Flexible Network Slicing Assisted 5G for Video Streaming with Effective and Efficient Isolation*

Araf Arefia*, Rachid El-Azouzi*, Majed Haddad° and Essaid Sabir°

Abstract—Network slicing assisted 5G is driven by providing a wide range of services that aims to satisfy various pre-service requirements. There is consensus that network slicing is a key enabler for the service-oriented 5G, that aims to cope with the increasing complexity of these networks. One of the major objectives of network slicing is to provide a different level of resource isolation, through resource abstraction and virtualization and the ability to efficiently share network resources. In this paper, we focus on video streaming services in the presence of other services with different QoS requirements. We propose a novel approach for resource sharing that provides inter-slice protection, flexibility, load-driven elasticity, and network efficiency. In particular, we design two-level multi-scale allocation schedulers for an efficient and low complexity RAN slicing by exploiting the characteristic of adaptive traffic such as video streaming service. Our mathematical analysis and simulation results confirm the benefits of resource abstraction and exhibit the added value of our solution.

I. INTRODUCTION

Mobile devices have been generating 60% of all Internet traffic, and the Cisco Visual Network index predicted that traffic from various services such as video broadcast, live streaming would grow by a five-to-seven fold by 2022 [1]. With the recent pandemic, perhaps we have already reached such growth. The obvious reason is that millions of people are engaged in remote work, interactive online education, and entertainment, which are mostly video over IP, voice over IP (VoIP), live broadcast, and streaming traffic. YouTube/Netflix are already downgrading their QoS by throttling the streaming bit-rates to accommodate essential services.

With the future deployment of 5G, there has been a bunch of service-oriented ideas that have been explored aiming to innovate on the architectural front with several resulting benefits. The idea underlying this service-oriented view is for the network to support a wide range of services, differing significantly in their service requirements and device types. To meet the diverse industrial and market demands, the International Telecommunication Union (ITU) [2] identifies three broad classes of 5G service, i.e., Enhanced Mobile Broadband (eMBB), Ultra-reliable and low latency communications (URLLC) and Massive Machine-Type Communications (mMTC).

Following this trend, RAN slicing is appealing in the fact that it provides different levels of resource isolation, through resource abstraction, virtualization, and splitting among different tenants/services while guaranteeing the requirements of each service. This also improves scalability by reducing the complexity of resource allocation on multi-service systems. To do so, the software-defined networking (SDN) concept has been adopted for future deployment of RAN. SDN is meant to abstract physical eNodeBs in a geographical area as a logical big eNodeB. This has the advantage of simplifying the management of the operations within the RAN by providing infrastructure flexibility and service-oriented customization [3]. In a software-defined RAN, the SDN orchestrator handles all control plane decisions including network sharing. Building upon the 3GPP TSG SA 5G network sharing paradigm [2], a software-defined RAN architecture and its integration with network function virtualization enable RAN-only slicing that splits the physical RAN infrastructure into multiple virtual slices [3], [4]. The RAN slices can be customized for diverse service requests with various QoS and Quality of Experience (QoE) requirements.

Inspired by the attractive features and potential advantages of RAN slicing, their development and deployment have been gaining momentum in the wireless industry and research communities during the past few years [5], [6]. However, RAN slicing also comes with its challenges, and there are significant technical issues that still need to be addressed for successful rollout and operation of these networks. In particular, one shall introduce efficient mechanisms to provide different levels of isolation to a slice owner and the ability to customize its service processing across different planes, while efficiently utilize the available radio resources.

A. Related works

To enable RAN slicing, 3GPP specifies several realization principles in TR38.801 such as RAN awareness slicing, QoS support, resource isolation, and SLA enforcement among others. Many works aim to realize the isolation among slices through the static partitioning of all resources according to an SLA [7]. Its objective is to provide ideal protection to each slice. In [8], the radio resource scheduler is separated into intra-slice and inter-slice scheduling but no resource abstraction/virtualization is provided. In [9], authors propose an application-oriented RAN sharing mechanism to meet QoS requirements from applications but in a static sharing. Several studies in recent years have been proposed to dynamically allocate resources to slices based on the time-varying demands of slices [3], [6], [10], [11]. However, such algorithms introduce additional complexity, and may, in some cases, not ensure resource isolation. Hence, designing an algorithm to ensure service customization, while realizing timely adaptation
to network changes, is very challenging. The above studies show that there is a trade-off between service customization, resource allocation efficiency, and complexity. Authors in [11] provide an empirical study to quantify the advantage of the dynamic orchestration at different timescales. Authors in [6] show that substantial multiplexing gains can be attained by designing a proper radio resource slicing solution.

B. Contribution and organisation

In this paper, we explore the idea of a software-defined RAN where the RAN slices are specifically tailored to accommodate both QoS and QoE requirements for 5G services. We assume that the physical infrastructure is divided into multiple logical networks or slices, one per service instance. Each slice in such an architecture is an end-to-end virtualized network instance, spanning both the core network and the radio access network, and is tailored in terms of resources to meet the requirements of the service in question. We focus on video streaming services in the presence of other services that have different QoS requirements, which are defined through the SLA (service level agreement). While several techniques on slicing/virtualization of the infrastructure exist, the RAN slicing is still very challenging. Indeed, we shall design a practical solution for dynamic resource allocation that takes into account the number and locations of active users and allow for flexible sharing to meet the QoS/QoE requirements. For practical purposes, we propose a two-level scheduling process including intra-slice traffic scheduler and inter-slice scheduler, to ensure fairness among users belonging to the same slice and flexible sharing between slices that are able to adapt to network changes (traffic load, SLAs constraints, etc.). Moreover, we show the advantage of our dynamic orchestration at different timescales and determine in which cases the gains in efficiency can be obtained by exploiting the flexibility of traffic streaming in terms of QoE.

