calculating the mean number of online services in the system for each category, as shown in Eqs. 6 and 7, and then dividing them by the accepted service rate.

$$\overline{U}_U = \sum_{i=0}^{k} \sum_{j=0}^{K} \sum_{l=0}^{min(c,i)} \sum_{m=0}^{min(c-l,j)} i\pi_{i,j,l,m} \tag{6}$$

$$\overline{U}_E = \sum_{i=0}^{k} \sum_{j=0}^{K} \sum_{l=0}^{min(c,i)} \sum_{m=0}^{min(c-l,j)} j\pi_{i,j,l,m} \tag{7}$$

$$T_U = \frac{\overline{U}_U}{\lambda_U A_U}. \tag{8}$$

$$T_E = \frac{\overline{U}_E}{\lambda_E A_E}. \quad (9)$$

After describing the CTMC-based model for MEC-NFV systems supporting two service types, including practical aspects (e.g., container setup time, processing failures, and service prioritization) and deriving useful performance metrics, the next section presents the model validation to demonstrate its consistency in evaluating MEC-NFV systems.

4. Validation and Analysis

The analytical results were validated against extensive discrete-event simulations (Figs. 3a-17) using a Coloured Petri Nets-based simulator [27], where the lines denote the analytical and the markers represent simulation results.We have followed 3GPP Release 16 (TR 38.824) [1], which identified URLLC use cases in the factory automation, smart transportation, and electrical power distribution, to define the range of arrival rate values used in our scenarios. These values were adapted to our node scale to capture scenarios ranging from low to high user loads. With the exception of the first scenario (Section 4.1), which evaluates the impact of each user type on each other by adopting multiple eMBB request rates (λ_E), each subsequent scenario simultaneously assesses the influence of a pair of parameters: (Section 4.2) container setup rates (α) and failure rates (γ), which concomitantly analyzes the impact of hardware and software improvements (lower time to make the functions ready to process the services) and different levels of container reliability on the service performance; (Section 4.3) URLLC service rate (μ_U) and eMBB service rate (μ_E), with the objective to illustrate how enhancements in service process speed, achieved through the utilization of advanced processing units and optimized algorithms, for example, can positively impact the system's overall functionality; (Section 4.4) total number of containers (c) and the buffer size for eMBB users (K), which demonstrates how augmenting the parallel processing capacity of the system affects both its cost and the quality of service. Concomitantly, it also considers the implications of increasing the system's capacity to admit a higher number of eMBB services; and (Section 4.5) total number of containers (c) and the buffer size for URLLC users (k). This section shares a similar objective to the previous one but focuses on the impact of expanding the system's capacity to accommodate URLLC service requests.

In all scenarios, the URLLC service arrivals (λ_U) ranged from 2.5 to 25 requests/ms in order to analyze the system performance under different URLLC loads. Unless stated otherwise, the baseline values for failure (γ) and setup rates (α) were set to 0.001 and 1 unit/ms, respectively, in accordance with [28]. For container power consumption in different operation states, we adopted the values from the network-intensive experiment in [22], which are summarized in Table 5. The remaining parameters can be found in Table 6 with their descriptions in Table 7.

Table 5
Power Consumption Values.

Parameter	Value
Idle Container Energy Consumption (P^{CT}_{idle})	0 W
Setup Container Energy Consumption (P^{CT}_{setup})	8 W
Busy Container Energy Consumption (P^{CT}_{busy})	23 W

The following sections (4.1 - 4.5) display average results for each scenario. For each point, 10 simulation instances were conducted, with each instance comprising 27,000,000 simulation steps and 2,200,000 services processed. The simulations were performed on a computer equipped with an Intel Core i7-9750H 6-core processor (4.50 GHz), 16 GB of memory, and running the Windows 10 operating system. The Bootstrap method [29] was employed, with both resample size and the number of (re)samplings set at 30 and 1000, respectively. This was done considering a 95% confidence level, nonetheless, bars were omitted due to the negligible difference between upper and lower bounds and to prevent overcrowding the graphs.

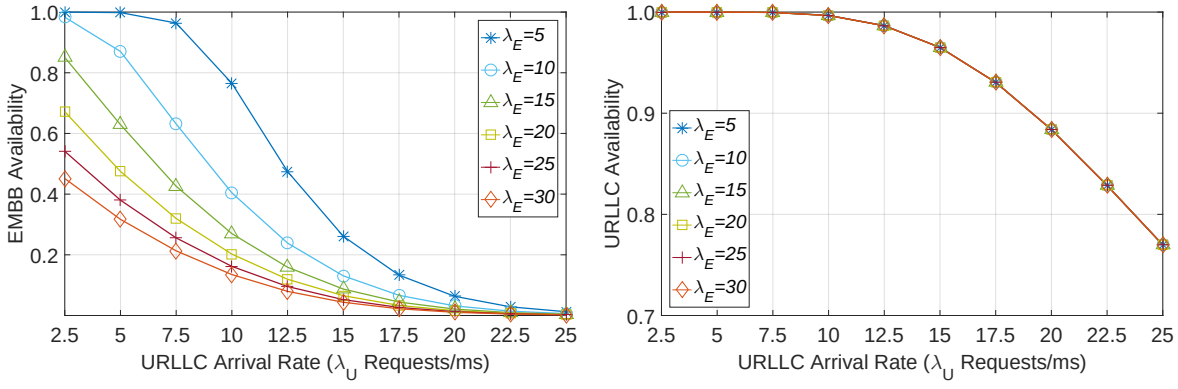
4.1. Effects of the eMBB load (λ_E)

The eMBB service may be seen as a natural evolution of the mobile broadband provided by LTE networks. This category typically exhibits varying user load demands over time and across different areas (e.g., urban, suburban, and rural). To analyze the impact of different eMBB loads coexisting with URLLC services in the same MEC-NFV node, this scenario varies the eMBB service request arrival rate, from 5 up to 30 arrivals/ms, resulting in six curves. They represent different eMBB loads, where the blue curves (light and dark) correspond to small loads (5 and 10, respectively), green and yellow to medium loads (15 and 20, respectively), and red and orange to higher loads (25 and 30, respectively).

Regarding the Availability of both eMBB and URLLC users, Figs. 3a-3b depict strictly decreasing curves. Notably, the Availability for eMBB users (Fig. 3a) displays a greater disparity among the configurations, whereas the results for URLLC users (Fig. 3b) exhibit overlapping patterns. This observation aligns with expectations, given that the URLLC service category is accorded higher priority over eMBB, rendering the eMBB arrival rate (λ_E) inconsequential for URLLC Availability. Conversely, in Fig. 3a, eMBB users contend for unoccupied containers, i.e., those not utilized by either eMBB or URLLC users. As the curves represent varying eMBB user loads, the overall eMBB Availability fluctuates, with higher values corresponding to curves indicating lower eMBB arrival rates (e.g., $\lambda_E = 5$ and $\lambda_E = 10$). Consequently, the curves in Fig. 3a exhibit a more pronounced decline compared to those in Fig. 3b, as the former is influenced by both eMBB and URLLC arrival rates while the latter is solely influenced by the URLLC arrival rate. Moreover, it is noteworthy that the eMBB user Availability (Fig. 3a) converges to zero at $\lambda_U = 22.5$, whereas the URLLC Availability (Fig. 3b) remains above 80% at the same point. These findings appear reasonable for

Table 6
Experiment Sets.

