Enabling Dynamic Resource Sharing for Slice Customization in 5G Networks

Alessandro Lieto*†, Ilaria Malanchini†, Antonio Capone*
*Dipartimento di Elettronica, Informazione e Bioingegneria Politecnico di Milano, Italy
Email: alessandro.lieto@polimi.it, antonio.capone@polimi.it
†Nokia Bell Labs, Stuttgart, Germany
Email: ilaria.malanchini@nokia-bell-labs.com

Abstract—**Network slicing is one of the main novelty of 5G systems and it is expected to radically change the approach of network operators to the support of vertical applications with specific and stringent performance requirements that can be managed by tenants. Static allocation of resources to slices is a straightforward approach to meet Service Level Agreements (SLAs), but it may lead to dramatic inefficiencies and high costs. On the other side, dynamic sharing can greatly improve resource utilization, but it requires a strict control of performance parameters so as to guarantee small congestion probability and smoothed degradation when it occurs. In this paper, we propose a utility-based approach that can properly map the slice customization strategies that tenants can use to address the different traffic requirements and their specific service management policies. We define resource scheduling mechanisms that allow differentiation among slices and service prioritization through slice-specific parameters. We model resource allocation as a mathematical programming problem and present numerical results that show the viability of the proposed approach.**

I. Introduction

In the next years, it is expected that mobile internet traffic not only will continue to increase fast, but it will also diversify with an unprecedented proliferation of connected devices that will support a wide range of new applications [1]. This growing trend towards an *everything*-and-*everywhere* connected society is exposing network operators to a substantial increase of their operating expenditure (OpEx) and capital expenditure (CapEx), which may not be easily matched by an increase in their profit [2].

For network operators it is therefore mandatory exploiting the new use cases enabled by 5G and reshaping their *modus operandi* in business. The NGMN Alliance, that includes most operators, has identified the concept of *Network Slicing* [3] as the one that will play a crucial role in new 5G networks and will influence the renovation of the business ecosystem. Network slicing allows to accommodate on the same physical infrastructure different end-to-end logical networks, referred to as *network slices*. Slices manage sets of virtualized communication resources, network elements and functions, exposed through abstractions by a shared infrastructure for supporting different services and isolating their traffic.

By enabling programmability, flexibility, and scalability also in the wireless domain, virtualization technologies, such as Software Defined Networks (SDN) and Network Function Virtualization (NFV), make network slicing a viable solution for the next generation networks. The logical partition of the network opens up opportunities for new roles in the mobile ecosystem: in [4] it is envisioned the new role of "network slice broker" governing the interactions among an infrastructure provider, responsible for the network deployment, and tenants, that run their services on top of it. Similarly, in [5] it is defined the concept of *Network Slicing as a Service*, NSaaS, that enables enterprises focused on specific application domains (also referred to as *verticals*) to deploy their own logical network and to deliver tailored services to their customers.

Although virtualization has been widely used in cloud systems and wired networks, several challenges are still open when applying it to the wireless domain, mainly due to radio resource abstraction and isolation, mobility management, and security compliance [6]. In particular, when considering the radio resource allocation problem, standardization organizations, e.g. 3GPP, emphasize the need of slice isolation [7]. This means that each slice is expected to be logically independent, and that quality guarantees included in its Service Level Agreement (SLA) should not be affected by the behavior of the other slices.

As a results, most of the related literature on network slicing [8]–[11], suggests to translate SLAs into a fixed share of the physical available resources to be statically assigned to each slice. Authors in [8] analyze admission control algorithms, implemented at the network side, that allow an infrastructure provider to maximize its revenue accepting slices only if the SLAs can be honored. Along the same lines, in [9] a prediction algorithm is proposed to predict the mobile traffic of each slice and apply different admission control policies, considering the tradeoff between accepting more slices and increasing the probability of SLAs violation.

Other previous work points out the need of schemes dealing with user demand uncertainties. In [10], a network slice allocation problem is solved by robust optimization, distributing resource shares so as to minimize the extra costs due to demand uncertainties. A fair resource allocation is considered in [11], where a distributed algorithm allocates resources to users of each slice, based on traffic demands on different locations. This allows for flexible assignment of resource shares on different locations, while guaranteeing SLAs on the overall network.