The remainder of the paper is organized as follows: in Section II, we present the system description including network slicing topology, virtualized resource mapping, and video streaming. In Section III, we provide the mechanism that allows reaching service requirements for both slices while ensuring complete resource isolation between both slices. Section IV presents the proposed scheduler, and Section V details the performance metrics for both slices. In Section VI, we provide the simulation results that show the performance of the proposed network slicing and validate our theoretical analysis. Section VII concludes the paper.

II. System description

We focus on a RAN slicing where we dynamically create and manage virtual resources in order to meet the requirements of each slice services and applications. In the proposed model, different services are supported by dedicated network slices and each can be operated separately with a customized scheduler.

We consider a collection of base stations, denoted $B$, shared by a set of network slices. Without loss of generality, we consider two slices: a video streaming slice and a second slice offering one of the three 5G applications, i.e., xMBB, uRLLC, and mMTC. Let $s_v$ and $s_u$ be the slice associated with the video streaming service and the other application, respectively.

A. Network slicing topology

The network slicing topology is composed of several bases stations $B$ shared by two slices $s_v$ and $s_u$. The RAN slicing provides partial/full functional isolation among deployed slices. Such an architecture reduces the risk of traffic overload generated by one service impacting negatively the performance of the other slice. Each slice can manage its allocated resources in a virtualized manner and each slice includes a dedicated scheduler to allocate the virtual resource to the attached users. We also consider a radio resource manager that efficiently handles resource allocation among co-located slices, while meeting the requirements requested by each slice. These requirements are defined through the SLA, which is a common agreement between the service operator and slices. However, in slice-based 5G networks, every slice needs an individual SLA, which would have unique elements, metrics, and structure in comparison to the SLAs of other slices within the same network.

Next, we highlight the proposed architecture for RAN slicing through the radio resource manager and the virtualization manager (see Fig. 1). This architecture is inspired by [3].

1) Slice Orchestration: The role of the slice orchestration module is to design strategies for slice scaling up/down so as the physical resources are efficiently shared among all the currently active slices. Yet, the orchestrator is in charge of deciding which actions need to be taken in the Cloud RAN (C-RAN). It also decides how the radio resources are allocated to active slices, and when/how these slices can be dynamically (re)configured in order to improve the overall resource usage. In section III, we detail the role of the slice orchestration module for video streaming services.

2) Virtualized resource management scheme: The main goal of the virtualized resource management scheme is to ensure radio resource isolation among slices in terms of physical network resources. To do so, we introduce an abstraction of radio resources, named virtual radio resource blocks (vRRBs),...
which is expressed as the number of OFDMA symbols allocated to a given slice. According to the agreed SLA, the virtual resource manager shall allocate a pool of vRRBs to each slice that can be used by mobile users attached to that slice. Each slice uses a customized scheduler, which is in charge of scheduling virtual resources for serving mobile users. The virtual resource mapper is in charge of accommodating the vRRBs to RRBS according to pre-assigned resources allocated to every single slice. In each subframe (TTI), the scheduler running at each slice receives the information needed to allocate the vRRBs to mobile users such as the Channel Quality Indicator (CQI) of attached UEs, the history, statistics of UEs’ traffic, etc.

3) Virtualized resource mapping: Virtualized Resource Mapper (VRM) is responsible for allocating the physical resource blocks to active slices flexibly and efficiently while ensuring slice isolation. The VRM allocates RRBS according to virtual resource allocation made by the schedulers of each slice and shall fulfill different types of SLAs. The radio resource allocation process occurs at each TTI and each slice will be allocated a set of RRBS according to the slice pre-assigned resource sharing, which provides a certain degree of protection among slices. This information is given by the SLA established between the slice owner and the service provider. It is understood that the dedicated bandwidth will be adapted according to the traffic intensity at each base station.

B. Video streaming as a slice

Most video streaming providers currently rely on HTTP Adaptive Streaming (HAS). A standard HAS is the Dynamic Adaptive Streaming over HTTP (DASH) issued by MPEG [12]. Under DASH, each video file is divided into multiple small segments and each segment is encoded into multiple quality levels. Based on the available capacity and bandwidth, the client dynamically chooses the quality level of the segment such that the visual quality is maximized while minimizing the probability of getting an empty playback buffer. HAS services are known to be greedy as the client always seeks to download the maximum number of segments with the highest quality. Thus, implementing adaptive video streams requires careful resource management in order to take into account such factors and overcome possible drawbacks, namely, (i) instability due to unnecessary switching of video bitrates [13], (ii) unfair resource allocation to video streaming traffic over other traffic types, and (iii) inefficient network utilization. Enabling network slicing seems to improve the whole chain. However, one major challenge is to improve the usage efficiency of infrastructure resources by scaling up/down each slice according to the variation of video streaming service. Such a goal can be met by orchestrating all physical base stations in a geographical area. Software-Defined Networking (SDN) concept brings infrastructure flexibility as well as adaptive rate throttling for video streaming services. To achieve better video quality, many metrics have been proposed to assess the QoE for streaming services. QoE metrics offer a means to describe, qualitatively and quantitatively, users’ perception of the quality of a video stream. The following list includes a few popular key performance indicators in adaptive video streaming services [14]: pre-fetching delay, probability of starvation, duration of starvation, average video bit-rate, and rate of video bit-rate switches.