Section	Varying Parameters	λ_E	α	γ	μ_U	μ_E	c	K	k
4.1	λ_E	5,10,15,20,25,30	1	10^{-3}	2	2	10	20	20
4.2	α, γ	10	1,2,4	10^{-2}, 10^{-3}	2	2	10	20	20
4.3	μ_U, μ_E	10	1	10^{-3}	1,2,4	1,2	10	20	20
4.4	c, K	10	1	10^{-3}	2	2	4,8,12	16,24	20
4.5	c, k	10	1	10^{-3}	2	2	4,8,12	20	16,24



(a) eMBB. (b) URLLC.

Figure 3: Availability under Different URLLC (λ_U) and eMBB(λ_E) loads.

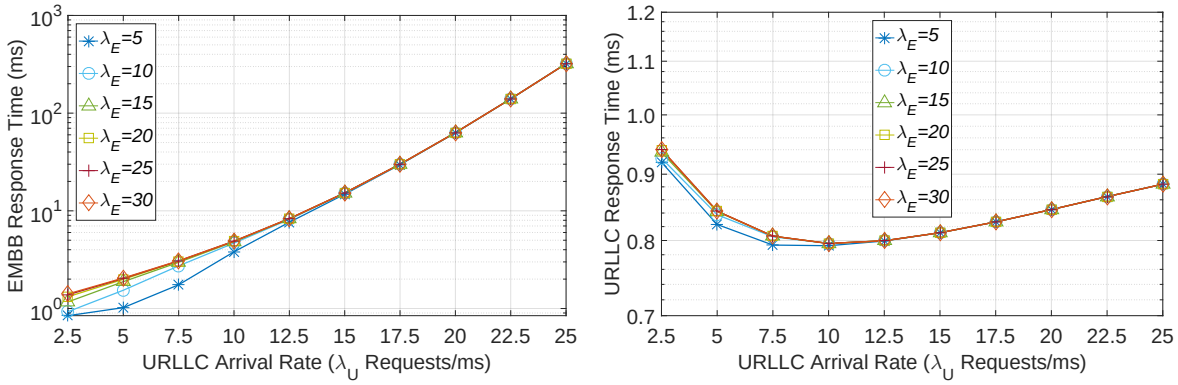
Table 7
System Parameters.

Parameter	Description
c	Maximum number of containers
k	URLLC buffer size
K	eMBB buffer size
λ_u	URLLC arrival rate
λ_e	eMBB arrival rate
μ_u	URLLC service rate
μ_e	eMBB service rate
α	Container setup rate
γ	Service failure rate

the majority of future service categories but are considered suboptimal for URLLC standards.

The Response Time (Figs. 4a-4b), showcase significant disparities, starting with the employed scale. In Fig. 4a, the Response Time for eMBB users exhibits a wide range of values spanning from 1 ms up to 300 ms. In contrast, Fig. 4b depicts a considerably narrower range, with the Response Time for URLLC users ranging from 0.8 ms to 0.94 ms, these values indicate that across all load scenarios assessed, the latency requirements for delivering all URLLC services listed in Table 8 are consistently met. Despite these distinctions, the curves in both figures exhibit substantial overlap across the majority of the evaluated points, converging to the same final value. However, the key distinction lies in their respective behaviors. In Fig. 4a, the curves demonstrate a monotonically increasing trend, while Fig. 4b displays a sudden drop in the Response Time for URLLC users until $\lambda_U = 10$. Beyond this, all curves resume an upward trajectory, converging to 0.89 ms at $\lambda_U = 25$, which is lower than the initial value of approximately 0.94 ms at $\lambda_U = 2.5$. This unexpected behavior can be attributed to the container setup delay, during which requests await the completion of container loading. At the beginning of the curves, the eMBB service loads are at least twice as high as the URLLC request loads. Consequently, the containers are predominantly occupied with eMBB services. As a result, when a URLLC service request arrives, it must wait not only for a container initialization but also for an ongoing eMBB service to complete, release the container, and allow it to be reconfigured for the URLLC request to serve the URLLC request. This explains the high URLLC response time under low URLLC load. However, as the URLLC service load increases, a greater number of VNFs remain instantiated for this service category, reducing the chance of eMBB accessing resources due to service prioritization. Consequently, the likelihood of URLLC service requests waiting for eMBB service completions and container reconfigurations decreases. Furthermore, the new URLLC request is more likely to be handled by a container that is already prepared for processing.

Due to this, all curves experience a decrease in Response Time from $\lambda_U = 2.5$ to $\lambda_U = 10$, followed by a steady increase. However, the Response Time values do not reach the same levels as for $\lambda_U = 2.5$, as all containers have already been initialized. Additionally, in Fig. 4b, slight variations in the results are observed between $\lambda_U = 2.5$ and $\lambda_U = 7.5$, attributed to the presence of eMBB users. These users also contribute to the (re)initialization of containers when an eMBB request is completed and immediately followed by



(a) eMBB. (b) URLLC.

Figure 4: Response Time under Different URLLC (λ_U) and eMBB(λ_E) loads.

Table 8
Examples of eMBB and URLLC Applications.

Work	Use Case	Latency	Category
[2]	Factory Automation	0.25 - 10 ms	URLLC
[2]	Smart Transportation	10 - 100 ms	URLLC
[2]	Robotics/Telepresence	1 ms	URLLC
[2]	Health Care	1 - 10 ms	URLLC
[30]	AR/VR 120 FPS	$<$ 8 ms	eMBB
[31]	Fixed Wireless Access	4 ms	eMBB
[32]	8K Video Streaming	$<$ 20 ms	eMBB
[33]	Smart Office	$<$ 10 ms	eMBB

a URLLC request, triggering a new container initialization, which explains the small differences in this interval.

The last performance metric for this scenario is the Energy Consumption (Fig. 5), which exhibited two different behaviors from $\lambda_U = 2.5$ to $\lambda_U = 10$: an increasing trend for part of the configurations ($\lambda_E = 5$ and $\lambda_E = 10$) and a decreasing for the remaining curves. This is due to the summation of the arrival rates of both user types, i.e., when the sum of the arrival rates is lower than the total processing container capacity, the curves tend to increase since the idle containers are being activated to meet newly arrived requests. Conversely, when these rates exceed the processing capacity of the system, a slight decreasing trend can be observed in the curves. This is attributed to the re-initialization of containers to prioritize URLLC requests. During container re-initialization, the containers spend more time in setup mode, which uses less energy compared to a processing state, thus resulting in lower energy consumption. The curves tend to converge as the arrival rate of URLLC requests increases, causing fewer eMBB requests to be served and subsequently reducing the number of container re-initializations for different service types. As the containers are no longer being reinitialized, they spend more time in the processing state, leading to a new increase in overall energy consumption.