However, it is rather obvious that a static allocation of resources per slice does not necessarily provide service guarantees nor resource efficient use, due to the stochastic nature of the wireless channel and traffic fluctuations across the network. Actually, the advantages of resource pooling and spectrum sharing have been already largely demonstrated [12], [13]. We, therefore, envision an allocation mechanism that exploits the advantages of resource pooling through a dynamic sharing among slices, and, at the same time, is able to guarantee slice-specific requirements, including those for critical applications. Dynamic sharing among slices has been also considered in [14], which however considers as quality parameter only the throughput achieved by the users. We propose here a more general approach based on utility functions that include also latency, the most critical parameter for several 5G application scenarios.

A. Our contribution

We propose a dynamic network slicing framework, where multiple slices share radio resources managed by a single network operator. We assume that the operator has the control on the overall network and orchestrates its capabilities by real-time network reconfiguration and flexible reassignment of the resources. It does so considering the current slice traffic and requirements, while maximizing the overall network efficiency.

We consider different metrics, namely latency and throughput, and evaluate the achieved performance by using slice-specific utility functions. In this way, we allow slice *customization*, by translating the service requirements (and achieved quality) of the specific slice in a measurable quantity that can be monitored in real-time by the tenants. To the best of our knowledge, this is one of the first works that considers both throughput and latency in utility definition and resource allocation, and allows slices to differentiate their requirements based on these parameters.

In particular, we define *differentiation* among slices and service prioritization, by introducing slice-specific parameters. These allow tenants to require different utility guarantees and to enforce specific service provisioning strategies, tuning the contribution of the considered metrics in their utility. In this paper we assume that these parameters are given and known to tenants and network operator. In a more general context, they can be directly mapped in SLAs, and dynamically renegotiated based on the tenants' needs and network conditions.

The paper is organized as follows: in Section II, we present the network slicing problem and describe all the parameters involved in the definition of the slice customization. In Section III, we present the approach and the algorithm that allocates the resource and the impact of the slice-specific parameters. Then, in Section IV, numerical results are discussed to assess the validity of the proposed solution and finally, our remarks conclude the paper in Section V.

II. System Model

We consider a scenario where a single Mobile Network Operator (MNO) manages the downlink of a base station scheduler, whose wireless physical resources are shared among different slices. We denote by S the set of instantiated slices (and tenants[1]); let K be the set of users in the system, and denote by K_s the subset of active users of slice s. Each tenant $s \in S$ defines its slice requirements as a set of network capabilities (network functions, coverage area, expected user density, quality requirements, etc.) that, in addition to the instantiation of end-to-end slice services, are translated at the base station scheduler in Key Performance Indicators (KPIs), such as latency and throughput requirements.

We model radio resource sharing as a queuing system that works in a slotted time $n \in N$, where packets arrive randomly at the scheduler of a base station at the start of each time slot n. The packet arrival process follows a Poisson distribution, with λ_s being the average arrival rate for packets of slice s. Then, we denote by b_s the packet length of slice s, normalized by the size of a Physical Resource Block (PRB), i.e., the system bandwidth multiplied by the duration of one time slot. Let $D_k[n]$ be the cumulative number of packets arrived in the system at time n for user k. Similarly, we denote by $z_k[n]$ the cumulative number of packets served to user k at time slot n. We assume that a packet is successfully received when the total number of transmitted bits is equal to the packet size[2].

The base station scheduler hosts one buffer per slice of infinite length. We denote the state of the s-th buffer by $Q_s[n] \in \{0, 1\}$, indicating whether at time n the buffer is empty or not. We also assume that, in each buffer, packets are served according to the First-In-First-Out (FIFO) policy: the rational behind this assumption is that each slice defines a unique service type and, then, all packets are treated equally within one slice. On the contrary, packets belonging to different buffers are served based on a scheduling policy that guarantees slice customization and differentiation. To implement this, we define a set of network performance metrics denoted by $\mathbb{O}$, e.g., average user throughput, minimum latency. For each element $o \in \mathbb{O}$, we define a generic utility function as:

$$U_s^o\left(f^o(x_s), \beta_s^o\right), \quad \forall o \in \mathbb{O}, \forall s \in S, \tag{1}$$

where β_s^o is the requirement of slice s for the specific metric o and $f^o(x_s)$ is a function of the allocated resources to slice s aggregated over all its users:

$$x_s = \sum_{n \in N} \sum_{k \in K_s} x_k[n] \tag{2}$$

and $x_k[n]$ is the fraction of assigned resources by the scheduler to the user k at time slot n. Finally, we assume perfect channel knowledge at the scheduler and denote by $r_k[n]$, the maximum achievable rate of user k at time slot n.