III. Slice Orchestration for Video Streaming

In this section, we provide a mechanism aiming to guarantee a requested QoE/QoS for both slices $s_b$ and $s_v$, and, at the same time, to ensure complete resource isolation between both slices. While doing this, we shall pay particular attention to reach a suitable tradeoff between fairness among users and efficient resource utilization. Indeed, the network slicing is intended to dynamically share resources among slices while satisfying constraints on spectral efficiency and fairness. The proposed mechanism shall thus be built in compliance with the slices’ requirements and SLAs. To do so, we need advanced scheduling algorithms that allocate resources among these slices while respecting specific service requirements that have to be met on each slice, regardless of network congestion and performance levels of other slices.

We assume that the traffic intensity in the network is subject to variations over time. This could create severe cellular network congestion, especially during peak hours. However, the strict isolation between slices is not always the most effective approach to increase network resource utilization.

<table>
<thead>
<tr>
<th>TABLE I: Major notations used in the paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B$</td>
</tr>
<tr>
<td>$s_b$</td>
</tr>
<tr>
<td>$s_v$</td>
</tr>
<tr>
<td>$Q$</td>
</tr>
<tr>
<td>$w$</td>
</tr>
<tr>
<td>$\lambda_b^k$</td>
</tr>
<tr>
<td>$\pi_T$</td>
</tr>
<tr>
<td>$C_v$</td>
</tr>
<tr>
<td>$\alpha(w)$</td>
</tr>
<tr>
<td>$\beta_w$</td>
</tr>
<tr>
<td>$\mathcal{L}$</td>
</tr>
<tr>
<td>$\pi_w^u(n_{av})$</td>
</tr>
<tr>
<td>$Z_b^c$</td>
</tr>
<tr>
<td>$Z_b^c$</td>
</tr>
<tr>
<td>$N_{av}$</td>
</tr>
<tr>
<td>$U_i$</td>
</tr>
<tr>
<td>$\tau_a$</td>
</tr>
<tr>
<td>$\mu_i^u$</td>
</tr>
<tr>
<td>$\rho_i^u$</td>
</tr>
<tr>
<td>$\rho_v^u$</td>
</tr>
<tr>
<td>$G(u)$</td>
</tr>
</tbody>
</table>

To model the variation of traffic intensity, we assume that the arrival rate of users at a given bases station $b$ is time-varying and is modeled by a Markov modulated Poisson process (MMPP). In particular, we consider $m$-state continuous Markov chain (CTMC) with infinitesimal generator $Q = (\sigma_{ij})$, and $m$-state Poisson arrival rates $\lambda_1, \lambda_2, \ldots, \lambda_m$.
with \( \lambda_w = (\lambda_w^v, \lambda_w^b) \), where \( \lambda_w^b \) represents the arrival rate of users associated to slice \( z \in \{ u, v \} \) at base station \( b \). The element \( \sigma_{uv} \) is the transition rate from regime state \( w \) to regime state \( y \). We use the term “regime” state to describe the traffic intensity state in the system. Thus, when the system is in regime \( w \), the exogenous arrival rate on slice \( z \in \{ u, v \} \) at the base station \( b \) follows a Poisson process with intensity \( \lambda_w^{z,b} \).

We further make a distinction for traffic profiles on both slices. Since slice \( s_u \) corresponds to a video streaming service, its traffic can be seen as an adaptive flow, which is characterized by a fixed duration and a variable amount of transferred data that depends on network conditions. This covers a wide range of applications, including voice, video streaming, IoT monitoring, and real-time control. For slice \( s_v \), the traffic can be seen as an adaptive flow, which is characterized by a fixed amount of data corresponding to the file size requested (FTP, HTTP, IoT traffic, etc.).

Let us consider that flows of users associated with slice \( s_u \) (resp. \( s_v \)) have a size that follows an exponential distribution with mean \( \frac{1}{\lambda_w^u} \) (resp. \( \frac{1}{\lambda_w^v} \)) represented in seconds (resp. in bits). A departure from the system occurs when a user finishes its service.

In the sequel, we consider that the resources allocated to a slice’s users depend on the regime state \( w \). Let \( C_v \) be the set of vRRBs pre-assigned to slice \( s_v \), which represents the maximum resources that can be allocated to slice \( s_v \). Hence, at each regime state \( w \), the orchestrator needs to shift a fraction of resources, noted \( \alpha(w) \), of \( C_v \) to slice \( s_u \) to ensure the SLAs for slice \( s_u \).

A. QoE Optimization and Control

QoE optimization and control is a challenging task in network slicing due to many issues, including the heterogeneity of multimedia capability of users’ devices. The challenge is thus to develop a statistical-based resource allocation amongst slices that takes into account the QoE for streaming and QoSs requirements for the other slice. As stated in [13], the main challenge that arises with regards to QoE is to ensure the stability of video streaming with respect to video switches. This can be done by allocating the resource to each user in a way that its average throughput is one of the bit-rates furnished by the content provider. Thus the goal of the orchestrator is to decide on the resource allocation for each active user belonging to slice \( s_u \). It takes as the input the various video bit-rates for all users and their channel conditions and selects the appropriate target rate for each user. In what follows, we shall explain how this target rate can be optimally computed through two approaches: the first one is based on a Mixed-Integer Nonlinear Programming (MINLP) [13] and the second one is based on a heuristic approach. The value of the parameters \( \alpha(w) \) should be chosen in a way that prevents the violation of \( s_v \)'s SLAs.

B. Mixed-Integer Nonlinear Programming (MINLP)

The proposed scheme aims to determine in a dynamic setting how the resources are shared among slice by taking into account the traffic intensity, i.e., regime state.