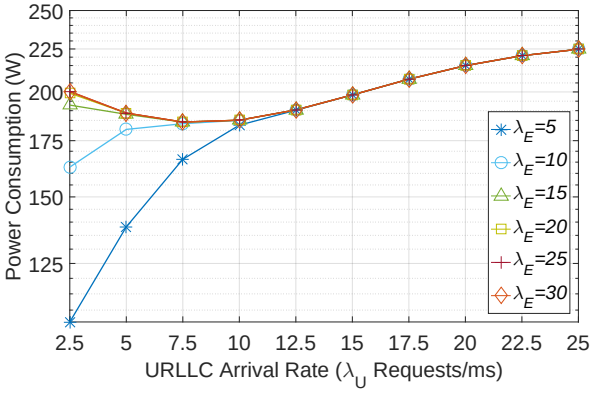
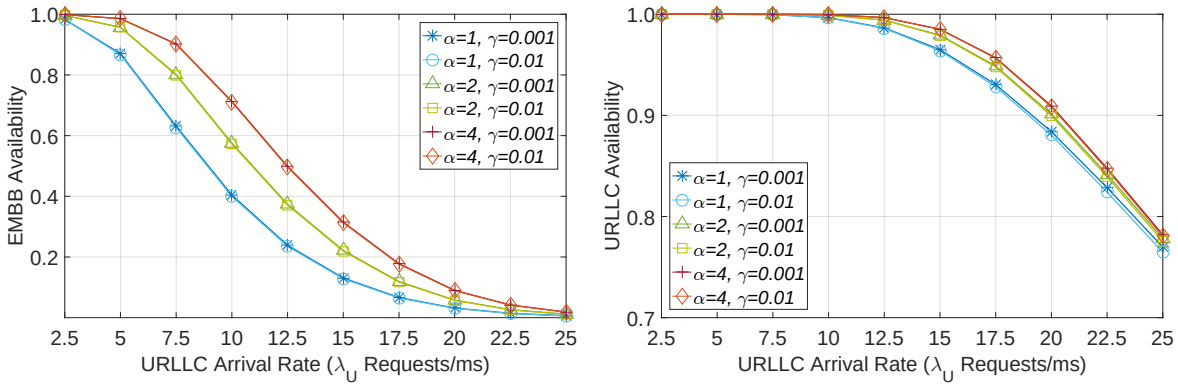


Figure 5: Power Consumption under Different URLLC (λ_U) and eMBB(λ_E) loads.

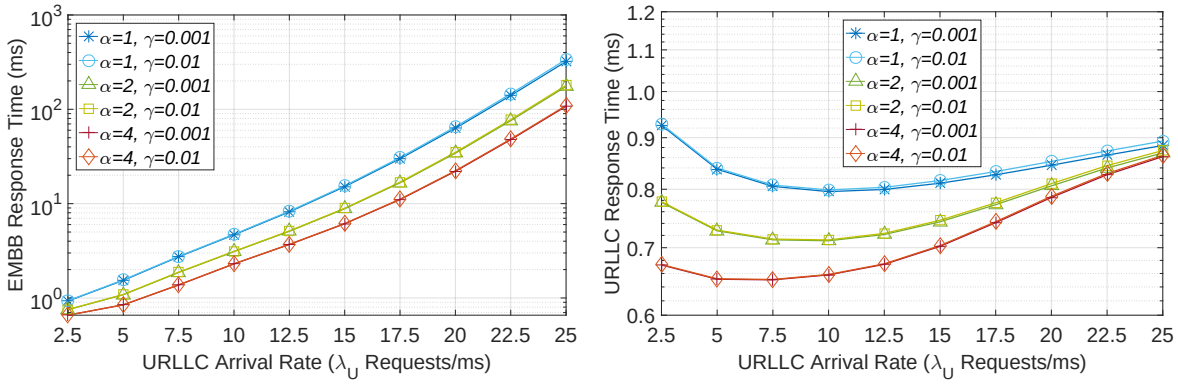
4.2. Effects of the container setup rate (α) and service failure rate (γ)

This section describes the impact of varying the container setup rate (α) in combination with changes in the service failure rate (γ). The availability of eMBB services, as in Fig. 6a, exhibited significant variations among the curves with different setup rates ($\alpha = 1$, $\alpha = 2$, and $\alpha = 4$), while overlapping with configurations having the same setup rate but different failure rates. Notably, the absolute differences in availability reached up to 30% for $\lambda_U = 10$ when comparing the $\alpha = 1$ (light and dark blue) and $\alpha = 4$ (red and orange) configurations. Higher container setup rates were observed to result in increased availability and reduced user waiting times in the buffer. Interestingly, the experiment revealed that even when the service failure rate was increased by a factor of ten, it did not significantly impact the system availability for eMBB users, which can be attributed to the buffer's capacity to accommodate failed service requests. Moreover, consistent with the previous scenario, the availability for eMBB applications diminished rapidly across all tested configurations, in contrast to the URLLC availability shown in Fig. 6b, which experienced a comparatively smaller impact due to its higher priority.



(a) eMBB. (b) URLLC.

Figure 6: Availability under Different Container Setup (α) and Failure (γ) Rates.



(a) eMBB. (b) URLLC.

Figure 7: Response Time under Different Container Setup (α) and Failure (γ) Rates.

Regarding the availability for URLLC users (Fig. 6b), it was observed that the container setup rate (α) had a relatively minor impact on the availability curves compared to the eMBB case. Specifically, the differences in availability among the curves with different α values were limited to approximately 2% at $\lambda_U = 15$, when comparing the $\alpha = 1$ (light and dark blue) and $\alpha = 4$ (red and orange) configurations. As for the impact of different failure rates, a more pronounced difference was noted when compared to the eMBB case in Fig.6a, where overlapping occurred. For the URLLC, container failures produced a slight difference among the curves with the same α, making it possible to distinguish between, for instance, the light and dark blue curves. In other words, the URLLC is significantly more sensitive to the failure rate than the eMBB.

When examining the eMBB Response Time depicted in Fig. 7a, it becomes apparent that a higher container setup rate leads to a reduced response time, as expected. Initially, since there is little competition for resources between eMBB and URLLC users, the difference between the evaluated configurations is of a few milliseconds. However, as the URLLC request arrival rate intensifies, this disparity becomes more pronounced. The increasing URLLC arrival rate creates a higher demand for resources, and since it has a higher priority, the eMBB requests are interrupted, either restarting service in another container or waiting in the buffer for available resources, causing the eMBB response time to be more affected. In such cases, only system configurations with α equal 4 has the capacity to handle high-resolution video streaming services, which demand a latency of under 20 milliseconds [32] when λ_U reaches 20 arrivals per millisecond. It was also noticeable that the failure rate had little impact in this experiment, which explains the pair of overlapped curves with the same values of α.

With regards to the Response Time of URLLC users (Fig. 7b), the container setup rate has a more pronounced impact compared to the previous scenario in Fig. 4a, where the only varying parameter was λ_E. This is particularly evident at the initial stages of the curves when containers are predominantly powered off or allocated to the eMBB users. During this period, the low arrival rate of URLLC services translates to shorter waiting times for a container to become available, reducing the overall response time. However, as the URLLC service arrival rate increases, this disparity diminishes, ultimately converging towards the end of the curves when the majority of containers are occupied by URLLC services.

Furthermore, it is noteworthy that a higher failure rate leads to an increase in the response time, since the failure occurrence becomes more frequent, especially for higher λ_U values, impacting the service time due to the need for container resets. However, similarly to the Availability in Fig. 6b, this remains relatively insignificant compared to the differences caused by altering the setup rate. This results in a more distinguishable difference among the pair of curves that were overlapping (e.g., light and dark blue). Finally, as the curves approach the system's capacity, a greater number of containers remain active to accommodate the incoming service requests, resulting in a temporary decline in the response time. Nevertheless, as resource competition intensifies within the URLLC service category, the response time gradually escalates once again and all curves tend to converge around 0.9 ms. At this point, all system configurations remain capable of providing service to robotic and telepresence systems, which require a latency of 1 ms [2].

Regarding energy consumption (Fig. 8), higher container setup rates, such as the green/yellow ($\alpha = 2$) and red/orange ($\alpha = 4$) curves, lead to greater energy consumption. This can be attributed to the fact that with higher setup rates, less time is spent in the setup phase, making containers more frequently available. Since the processing phase requires more power compared to the setup phase, the total energy consumption monotonically increases, converging around $\lambda_U = 25$ to 225 W. In brief, while higher container setup rates enhance both availability (Figs. 6a-6b) and response time (Figs. 7a-7b), they also contribute to higher energy consumption. Additionally, although the impact was small, it is worth noting that curves depicting higher service failure rates exhibit lower energy consumption when comparing the pair of curves with the same α (e.g., light and dark blue lines). This is due to the increased number of container resets for failed requests, leading to a higher proportion of containers in the setup state.