[1]In this work, we assume that a tenant runs a single slice, and therefore, hereinafter, we will refer interchangeably either to tenant or slice. However, the proposed framework can be easily extended to the case of multiple slices per tenant.

[2]Note that we refer to application layer packets (messages): this implies that at each n it is possible that only a portion of a packet is transmitted, e.g., after Service Data Unit (SDU) segmentation at Radio Link Control (RLC) layer.

A. Utility function definition

We assume that each slice defines a specific service, for which the tenant requires tailored and customized network configurations. We model the slice customization at the radio scheduler through piece-wise linear utility functions, that are able to map the achieved network performances with the requirements set by the tenants. The linearity assumption is mainly made for mathematical tractability. Alternative utility functions can be considered as well, without affecting the proposed formulation.

In this work, we focus on two service performance metrics, *latency* and *throughput*, and provide a utility characterization for each of them.

1) Latency utility function: For each packet in the buffer, we measure the waiting time before it is transmitted, and define this time as latency. Then, we define (i) the *maximum experienced latency* as:

$$L_s^{\text{max}} = \max_{k \in K_s} \{(n-i), \forall i \leq n : z_k[n] = D_k[i])\}, \ \ \forall s \in S, \quad (3)$$

and (ii) the *average experienced latency* as:

$$L_s^{\text{ave}} = \frac{1}{|K_s|} \sum_{k \in K_s} \frac{1}{D_k} \sum_{d \in D_k} l_d, \ \ \forall s \in S \quad (4)$$

where l_d denotes the latency of the single packet d and D_k the total number of packets arrived in the system for user k. The former describes the maximum packet latency experienced by any user of slice s; the latter evaluates the average latency experienced by all the users of that slice.

For each of these two measurements, we define a utility function denoted by:

$$\hat{U}_s^{\text{L}}(y, \tau_s) = \begin{cases} U_{\text{tar}}^L, & \text{if } y \leq \tau_{\text{tar}}, \\ U_{\text{tar}}^L - \frac{(U_{\text{tar}}^L - U_{\text{min}}^L)(y - \tau_{\text{tar}})}{(\tau_{\text{max}} - \tau_{\text{tar}})}, & \text{if } \tau_{\text{tar}} \leq y \leq \tau_{\text{max}}, \\ U_{\text{min}}^L, & \text{if } y \geq \tau_{\text{max}}, \end{cases} \quad (5)$$

where y is the variable denoting the latency performance metrics, i.e., either L_s^{max} or L_s^{ave}, and τ_s is the latency requirement of the slice. Namely, we assume that each slice specifies an interval of required latency, defined by τ_{tar}, the target latency, and τ_{max}, the maximum tolerable latency, and $\tau_s = \{\tau_{\text{tar}}, \tau_{\text{max}}\}$. An illustrative example is shown in Fig. 1(a): for y values below τ_{tar}, utility is equal to its maximum value, set to U_{tar}^L, while for values above τ_{max}, it drops to U_{min}^L. Between these two values, we assume that the utility decreases linearly w.r.t. the latency. The slope of the curve describes the tolerance of the slice to latency. Note that by setting properly the latency parameters, slice can obtain either a step function (infinite slope when $\tau_{\text{tar}} = \tau_{\text{max}}$) or a flat function (zero slope when $\tau_{\text{max}} = \infty$).

Based on the definitions above, the overall *latency utility* can be expressed as:

$$U_s^L = \delta_s \cdot \hat{U}_s^L \left(L_s^{\text{max}}, \tau_s\right) + (1 - \delta_s) \cdot \hat{U}_s^L \left(L_s^{\text{ave}}, \tau_s\right), \ \ \forall s \in S, \quad (6)$$

where δ_s is a weighting factor set by the tenant, that defines the slice priority of the latency utility, in terms of maximum latency, L_s^{max}, or average latency, L_s^{ave}.