At regime state \( w \), the orchestrator determines the target rate of each active user in slice \( s_u \) by assigning only a fraction \( 1 - \alpha(w) \) of resources in \( C_v \). Let \( L = \{ l_1, l_2, \ldots, l_m \} \) be the set of video bit-rates supported by the adaptive flows in slice \( s_u \), where \( l_1 \) is the minimum video bit-rate agreed upon in the SLA regarding the video streaming slice. Let \( N_{bv} \) be a random variable denoting the number of users on slice \( s_u \) at base station \( b \), and \( n_{bw}^{l_1}(\nu_{bw}) \) be the occupation measure, which gives the steady-state probability that the base station \( b \) has \( n_{bw} \) active users on slice \( s_u \) at the regime state \( w \). When there are \( N_{bv} \) users of slice \( s_v \) in base station \( b \), let \( Z_b^v \) denote the set of possible channel rates for user \( i \) on vRRB \( c \in C_v \). Then, the joint channel state space, at the base station \( b \), is given by \( Z_b^v = Z_b^1 \times Z_b^2 \times \ldots \times Z_b^N_{bv} \), where \( Z_b^i = \prod_{c \in C_v} Z_b^{c,i} \). Let \( N_{bv} \) denote the set of active users on slice \( s_v \) at the base station \( b \). Thus, the cardinalities of \( N_{bv} \) corresponds to a realization of the regime state \( w \).

At regime state \( w \), the problem of deriving the optimal target rate vector \( r_w \) is formulated as a mixed-integer nonlinear program (MINLP) as shown below:

\[
\text{MINLP} : \max_{(a(c,z), b) \in C_v, z \in Z_b^1, y_{ij} \geq 0, i \leq m} \left\{ \sum_{i=1}^{m} \alpha_i(c,z) : \sum_{i=1}^{m} \alpha_i(c,z) = \sum_{i=1}^{m} \lambda_i y_{ij}, \forall i \in N_{b,v}, b \in B \right\}
\]

Subject to:

\[
\sum_{i=1}^{m} \alpha_i(c,z) \cdot \lambda_i y_{ij}, \forall i \in N_{b,v}, b \in B \leq 1 - \alpha(w), \forall c \in C_v, \forall z \in Z_b^v, b \in B
\]

where \( n_{bw}^{l_1}(\nu_{bw}) \) is the number users in slice \( s_u \) that the orchestrator uses to compute the target rate, i.e one of the bit-rate in \( L \), \( U_{ij}(\cdot) \) is the utility function of user \( i \) which is assumed belonging to the class of \( \alpha \)-fairness measure, \( \alpha(w) \) is the probability of occurrence of the joint channel state \( z \), \( z_{ij}^{c} \) is the instantaneous channel capacity of user \( i \) on the vRRB \( c \). Note that \( z_{ij}^{c} \) captures both inter-cell and intra-cell interferences, \( \alpha_i(c,z) \) represents the fraction of frames where vRRB \( c \) is allocated to user \( i \) and \( y_{ij} \) is an indicator variable that takes 1 if and only if the segments streamed by user \( i \) are encoded using the bit-rate \( l_j \). Constraints (3) and (4) are intended to force the average throughput of each user to take values only within the set \( \mathcal{L} \). The constraint (1) ensures the existence of a utility that achieves an average throughput of \( \sum_{i,j} l_j y_{ij} \) for user \( i \).

If \( \{ a_i(c,z) \} \) for \( i \in N_{b,v}, c \in C_v, z \in Z_b^v, y_{ij}, i \in N_{b,v}, 1 \leq j \leq m \} \) is the optimal solution of the problem MINLP, then the target rate \( r_{bw}(i) \) of user \( i \) in \( N_{b,v} \) can be computed as \( r_{bw}(i) = \sum_{j=1}^{m} l_j y_{ij} \). Constraint (2) captures the maximum resource shared amongst users in slice \( s_u \). At regime state \( w \), a situation where \( \alpha(w) = 0 \) means that all resources pre-assigned to slice \( s_u \) could be used by all users of slice \( s_u \). This value is controlled by the orchestrator in order to
Indeed, at each regime, guaranty the possibility of the orchestrator to shift the fraction bit-rate switching. However, admission control is needed to session. This could decrease significantly the average video quality. Obviously, the value of the parameters $\alpha(w)$ shall be chosen in a way that prevents the violation of slice $s_v$’s constraints.

Since the target rate is computed by considering a fixed number of users at each base station $b$, the real active users change dynamically over time and of course, the fraction of resources shifted to slice $s_v$ depends on the number of active users. Moreover, if a set of base stations is overloaded, the target rate provided by the orchestrator is not admissible, hence admission control needs to be applied.

Below we consider admission control policies that adapt to change on load regime $w$. Specifically, an admission control policy for slice $s_v$ is parameterized by $(n_{bw}^w)_{b \in B}$, where $n_{bw}^w$ is the maximum number of active users that can be accepted on slice $s_v$ at base station $b$. Such decisions are assumed to be made independently thus admitted users for slice $s_v$ at base station $b$ still follow a Poisson Process with the rate $\lambda_{bw}^w$. Since the target rate is computed as a function of $(\tilde{n}_{bw}(w))_{b \in B}$, the resource that can be shifted to slice $s_v$ depends on the number of active users on slice $s_v$ at base station $b$. Let $r_i(w,n_{bw})$ be the target rate solution of MINLP by considering as input $n_{bw}$ instead of $\tilde{n}_{bw}(w)$. Note that, this target $r_i(w,n_{bw})$ is decreasing function on the number of active users. Thus when the number of active users is $\bar{n}_{bw}$, the fraction of capacity shifted to slice $s_v$ is approximated by:

$$\beta(n_{bw}) := \alpha(w) + (1 - \alpha(w)) \frac{r_i(w,n_{bw}) - \bar{r}(w,\tilde{n}_{bw}(w))}{r_i(w,n_{bw})}. \quad (6)$$