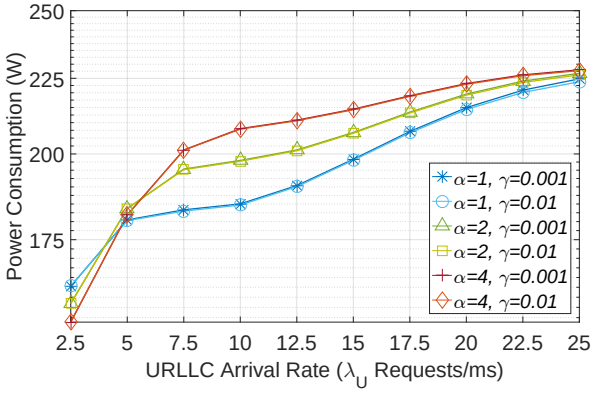


Figure 8: Power Consumption under Different Container Setup (α) and Failure (γ) Rates.

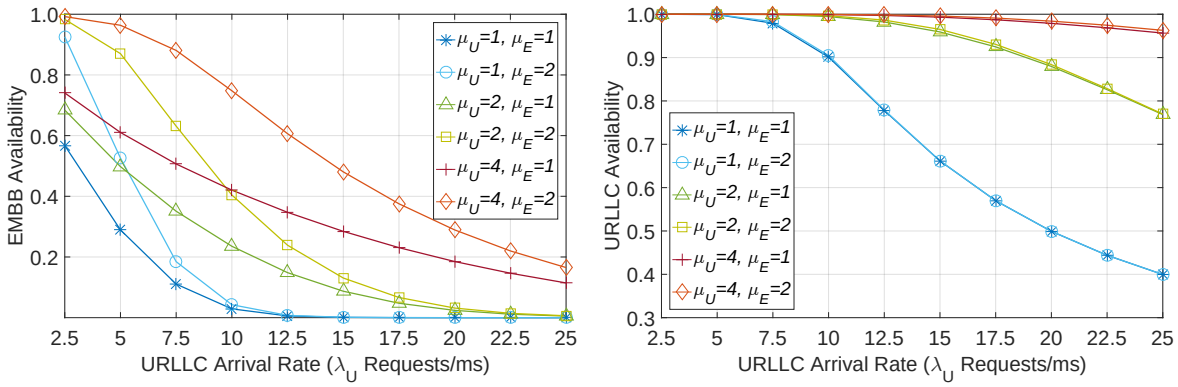
4.3. Effects of the URLLC service rate (μ_U) and eMBB service rate (μ_E)

This section assesses the influence of different service rates on each user type, specifically the URLLC service rate (μ_U) and the eMBB service rate (μ_E). Fig. 9a illustrates that a higher eMBB service rate leads to increased availability for this service category, particularly in the leftmost region of the graph. For configurations with the same μ_U values, the curve with $\mu_E = 2$ exhibits higher availability compared to those with $\mu_E = 1$. For example, at $\lambda_U = 7.5$, the configuration with ($\mu_U = 2$, $\mu_E = 1$) demonstrates an availability of 38%, while its counterpart ($\mu_U = 2$, $\mu_E = 2$) exhibits 62%, representing a significant difference of 24%. However, this effect diminishes as the URLLC arrival rate increases, resulting in convergence at the rightmost part of the graph. Moreover, a higher URLLC service rate implies less time spent by these requests monopolizing the resources, leading to greater availability. This explains why configurations with $\mu_U = 1$ and $\mu_U = 4$ are shifted to the left and right, respectively, compared to the adopted baseline ($\mu_U = 2$).

From the perspective of URLLC user availability (Fig. 9b), it is observed that the eMBB service rate (μ_E) has an insignificant impact on this performance metric, resulting in overlapping curves. Conversely, higher URLLC service rates ($\mu_U = 2$ and $\mu_U = 4$) lead to greater availability as the requests are serviced more rapidly. For instance, at $\lambda_U = 20$, configurations with $\mu_U = 1$ (light and dark blue) exhibit an availability of approximately 50%, while those with $\mu_U = 2$ (green and yellow) achieve around 88%, i.e., a substantial difference of 48%.

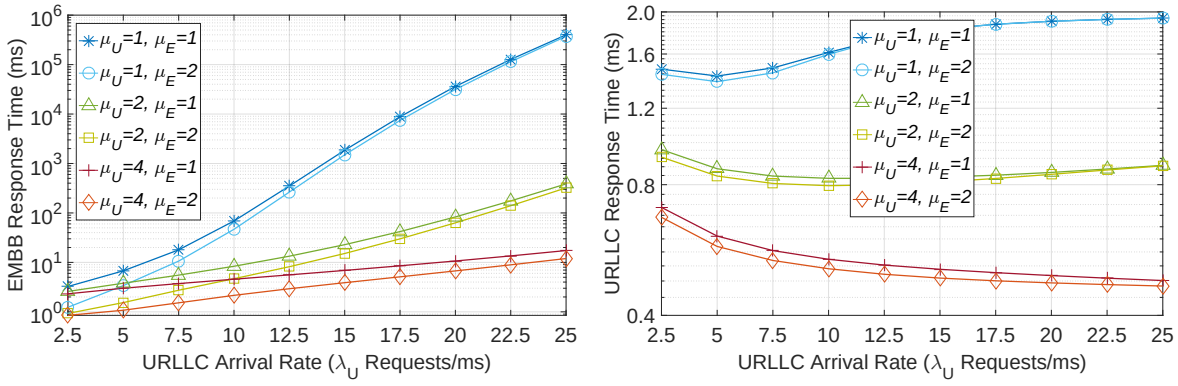
Regarding the eMBB response time in Fig. 10a, the experiment demonstrates that a higher service rate for this category, represented by configurations where $\mu_E = 2$ (light blue, yellow, and orange lines), results in shorter response times compared to their respective counterparts with $\mu_E = 1$ (dark blue, green, and red lines). However, the performance difference between the two curves with $\mu_U = 1$ (light and dark blue) and the two curves with $\mu_U = 2$ (green and yellow) is minimal. Notably, the performance difference becomes more pronounced for configurations with $\mu_U = 4$ (red and orange lines). These configurations consistently maintain the eMBB response time below 100 ms, a threshold considered crucial for multiple eMBB applications such as the Fixed Wireless Access (FWA) service.

Fig. 10b further reveals that a higher service rate for eMBB users, represented by configurations with $\mu_E = 2$ (light blue, yellow, and orange lines), also leads to shorter URLLC response times compared to their respective counterparts with $\mu_E = 1$ (dark blue, green, and red lines). This is attributed to eMBB requests spending less time occupying containers, which are then reinitialized to handle incoming URLLC requests. However, in most cases, this difference is below 0.1 ms and may not be significant even for URLLC applications. Conversely, an increase in the URLLC service rate ($\mu_U = 1$, $\mu_U = 2$, and $\mu_U = 4$) results in shorter response times for this service category, with a more substantial impact. For example, at $\lambda_U = 10$, the orange curve ($\mu_U = 4$, $\mu_E = 2$) shows a response time of approximately 0.5 ms, whereas the yellow curve ($\mu_U = 2$, $\mu_E = 2$) exhibits 0.8 ms. This 0.3 ms difference is significant for URLLC applications, as some require a response time of 1.2 ms



(a) eMBB. (b) URLLC.

Figure 9: Availability under Different URLLC (μ_U) and eMBB (μ_E) Service Rates.