Hence the average fraction of resource that will be shifted to slice $s_v$ is given by:

$$\bar{\beta} := \alpha(w) + (1 - \alpha(w)) \sum_{n_{bw}=0}^{n_{bw}^w} \bar{\pi}_b(n_{bw}) \frac{r_i(w,n_{bw}) - \bar{r}(w,\tilde{n}_{bw}(w))}{r_i(w,n_{bw})}. \quad (7)$$

where $\bar{\pi}_b(n_{bw})$ is the stationary distribution that indicates the probability to have $n_{bw}$ users of slice $s_v$ at base station $b$ at regime $w$. Now setting $\bar{\beta} = \alpha(w)$, it is easy to see that:

$$\sum_{n_{bw}=0}^{n_{bw}^w} \bar{\pi}_b(n_{bw}) \frac{r_i(w,n_{bw}) - \bar{r}(w,\tilde{n}_{bw}(w))}{r_i(w,n_{bw})} = 0. \quad (8)$$

Thus the value $\tilde{n}_{bw}(w)$ used by the orchestrator in MINLP is computed using equation (8).

The advantage of this scheme is to ensure that each active user in slice $s_v$ will get a constant bit-rate $\bar{r}_i$ during its session. Indeed at each regime $w$, the target rate of each active user is unchanged even if there are arrivals or departures during its session. This could decrease significantly the average video bit-rate switching. However, admission control is needed to guaranty the possibility of the orchestrator to shift the fraction $\alpha(w)$ from slice $s_v$ to slice $s_u$ even if the number of active users changes over time.

If the SLA of slice $s_v$ imposed a target requirement $P_a$ concerning the probability to reject a new arrival user, thus the maximum number of active users that can be accepted on slice $s_u$ at base station $b$ is given by:

$$n_{bw}^w = \arg \max_{n_{bw}} \{P(N_{bw} \leq n_{bw}) = \sum_{n=0}^{n_{bw}} \pi_b^n(n_{bw}) \geq 1 - P_a\} \quad (9)$$

and thus the orchestrator determines the value $\tilde{n}_{bw}(w)$ that satisfies the condition (7). In practice, the above solution needs an accurate approximation of the capacity attained at each regime $w$ as well as the active number of users for each slice and at each base station. Although, in a practical system, it would introduce very high signaling overhead to realize channel estimation for all active users at every RRBs.

C. A Heuristic Approach for target rate assignment

Instead of computing an optimal target rate vector, we can dynamically throttle the average throughputs to sub-optimal target rates. This can be done by tracking the evolution of the average throughputs of all the users in slice $s_v$. To do this, we virtually run a scheduler at the level of the orchestrator to evaluate the average channel rate of each user belonging to slice $s_v$ when all resources pre-assigned to slice $s_v$ are used, which corresponds to $\alpha(w) = 0$. The orchestrator could use one of the well-known opportunistic scheduler called proportional scheduler. The orchestrator obtains the feedback of the instantaneous channel quality gain for each user in slice $s_v$ and each virtual resource block $c$ in time slot $t$. Then, it keeps track of the average throughput for each user $i$ and allocates resource blocks to users as follows:

$$i^*_v(t) = \arg \max_{i \in X_M} u_i^v(r_i(t)) \cdot z_{i,c}(t),$$

where $z_{i,c}$ is the instantaneous capacity of user $i$ on resource block $c$ in time-slot $t$, $i^*_v(t)$ is the user allocated to resource block $c$ at time slot $t$, and $r_i(t)$ is the average throughput of user $i$ till time $t$. The value of the average rate $r_i(t)$ is updated as follows:

$$r_i(t+1) = (1 - \alpha_t) \cdot r_i(t) + \alpha_t \sum_{c \in C} z_{i,c}(t) \cdot \mathbb{I}_{\{z_{i,c}(t)=1\}},$$

where $\mathbb{I}_{\{z_{i,c}(t)=1\}}$ is the indicator function for the event that user $i$ is allocated resource block $c$ at time slot $t$ by the Shadow scheduler. Here, $\alpha_t$ is the memory of the averaging filter. Since at regime state $w$, the orchestrator allocates a fraction $\alpha(w)$ of pre-assigned resources to slice $s_v$ to slice $s_u$ in order to guarantee the QoS requirement for slice $s_u$. Let us define $\bar{r}_i(M)$ as follows:

$$r_i(w,n_v,M) = \max\{l_j | l_j \leq r_i(w,n_v), 1 \leq j \leq m\}, \quad (10)$$

where $M$ is used by the orchestrator to calculate the target rate of all users of slice $s_v$ in order to free up $\alpha(w)$ of resources pre-assigned to slice $s_v$. Thus, the fraction of resources that can be allocated to slice $s_u$ as a function of $M$ is:

$$\beta_w(M) = \sum_{n_v} \pi_b^w(n_{bw}) \frac{r_i(w,n_v,M) - r_i((w,n_v),M)}{r_i(w,n_v,M)}.$$

Increasing $M$ leads to an increase in the portion of the resources that can be exploited by slice $s_u$. To free up $\alpha(w)$ of pre-assigned resources of slice $s_v$, the value $M_w$ must satisfy the following condition:

$$M_w = \min\{M | \beta_w(M) \geq \alpha(w)\}.$$
The target rate of each user at regime state \( w \) is given by (10) with \( M = M_v \).

IV. SCHEDULING PROCESS AT BASE STATION

A scheduler is located at slice \( s_u \) to allocate the virtual resource vRRBs to all active users on slice \( s_u \) at base station \( b \). It is in charge of executing the configuration sent by the orchestrator, i.e., schedule the users on slice \( s_u \) per time slot according to their instantaneous channel rate and calculate the scheduling priorities of users in order to achieve these target rates sent by the orchestrator. Overall, the functionality of the scheduler design is summarized in Algorithm 1.

We first update the target rate \( \bar{r}_i \) of each active user \( i \in \mathcal{N}_{b}\) and their instantaneous channel rates (line 2 et 3). Lines 4-6 allocate the vRRB in \( C_u \) only to users that their average throughput is less than their target rate provided by the orchestrator. This imposes all active users to be close to their target rate. In doing so, all unused pre-assigned vRRBs of slice \( s_u \) will be shifted to slice \( s_u \). When the network regime is in state \( w \), the scheduler of slice \( s_u \) can thus shift \( \alpha(w) \) fraction of resources in \( C_u \) to slice \( s_u \).

V. PERFORMANCE EVALUATION

In this section, we study the expected performance experienced by users of both slices. For video streaming services, we focus on four metrics to measure the QoE for video streaming service: the probability of starvation, average video bit-rate, average video bitrate switching, and start-up delay. For slice \( s_u \), we focus on some metrics such as average throughput, average delay and average Bit transmission delay.

A. Analysis of QoS metrics for video streaming slice \( s_u \)

The dynamic of slice \( s_u \) at each base station can be modeled by a continuous-time Markov chain (CTMC). Here we characterize the dynamics of the system at a given regime state \( w \). Let \( r_w(w, n_{bh}) \) be the average rate of a user on slice \( s_u \) at base station \( b \) when the state is in \( (w, n_{bh}) \). Given the assumption of exponentially distributed video size, the service time of a mobile user is also exponentially distributed. This implies that the departure of a mobile user given the current state at time \( t \) is independent of the past. Under the above-mentioned assumptions, the dynamics of coexisting mobile users on slice \( s_u \) at base station \( b \) can be depicted as a continuous-time Markov chain (CTMC) with state-space \( \mathcal{X}_{bh} \). The slots for the scheduler at the base station are in milliseconds, whereas the video segment playout happens in seconds. The timescale of the user arrival and departure process is in hundreds of seconds. Therefore, the scheduler dynamics happen at a much faster timescale than the video segment playout and the user arrival and departure dynamics. Due to this timescale separation, the slot-wise variations in the channel rate and users’ average channel rate between the state transitions are negligible. Thus, we can then assume that when the system state is \( (w, n_{bh}) \) the average channel rate of a user is \( r_w(w, n_{bh}) \).

Let us denote the transition rate matrix for the CTMC, \( \{n_{bh}(t) \in \mathcal{X}_{bh}\} \) by \( Q_{bh}(w) \). Now, \( Q_{bh}(n_{bh}, n_{bh} + 1) = \lambda_{bh}^{ub} \) and \( Q_{bh}(n_{bh} - 1, n_{bh}) = n_{bh} \theta_u \). The CTMC, \( \{n_{bh}(t) \} \) is a finite irreducible Markov chain and hence has a unique stationary distribution. Let us denote this distribution at regime state \( w \) by \( \pi_w \triangleq \{\pi_w(n_{bh}) \colon n_{bh} \in \mathcal{X}_{bh}\} \).

Given the regime state on the network, we shall evaluate QoS metrics seen by typical (i.e., randomly) users on slice \( s_u \), i.e., averaged over the stationary distribution of the network state and transition from \( n \) to \( n + 1 \). Due to lack of space, we omit all analysis for computing the metrics to measure the QoE for video streaming service: the probability of starvation, average video bit-rate, average video bitrate switching, and start-up delay: all the details can be found in the technical report [15].

B. Analysis of QoS metrics for slice \( s_u \)

We shall study the QoS metrics for slice \( s_u \). Let \( R^{ub}_w \) be a random variable denoting the rate of a typical user on slice \( s_u \) at base station \( b \). The performance of users on slice \( s_u \) depends on the dynamics of users in slice \( s_u \). In what follows, we propose an approximation in which users on slice \( s_u \) at the base station \( b \) generates a traffic volume of \( \lambda_{bh}^{ub} \) and shares a set of virtual resource blocks \( \mathcal{C}_u(w) \cup \mathcal{C}_u \), where \( \mathcal{C}_u(w) \) is the set of virtual resource blocks that were shifted to users on slice \( s_u \) and \( \mathcal{C}_u \) is the set of virtual resource blocks pre-assigned to slice \( s_u \). Without loss of generality, we assume that all users on slice \( s_u \) at base station \( b \) are statistically identical users in terms of flow sizes, arrival rates, and channel statistics. Let \( R^{ub}_w \) be the average throughput when there is a single user on slice \( s_u \) at base station \( b \). We define \( \rho^{ub}_w = \frac{\lambda_{bh}^{ub}}{R^{ub}_w} \). Under a given scheduler used at slice \( s_u \), let \( G(n) \) be the gain in throughput in comparison to a channel oblivious round-robin scheduling. Since the relative scheduling gains are identical for all slices then the throughput received by a user on slice \( s_u \) at base station \( b \) is \( r(n) = \frac{G(n)}{\rho^{ub}_w} \). This gain function will be increasing, reflecting the fact that the total throughput

Algorithm 1 scheduling policy of slice \( s_u \)

**Input:** target rate of active users on slice \( s_u \)

**Output:** User-resource virtual block allocation for each time slot \( t \geq 1 \), i.e., \( \{\mathcal{T}_w(t), c_w \in C_u\} \) and a part of virtual resource block set of \( C_u \) that can be shifted to slice \( s_u \).