(a) eMBB. (b) URLLC.

Figure 10: Response Time under Different URLLC (μ_U) and eMBB (μ_E) Service Rates.

or less, while others, such as Robotics and Telepresence, demand at most only 1 ms [2].

In configurations where $\mu_U = 1$ (light and dark blue lines), an interesting behavior is observed in Fig. 10b. As the URLLC request arrival rate approaches the system's processing capacity, a decrease in the response time for this service category is observed. This is attributed to URLLC containers spending more time active and less time in the setup state, thereby reducing the impact of this component. However, shortly thereafter, there is an increase in the response time due to competition for processing resources within the same service category, resulting from a larger number of URLLC requests waiting in the buffer. This behavior is also present in configurations with $\mu_U = 2$ and $\mu_U = 4$, but for larger $\lambda_U > 25$ values, which are not represented in this figure.

In terms of energy consumption (Fig. 11), once again, a higher service rate for eMBB users leads to lower energy consumption, especially in the leftmost region of the figure, corresponding to low URLLC loads, i.e., when the system is predominantly occupied by eMBB requests. This observation aligns with our earlier analysis on availability (Figs. 9a-9b), where configurations with $\mu_E = 2$ (light blue, yellow, and orange lines) outperform their respective counterparts with $\mu_E = 1$ (dark blue, green, and orange lines). In other words, higher availability corresponds to lower energy consumption. Consequently, the configuration order is inverted in Fig. 11, with the red and orange lines representing the most energy-efficient configurations.

In addition, when considering the three different configurations with $\mu_U = 1$, $\mu_U = 2$, and $\mu_U = 4$, significant differences of up to 40 W were observed. For instance, at $\lambda_U = 10$, the configuration with $\mu_U = 4$ and $\mu_E = 2$ (orange line) exhibits a consumption of approximately 175 W, while the configuration with $\mu_U = 2$ and $\mu_E = 2$ (yellow line) consumes around 215 W. This finding is particularly relevant as the experiment maintained the same amount of resources (containers) for all curves, varying only the service rates. In subsequent experiments, different resource and buffer amounts will be analyzed.

4.4. Effects of the number of containers (c) and eMBB buffer size (K)

This scenario evaluates the impact of varying the number of containers (c) concomitantly with the buffer size for eMBB users (K). In both Figs. 12a-12b, it is noticeable that the number of containers has a significant impact on the availability for both user service classes, showing higher

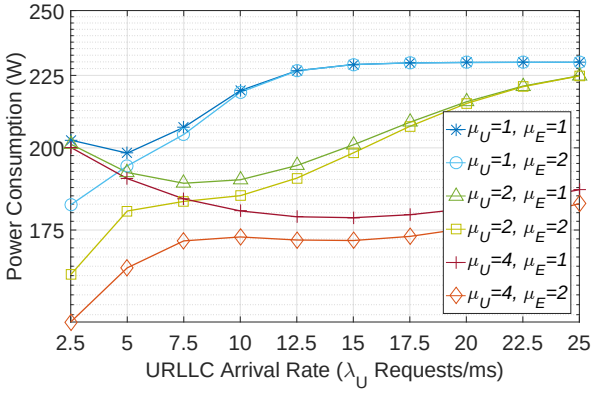


Figure 11: Power Consumption under Different URLLC (μ_U) and eMBB (μ_E) Service Rates.

availability for environments with a greater number of containers, represented by the configurations where $c = 12$ (red and orange lines), followed by $c = 8$ (green and yellow). For instance, in Fig. 12a at $\lambda_U = 10$, the availability for the configurations with $c = 8$ is approximately 20% whereas for the configurations with $c = 12$ is around 69%, i.e., a gap of almost 49%. On the other hand, the tested buffer alternatives had little impact on the eMBB availability, indicating that it would require much larger values than the adopted ones ($K = 16$ and $K = 24$). However, this is not feasible since the buffer will also impact the response time, which will be further evaluated. As for the system's URLLC availability (Fig. 12b), the analysis follows the same pattern for the eMBB, i.e., the container number drastically impacts the availability whereas the eMBB buffer sizes had barely any effect, resulting in overlapping pair of curves: light/dark blue, green/yellow, and red/orange.

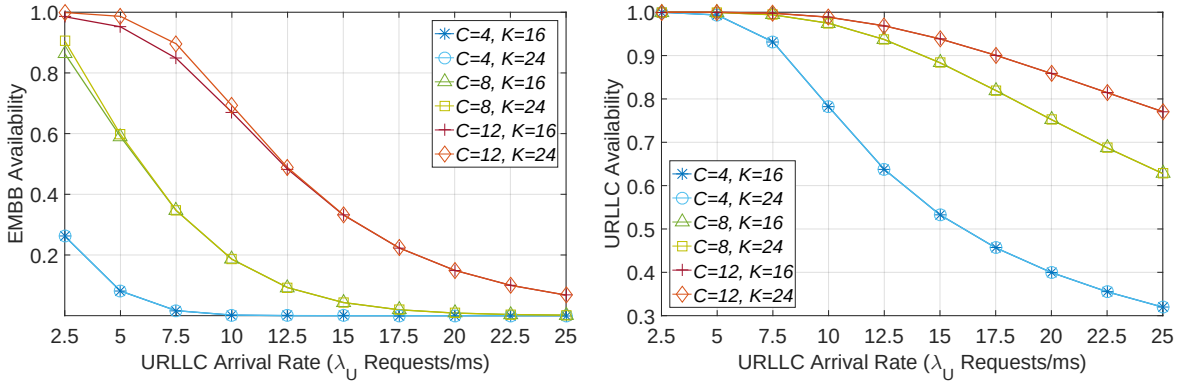
In Fig. 13a, a larger buffer size for the eMBB service category also increases in the proportional response time. This is due to the number of service requests ahead of each newly admitted eMBB request, which has to wait in queue. On the other hand, a greater number of available containers also implies a shorter queue time, reducing the contribution of this component to the response time. Once again, it can be observed that undersizing the number of containers can render the service unfeasible for lower-priority users, resulting in large response times, e.g., for configurations where $c = 4$ (light and dark blue lines), these particular configurations are suited for the Smart Office service, which requires a maximum latency of 10 ms [33], only when $\lambda_U = 2.5$. In contrast, the remaining configurations under evaluation can accommodate this application with a λ_U as high as 12.5.

Similar to the URLLC availability in Fig. 12a, varying the eMBB buffer size has also little impact on the URLLC response time in Fig. 13b. In other words, the response time is solely impacted by the variation in the number of containers. Only system configurations with $c = 8$ and $c = 12$ are capable of serving Robotics services, because even for $\lambda_U = 2.5$, which is the smallest evaluated in the experiment, configurations with $c = 4$ presented a response time greater than 1 ms. Despite this, the configurations with $c = 4$ presented a response time of less than 2 ms for all evaluated λ_U, proving to be capable of serving the Smart Transportation Systems service that allows latencies between 10 and 100 ms [2]. A particularity can be found on the leftmost part of this figure, where the curves with $c = 8$ (green and yellow lines) and $c = 12$ (red and orange lines) first decrease the response time, and, in the case of $c = 8$ it rises again, reaching the same initial value at $\lambda_U = 25$. This is likely due to the container setup time, which is either serving eMBB requests or powered off, considering the low URLLC demand from $\lambda_U = 2.5$ until $\lambda_U = 10$. On the other hand, as the URLLC arrival rate increases, a decrease in the response time of service requests can be observed. This happens since more containers become available, reducing the waiting time in relation to the container setup delay.