1. for all \( i \in \mathcal{N}_{bh} \), initialize \( r_i(0) = \gamma_i(0) = 0 \) for new arrival users of slice \( s_u \).
2. for time slot \( t \geq 0 \)
3. obtain \( z(t) = \{z_i,c(t), c \in C_u, i \in \mathcal{N}_{bh}\} \) — the instantaneous channel capacity vector
4. for each virtual resource block \( c_w \in C_u \) do
5. if \( \max_{i \in \mathcal{N}_{bh}} (\bar{r}_i(t) - \gamma_i(t)) \geq 0 \) then
6. \( t_c(t) = \arg \max_{i \in \mathcal{N}_{bh}} (\bar{r}_i(t) - \gamma_i(t)) \cdot z_{i,c(t)}(t) \)
7. end if
8. Virtual resource block \( c_w \) will be shifted to slice \( s_u \).
9. end for
10. for \( i \in \mathcal{N}_{bh} \) do
11. \( \gamma_i(t + 1) = (1 - a_i) \cdot \gamma_i(t) + a_i \cdot \sum_{c \in C_u} z_i,c(t) \cdot \mathcal{T}_w(t - 1) \)
12. end for
13. end for
gains increase with the degree of multi-user diversity. It has its limit \( G^* = \lim_{n \to \infty} G_u(n) \). Hence the departure rate of users on slice \( s_u \) at base station \( b \) is \( \rho_w^{b,s_u} = R_w^{b,s_u} G_u(n) \theta_u \), where \( n \) is the number of active users on slice \( s_u \) at the base station \( b \). Under stability condition \( \frac{\lambda^b}{\rho_w^{b,s_u}} < 1 \), the probability of having \( n_u \) users on slice \( s_u \) at base station \( b \) is:

\[
\pi_w(n_u) = \eta \left( \frac{\rho_w^{b,s_u}}{G_u(n)} \right)^{n_u} \left\{ \prod_{j=1}^{n_u} G_u(j) \right\}^{-1}
\]

where \( \eta \) is a normalization constant.

As for the approximation performance metrics, the average throughput, the average bit transmission delay, and latency, they can be computed with newly calculated stationary distribution \( \pi_w(n_u) \) as follows:

- Average throughput:

\[
\mathbb{E}(R_w^{b,s_u}) = \sum_{n_u} \pi_w(n_u) \frac{R_w^{b,s_u} G_u(n_u)}{n_u}
\]

- Average Bit transmission delay

\[
\mathbb{E}\left( \frac{1}{R_w^{b,s_u}} \right) = \sum_{n_u} \pi_w(n_u) \frac{n_u}{R_w^{b,s_u} G_u(n_u)}
\]

- Average sojourn time: Using Little’s law, we get the expected time in the system as

\[
\mathbb{E}[S_w^{b,u}] = \frac{\mathbb{E}[n_u]}{\theta_u \rho_w^{b,u}}
\]

C. Share dimensioning for the orchestrator

In the following, we describe how the orchestrator determines the value \( \alpha(w) \) at each regime \( w \). We recall that all performance evaluation metrics are obtained at regime \( w \). Below, we investigate how to dimension network sharing to support slice load subject to SLA of slice \( s_v \) and \( s_u \).

At regime \( w \), the orchestrator needs to satisfy a target requirement of slice \( s_u \) in terms of the QoS metrics. Without loss of generality, we consider the average bit transmission delay as a target requirement that the orchestrator should satisfy [6], i.e., \( \mathbb{E}\left( \frac{1}{R_w^{b,s_u}} \right) \leq d_u \). From the above equation, the orchestrator computes the minimum virtual resource blocks \( C_u(w) \cup C_v \) needed to achieve the average throughput \( R_w^{b,s_u} \) such that:

\[
\sum_{n_u} \eta \frac{n_u}{G_u(n_u)} \left( \frac{\rho_w^{b,s_u}}{G_u(n)} \right)^{n_u} \left\{ \prod_{j=1}^{n_u} G_u(j) \right\}^{-1} \frac{1}{R_w^{b,s_u}} n_u+1 = d_u
\]

Note that \( C_u(w) \) is the fraction \( \alpha(w) \) of pre-assigned virtual resource blocks of slice \( s_u \) that shifted to slice \( s_v \) by the orchestrator.

More analyses are provided in the full version [15], in particular, the analysis based on singular perturbation techniques to provide an analytical expression for all QoS metrics for both slices under a dynamic regime.

VI. SIMULATION RESULTS

In this section, we analyze the behavior of our architecture and evaluate its performance under different regime states. Mainly we compare the numerical results with the results obtained by the framework developed using MATLAB. Extensive simulations have been conducted to validate our analysis.

A. Simulation Setup

We have deployed a 5G network, implementing our slicing scheme with partial isolation, using MATLAB 5G toolbox. We consider a scenario where eNodeBs are controlled by an orchestrator and focus on downlink traffic. Due to users’ mobility, the number of users arriving/departing at/from the network varies over time. The two following slices were created: \( s_v \) offering a video streaming service and \( s_u \) running applications that are sensitive to delay or throughput.

Regarding the adaptive traffic, DASH video streaming was implemented on top of the HTTP protocol along with the adaptation algorithm at the clients’ side. The duration of DASH videos is exponentially distributed with mean 50 seconds at multiple bit-rate versions \( L = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\} \) Mbps. The adaptation algorithm tracks the playback buffer status of the client and requests segments at the suitable bit-rate. As for users in slice \( s_u \), they are assumed to download a fixed file size following an exponential distribution with mean 14 Mbits.

The simulations are done for a bandwidth of 5MHz (25 RBs) in a single antenna FDD mode. The number of eNodeB RRBs is 12, virtually distributed among the two slices as follows: in slice \( s_u \), users share 12 fixed RBs, whereas adaptive users share the rest. As abovementioned, the RRBs dedicated to slice \( s_v \) can be opportunistically assigned to slice \( s_u \), if necessary, by the orchestrator. Signal Interference to Noise Ratio (SINR) is composed based on the physical layer, which includes path loss, shadowing, fast fading, and antenna gain.