As opposed to the response time, a higher amount of containers inevitably implies a higher energy consumption (Fig. 14). The energy consumption is not exactly proportional to the increase in the number of containers; for instance, on $\lambda_U = 10$, between the blue and green curves, the number of containers doubles from 4 to 8, but the same does not occur with the energy consumption that increases by approximately 70%. This is because the number of containers being processed also depends on the workload that arrives in the system, that is, the energy consumption would only double together with the number of containers if the demand for service was sufficient to occupy all the containers available in the two configurations. However, there is very little difference in the energy consumption comparing each pair of configurations with the same container amounts, i.e., different buffer sizes. A larger eMBB buffer results only in slightly higher energy consumption because more users tend to wait in the queue. This prevents the container from being powered off and restarted, resulting in less setup time and more time in processing, consuming more energy.

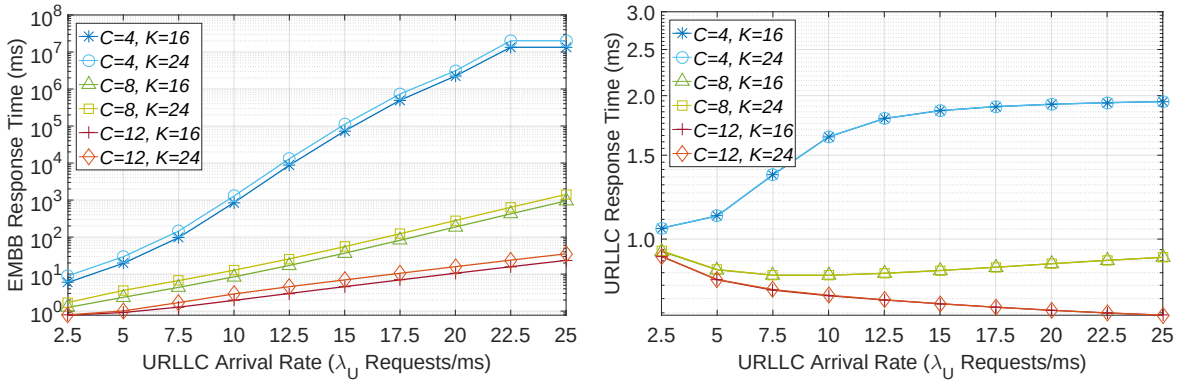
4.5. Effects of the number of containers (c) and URLLC buffer size (k)

This section evaluates the impact of the number of containers (c) along with the URLLC buffer size (k). With regards to the Availability (Figs. 15a-15b) we have very similar observations as those conducted in the previous scenario. However, in Fig. 15a it is noticeable that there is an inversion in the order of the curves (from the most to the least available), which is clearly shown by comparing the curves with $c = 12$ (red and orange). In this case, the red curve, which has fewer URLLC buffer positions ($k = 16$) presents a greater eMBB availability than the orange ($k = 24$). This happens because as more URLLC requests are stored, there is a guarantee that they will be serviced instead of dropped, as it happens with the curve with fewer URLLC buffer positions. Thus, the overall URLLC load increases, pressuring down the eMBB availability. Conversely, in Fig. 15b, the orange curve ($k = 24$) displays a greater availability than the red one ($k = 16$), which was expected since the evaluated metric is the URLLC Availability, i.e., a larger



(a) eMBB. (b) URLLC.

Figure 12: Availability under Different Amounts of Containers (c) and eMBB Buffer Sizes (K).



(a) eMBB. (b) URLLC.

Figure 13: Response Time under Different Amounts of Containers (c) and eMBB Buffer Sizes (K).

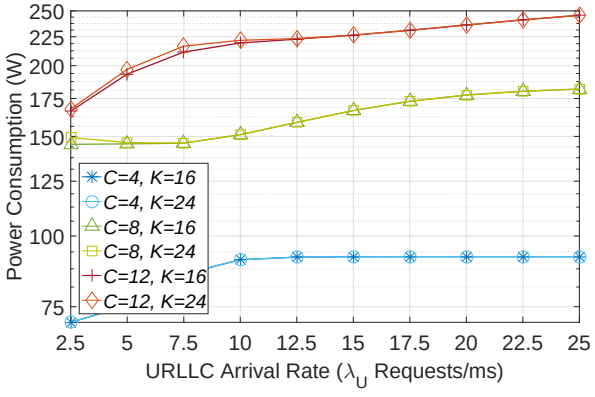
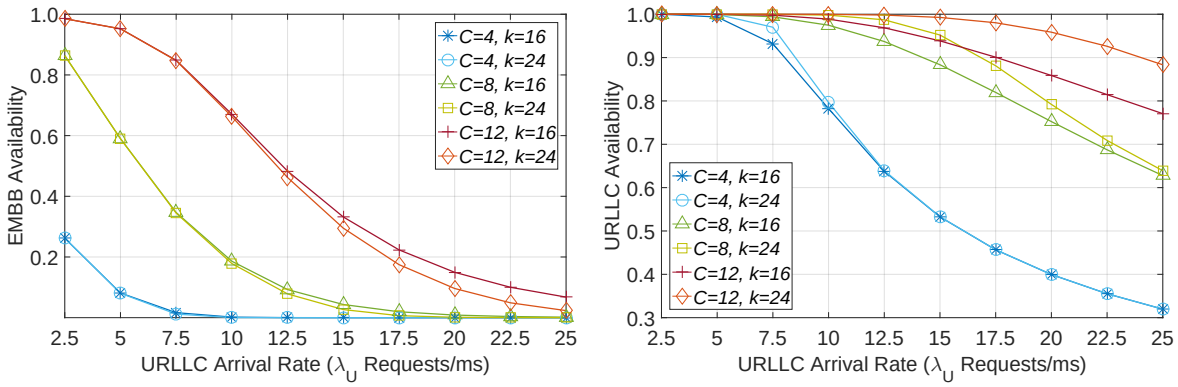


Figure 14: Power Consumption under Different Amounts of Containers (c) and eMBB Buffer Sizes (K).

URLLC buffer size enhances the URLLC availability, such that when the sum of the arrival rates for both service categories approaches the total processing capacity, a larger buffer size implies greater availability.

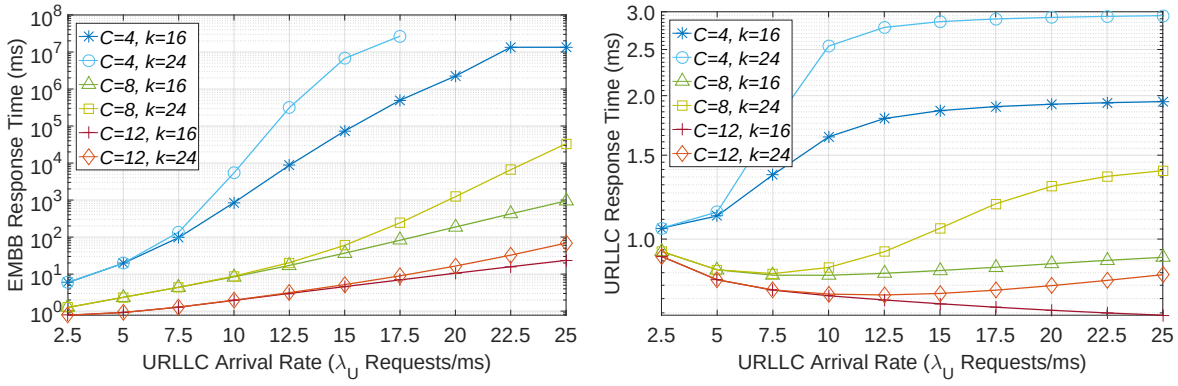
Regarding the response times depicted in Figs. 16a-16b, it is evident that the larger URLLC buffer size exerts a significant negative impact on both the eMBB and URLLC response times. However, this impact can be alleviated by increasing the total number of containers, as presented in the curves for both figures. At the leftmost part of Fig. 16a ($\lambda_U = 2.5$), the eMBB response time remains below 10 ms for all tested configurations, albeit with varying growth rates. For instance, the curves corresponding to $c = 4$ exhibit exponential growth, while those associated with $c = 12$ display linear increase. Consequently, higher container quantities result in improved eMBB response times, particularly under higher URLLC loads approaching system capacity. In this scenario, it becomes evident that system configurations with $c = 8$ and $c = 12$ can effectively fulfill the demands of Virtual and Augmented Reality services, which necessitate a latency of up to 8 milliseconds [34], for λ_U as high as 10, whereas configurations with $c = 4$ can only accommodate these services for lambda values equal to 2.5.