The system operates in a discrete-time fashion with a slot duration of 0.5 ms. We simulate a total of 600 users. Users of slice \( s_u \) enter the network with arrival rate \( \lambda^v = 0.2 \). In order to investigate the impact of the traffic intensity in the network, we consider three values of the arrival rate \( \lambda^v \in \{0.22, 0.3, 0.35\} \) flows/s corresponding to regime \( w = \{1, 2, 3\} \) respectively. In Figs. 2 and 5, we compare the simulations’ results to the analytical ones. The simulations’ results have a variance about the mean which is represented by an error bar.

B. Experimental results

We shall first evaluate the performance of our architecture in terms of the QoS of slice \( s_u \) under different regimes. Next, we investigate the impact on slice \( s_v \): we show how it stabilizes the video quality under users’ dynamics and compares it to the case where there is no target rate dictated upon adaptive users. Finally, we exhibit the added-value of our architecture on \( s_u \) QoE under the considered regimes.

1The set of bit-rate chosen were kindly provided by YouTube.
1) Performance regarding non-adaptive traffic:

We assume that the bit transmission delay (BTD) agreed upon it in the SLA of slice $s_u$, is $6 \times 10^{-7}$ s/bit. Under the regime $w = 1$ characterized by $\lambda^1 = 0.22$ flows/s and under perfect isolation between the two slices, this value can still be respected (Fig. 2a). Now, when the traffic intensity increases, i.e., $\lambda^2 = 0.3$ flows/s or $\lambda^3 = 0.35$ flows/s, the SLA of the slice $s_u$ is violated since the value of BTD increases to the point of exceeding $9 \times 10^{-7}$ s/bit.

Based on our mathematical analysis, the orchestrator determines the portion of resources $\alpha(w)$, that should be shifted to slice $s_u$ in order to bring the BTD to a value less than or equal to the one in the SLA requirements. In particular, it determines $M_u$ corresponding to $\alpha(w)$ and apply it to reduce the target rates of the adaptive users, which consequently reduces the number of vRRBs allocated to $s_v$ and then re-allocate them to $s_u$.

In Fig. 2a, it can be observed that the vRRBs $C_v(w)$ freed up by slice $s_u$ made it possible to reduce the BTD. To go further with the analysis, we evaluate the average sojourn time and the average throughput. In Figs. 2b and 2c, we plot the scenario of perfect isolation and the one of the sharing policy. It can be seen that the average throughput has significantly increased since the set of vRRBs used by the slice $s_u$ has been expanded. The sojourn time has been drastically reduced since users in slice $s_u$ are getting relatively more air-time compared to the case of perfect isolation.

2) Performance regarding adaptive traffic:

Fig. 3 shows the impact of throttling the average throughput of the user to the target rate - enforced by the orchestrator and belonging to the set $L$ - on limiting video quality switches under users’ dynamics. The available video bit-rates are represented by the horizontal gray dotted lines, whereas the blue line indicates the average throughput of the user which closely tracks the target rate (purple line). The video quality (depicted by the red line) follows as well and quickly stabilizes to the next video bit-rate after arrivals/departures events. Conversely, frequent fluctuations of the video bit-rate can be observed when there is no target rate imposed by the orchestrator (see Figs 4). In fact, the greedy DASH adaptation algorithm at the client’s side tends to request segments at video bit-rates that the average throughput cannot constantly sustain, resulting in fluctuations of the buffer level.

Let us now evaluate the effect of resource reallocation on adaptive flows under the regimes $w = 2$ and $w = 3$ and benchmark it with the regime $w = 1$ which represents a low traffic intensity and where the slice $s_u$ only uses its own RRBs. Here, we assume that the minimum video bit-rate that an adaptive user can experience is 0.3 Mbps.

In Fig. 5a, it is observed that the probability of users encountering a stall has been drastically reduced under the sharing policy compared to the case of perfect isolation. This is due to the fact that the video segments are downloaded with lower video bit-rates which means that the probability of the playout buffer being empty decreases. Fig. 5c shows that the average number of video bit-rate switches decreased for the sharing policy. This is because adaptive flows release some of their vRRBs which means that their target rates take value in a fixed subset of $L$ and so their video quality as well since this latter follows the target (see Fig. 3). We further observe a drop in video quality for the sharing policy in Fig. 5b. This is expected since, under sharing policy, adaptive flows release a fraction of their resource blocks to be used by non-adaptive flows in order to respect the SLA of slice $s_u$.

VII. CONCLUSION

The paper has addressed a radio access network slicing mechanism that provides different levels of resource isolation through resource abstraction, virtualization, and the ability to efficiently share network resources. We have explored the idea of a software-defined RAN where the RAN slices are specifically tailored to accommodate both QoS and QoE requirements for 5G services. We have proposed a two-level scheduling process including intra-slice traffic scheduler and inter-slice scheduler to ensure fairness among users belonging to the same slice and flexible resource sharing between slices while adapting to network changes (traffic load, SLAs constraints, etc.). The results obtained have shown that the proposed approach allows us to design an efficient and low complexity RAN slicing by exploiting the characteristic of adaptive traffic. Our mathematical analysis and simulation results have confirmed the benefits of resource abstraction. We have further determined in which cases the gains in efficiency can be obtained by exploiting the flexibility of traffic streaming in terms of QoE.

REFERENCES

Fig. 2: Performance results of slice $s_u$ under different regimes: (a) Bit transmission delay, (b) Sojourn time and (c) Average throughput.

Fig. 5: Performance results regarding slice $s_v$ under different regimes: (a) Probability of stalls (b) Average video bit-rate (c) Average number of video bit-rate switches.