Conversely, larger URLLC buffers lead to degraded eMBB response times, as evidenced by the curves with $k = 24$ presenting higher response times compared to their respective counterparts with $k = 16$. Notably, a distinct characteristic observed in this experiment is that starting from $\lambda_U = 17.5$ and beyond, the light blue curve (representing $c = 4$ and $k = 24$) assumes unfeasible values (too high magnitude). This occurrence is likely attributed to



(a) eMBB. (b) URLLC.

Figure 15: Availability under Different Amounts of Containers (c) and URLLC Buffer Sizes (k).



(a) eMBB. (b) URLLC.

Figure 16: Response Time under Different Amounts of Containers (c) and URLLC Buffer Sizes (k).

the intensified pressure from URLLC arrivals coupled with the adoption of a large buffer size, resulting in an excessively large eMBB response time.

In Fig. 16b, the range of possible URLLC response time values is considerably lower than that of the previous experiment, owing to the higher priority accorded to URLLC requests. Nevertheless, there is considerable variation in the behavior of each curve. Some curves exhibit strictly ascending behavior, while others display both descending and ascending phases. Furthermore, one curve exhibits a strictly descending pattern. Nonetheless, the order of curves in terms of URLLC response time remains consistent with the previous experiment (Fig. 16a). It is worth noting that, for a larger interval of λ_U, the curves are expected to exhibit similar behavior with minor shifts. Regarding the light and dark blue curves, it can be inferred that the system capacity is swiftly reached, resulting in higher URLLC response times as the buffer becomes more heavily utilized. Nonetheless, even in these cases, the URLLC response time remains at an acceptable level of 3 ms, which is highly suitable for the majority of URLLC applications that typically require response times ranging from up to 10 ms, such as Factory Automation [2]. As for the strictly descending curve (in red), it is likely that the URLLC response time decreases because new URLLC arrivals are promptly processed by containers that were previously in the setup mode, thereby bypassing the setup delay. Additionally, the smaller buffer size ($k = 16$) leads to fewer requests in the waiting queue, thereby contributing to a lower overall URLLC response time compared to configurations with larger buffer sizes, such as $k = 24$ (represented by the orange line).

Regarding the energy consumption illustrated in Fig. 17 within this scenario, a similar observation can be made compared to the previous experiment. It is evident that an increased number of containers leads to higher energy consumption, aligning with our expectations. Moreover, for the majority of the evaluation frame, the size of the URLLC buffer exhibits minimal influence on this particular performance metric. This is evident from the overlapping pair of curves, particularly noticeable for low URLLC arrival rates. This outcome was anticipated since the buffered requests do not consume resources while in the queue. Thus, in most cases, the buffer size does not significantly impact the power consumption. However, a slight increase in energy consumption is observed when the system approaches full capacity and utilizes more buffer positions. This phenomenon occurs due to the containers spending a greater amount of time

in a processing state, resulting in reduced periods of being powered off or undergoing restart procedures.

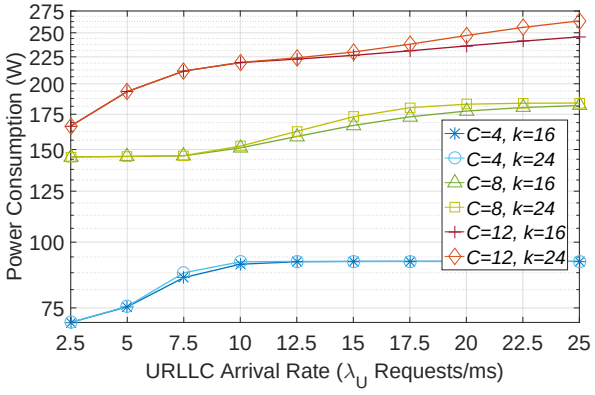


Figure 17: Power Consumption under Different Amounts of Containers (c) and URLLC Buffer Sizes (k).

5. Conclusions and Future Directions

This work investigated the interactions between MEC, NFV, and dynamic virtual resource allocation within the environment of 5G networks accommodating both URLLC and eMBB. The framework employs a CTMC-based model to describe the the dynamics of virtual resource allocation, guided by three performance metrics. In order to yield the model more practical, resource failures, service prioritization, and setup (repair) times were integrated, since they can incur significant impacts on the 5G applications' requirements. The resulting model provides valuable insights on how the MEC-NFV 5G network handles different service categories, applying service prioritization to efficiently allocate processing resources.

The proposed model can assist network operators in effectively dimensioning edge nodes to ensure the coexistence of URLLC and eMBB services. Some of our key findings include evidence that higher eMBB arrival rates lead to decreased availability and increased response times, while URLLC availability remains relatively unaffected. Container setup and failure rates significantly influence both availability and response times, with faster setup rates improving availability and reducing response times. The availability of URLLC services is primarily affected by URLLC service rates, while eMBB service rates impact eMBB availability. The number of containers also plays a critical role, enhancing both availability and response times, whereas buffer sizes primarily affect response times. Furthermore, power consumption showed minimal sensitivity to changes in buffer size while largely affected by the number of containers.

For future work, we advocate for multiobjective approaches to resource allocation problems within the MEC-NFV framework, with a particular focus on supporting the coexistence of eMBB and URLLC services. Additionally, efforts should be directed toward developing cost-effective strategies to optimize resource allocation further, addressing the evolving needs of 5G networks. Developing Artificial Intelligence (AI)-based solutions is a promising and natural next step, as the next generation of mobile communications is designed to be AI-native. Furthermore, since the models in the literature do not encompass all the features presented in our proposed model, a numerical comparative analysis could be conducted to highlight the semantic differences between our model, existing models, and real systems. Finally, future studies could extend the analytical model to include mMTC in the coexistence analysis or adopt alternative formalisms, such as Coloured Petri Nets (CPN), to address the potential high complexity of the Markov chain-based model.

Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by the National Council for Scientific, Technological Development (CNPq)- Project No. 433142/2018-9, and by the UFPE/ Propesqi via Edital Nº 06/2024. This research work was conducted in part in ICTFICIAL Oy and was supported in part by the European Union's HE Research and Innovation Program HORIZON-JUSNS-2023 through the 6G-Path Project under Grant 101139172. The paper reflects only the authors' views. The Commission is not responsible for any use that may be made of the information it contains.

Authors' contributions

This paper is equally contributed by each author. Besides, there were collaborative efforts in brainstorming the idea of this paper, proofreading, and formatting of this paper.

Conflict of interest

The authors declare there is no conflict of interest.

References

[1] 3GPP. System architecture for the 5g system (5gs). *White Paper*, 2020.

[2] Maraj Uddin Ahmed Siddiqui, Hanaa Abumarshoud, Lina Bariah, Sami Muhaidat, Muhammad Ali Imran, and Lina Mohjazi. Urllc in beyond 5g and 6g networks: An interference management perspective. *IEEE Access*, 11:54639–54663, 2023.

[3] Mehdi Setayesh and Shahab Bahrami. Resource slicing for embb and urllc services in radio access network using hierarchical deep learning. *IEEE Transactions on Wireless Communications*, 2022.

[4] Anupam Kumar Bairagi, Md. Shirajum Munir, Madyan Alsenwi, Nguyen H. Tran, Sultan S. Alshamrani, Mehedi Masud, Zhu Han, and Choong Seon Hong. Coexistence mechanism between embb and urllc in 5g wireless networks. *IEEE Transactions on Communications*, 69(3):1736–1749, 2021.

[5] Tianzhu Zhang, Han Qiu, Leonardo Linguaglossa, Walter Cerroni, and Paolo Giaccone. Nfv platforms: Taxonomy, design choices and future challenges. *IEEE Transactions on Network and Service Management*, 18(1):30–48, 2021.

[6] Yunbae Kim and Seungkeun Park. Calculation method of spectrum requirement for imt-2020 embb and urllc with puncturing based on m/g/1 priority queuing model. *IEEE Access*, 8:25027–25040, 2020.

[7] Haojun Huang, Wang Miao, Geyong Min, Jialin Tian, and Atif Alamri. Nfv and blockchain enabled 5g for ultra-reliable and low-latency communications in industry: Architecture and performance

evaluation. *IEEE Transactions on Industrial Informatics*, 17(8):5595–5604, 2021.

[8] Wei Li and Shunfu Jin. Performance evaluation and optimization of a task offloading strategy on the mobile edge computing with edge heterogeneity. *The Journal of Supercomputing*, 77(8), 2021.

[9] Zhou Tong, Tiankui Zhang, Yutao Zhu, and Rong Huang. Communication and computation resource allocation for end-to-end slicing in mobile networks. *2020 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1286–1291, 2020.

[10] Shengcheng Ma, Xin Chen, Zhuo Li, and Ying Chen. Performance evaluation of urllc in 5g based on stochastic network calculus. *Mobile Networks and Applications*, 26, 2021.

[11] Tong Liu, Lu Fang, Yanmin Zhu, Weiqin Tong, and Yuanyuan Yang. A near-optimal approach for online task offloading and resource allocation in edge-cloud orchestrated computing. *IEEE Transactions on Mobile Computing*, 21(8):2687–2700, 2022.

[12] Mahmoud Abdelhadi, Sameh Sorour, Hesham ElSawy, Sara A. Elsayed, and Hossam Hassanein. Parallel computing at the extreme edge: Spatiotemporal analysis. In *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, pages 5692–5698, 2022.

[13] Mustafa Emara, Hesham ElSawy, Miltiades C. Filippou, and Gerhard Bauch. Spatiotemporal dependable task execution services in mec-enabled wireless systems. *IEEE Wireless Communications Letters*, 10(2):211–215, 2021.

[14] Chunlin Li, Qianqian Cai, Chaokun Zhang, Bingbin Ma, and Youlong Luo. Computation offloading and service allocation in mobile edge computing. *The Journal of Supercomputing*, 77:1–30, 2021.

[15] Kun Xiong, Sebakara Samuel Rene Adolphe, Gordon Owusu Boateng, Guisong Liu, and Guolin Sun. Dynamic resource provisioning and resource customization for mixed traffics in virtualized radio access network. *IEEE Access*, 7:115440–115453, 2019.

[16] Guolin Sun, Kun Xiong, Gordon Owusu Boateng, Daniel Ayepah-Mensah, Guisong Liu, and Wei Jiang. Autonomous resource provisioning and resource customization for mixed traffics in virtualized radio access network. *IEEE Systems Journal*, 13(3):2454–2465, 2019.

[17] Marcos Falcao, Caio Souza, Andson Balieiro, and Kelvin Dias. An analytical framework for urllc in hybrid mec environments. *The Journal of Supercomputing*, 78, 2022.

[18] Caio Souza, Marcos Falcao, Andson Balieiro, and Kelvin Dias. Modelling and analysis of 5g networks based on mec-nfv for urllc services. *IEEE Latin America Transactions*, 19(10):1745–1753, 2021.

[19] Marcos Falcão, Caio Souza, Andson Balieiro, and Kelvin Dias. Dynamic resource allocation for urllc in uav-enabled multi-access edge computing. In *2023 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*, pages 293–298, 2023.

[20] Alejandro Santoyo-Gonzalez and Cristina Cervello-Pastor. Edge nodes infrastructure placement parameters for 5g networks. *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 1–6, 2018.

[21] Sami Kekki and Walter Featherstone. Mec in 5g networks. *ETSI White Paper*, (28):1–28, 2018.

[22] Roberto Morabito. Power consumption of virtualization technologies: An empirical investigation. *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, pages 522–527, 2015.

[23] David Lake, Ning Wang, Rahim Tafazolli, and Louis Samuel. Softwarization of 5g networks–implications to open platforms and standardizations. *IEEE Access*, 9:88902–88930, 2021.

[24] Miguel G. Xavier, Israel C. De Oliveira, Fabio D. Rossi, Robson D. Dos Passos, Kassiano J. Matteussi, and Cesar A.F. De Rose. A performance isolation analysis of disk-intensive workloads on container-based clouds. In *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 253–260, 2015.

[25] Leonard Kleinrock. *Theory, Volume 1, Queueing Systems*. Wiley-Interscience, USA, 1975.

[26] Ioannis Sarrigiannis, Kostas Ramantas, Elli Kartsakli, Prodromos-Vasileios Mekikis, Angelos Antonopoulos, and Christos Verikoukis. Online vnf lifecycle management in an mec-enabled 5g iot architecture. *IEEE Internet of Things Journal*, 7(5):4183–4194, 2020.

[27] Ali Shahidinejad, Mostafa Ghobaei-Arani, and Leila Esmaeili. An elastic controller using colored petri nets in cloud computing environment. *Cluster Computing*, pages 1–27, 2019.

[28] Kuljeet Kaur, Tanya Dhand, Neeraj Kumar, and Sherali Zeadally. Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers. *IEEE Wireless Communications*, 24(3):48–56, 2017.

[29] K. Singh and M Xie. Bootstrap: A statistical method, 2008.

[30] Mohammed Najah Mahdi, Abdul Rahim Ahmad, Qais Saif Qassim, Hayder Natiq, Mohammed Ahmed Subhi, and Moamin Mahmoud. From 5g to 6g technology: Meets energy, internet-of-things and machine learning: A survey. *Applied Sciences*, 11(17), 2021.

[31] Ericsson. Fixed wireless access on a massive scale with 5g, 2016.

[32] Yasuko Sugito, Shinya Iwasaki, Kazuhiro Chida, Kazuhisa Iguchi, Kikufumi Kanda, Xuying Lei, Hidenobu Miyoshi, and Kimihiko Kazui. Video bit-rate requirements for 8k 120-hz hevc/h.265 temporal scalable coding: experimental study based on 8k subjective evaluations. *APSIPA Transactions on Signal and Information Processing*, 9:e5, 2020.

[33] W. Stallings. *5G Wireless: A Comprehensive Introduction*. Addison-Wesley, 2021.

[34] Darijo Raca, Dylan Leahy, Cormac J. Sreenan, and Jason J. Quinlan. Beyond throughput, the next generation: A 5g dataset with channel and context metrics. In *Proceedings of the 11th ACM Multimedia Systems Conference*, MMSys '20, page 303–308. Association for Computing Machinery, 2020.