2) Throughput utility function: We define the aggregate user throughput of one slice as:

$$R_s = \frac{1}{T_s^{\text{ACT}}} \sum_{k \in K_s} z_k[N] \cdot b_k, \ \ \forall s \in S, \quad (7)$$

where we denote by $z_k[N]$ the total number of packets transmitted to user k and by T_s^{ACT} the overall time during which the buffer is active to transmit packets. We represent the *throughput utility* function as:

$$U_s^{\text{T}}(y, \rho_s) = \begin{cases} U_{\text{tar}}^T, & \text{if } y \geq \rho_{\text{tar}}, \\ U_{\text{tar}}^T - \frac{(U_{\text{tar}}^T - U_{\text{min}}^T)(y - \rho_{\text{tar}})}{(\rho_{\text{min}} - \rho_{\text{tar}})}, & \text{if } \rho_{\text{tar}} \geq y \geq \rho_{\text{min}}, \\ U_{\text{min}}^T \frac{(y - \rho_{\text{zero}})}{(\rho_{\text{min}} - \rho_{\text{zero}})}, & \text{if } \rho_{\text{min}} \geq y \geq \rho_{\text{zero}}, \\ 0, & \text{if } y \leq \rho_{\text{zero}}, \end{cases} \quad (8)$$

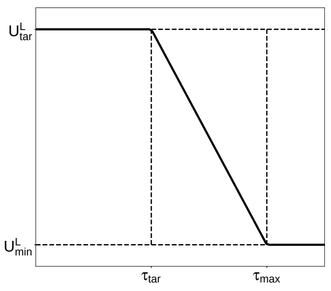
where $y = R_s$ is the aggregate user throughput of slice s and $\rho_s = \{\rho_{\text{tar}}, \rho_{\text{min}}, \rho_{\text{zero}}\}$ is the corresponding throughput requirement. While for latency we consider a single linearity region, for the throughput we propose a piece-wise linear function, as in [14]. As shown in Fig. 1(b), we assume that for each slice a basic bit-rate, ρ_{zero}, has to be provided. To stress this request, the utility value is set to 0 for all achieved bit-rates below the threshold. Then, we identify two linear regions with different slopes. In the first region, the tenants set a minimum guaranteed bit-rate, ρ_{min} necessary to deliver standard quality services, for which a corresponding utility U_{min}^T is assigned. In the second region, we model the increase in throughput necessary to deliver high quality services to the slice, so that when the achieved bit rate is greater or equal than ρ_{tar}, the utility saturates to its maximum value, U_{tar}^T. Being a measure of the aggregate throughput of the users, the slopes of the two linear regions will also vary in time according to the traffic load variations of the slice.

B. Slice type characterization

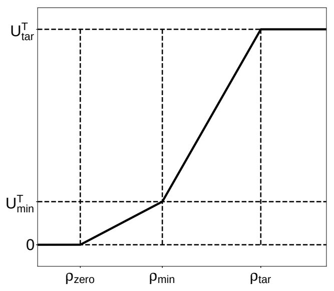
By considering both latency and throughput in the utility definition, we are now able to cope with a big plurality of services, varying from very low latency to very high throughput applications, hence exploiting the potential of network slicing.

In particular, we consider some of the well-know categories of services envisioned for 5G: Tactile Internet (TI), enhanced Mobile BroadBand (eMBB), massive Machine Type Communication (mMTC) and critical Machine Type Communication (cMTC) [15].

1) Latency utility: Based on the definition of latency provided in the previous section, the slice latency utility for the four slice types is shown in Fig. 2(a). Among the applications that are included under the big umbrella of Ultra-Reliable Low Latency Communication (URLLC), i.e. latency critical, we select as use cases the cMTC and TI. These two slices represent the most-critical applications in terms of latency, whose requirements are identified with values below 1 ms [15]. Furthermore, to provide service differentiation, we assume that TI can tolerate slightly higher latency values, given that for these applications high throughput is required as well [16].

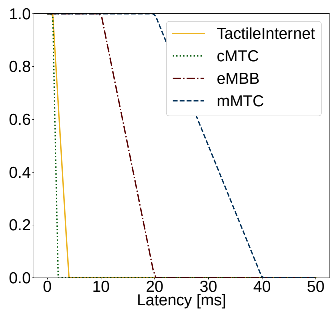


(a) Latency utility function

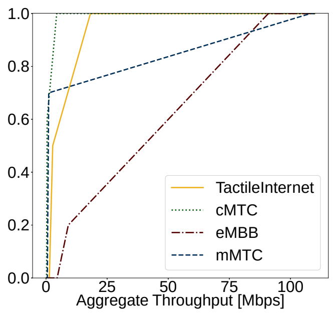


(b) Throughput utility function

Fig. 1: Generic utility function definition.



(a) Latency utility function



(b) Throughput utility function

Fig. 2: Slice utility customization.

Differently, we assume that the eMBB and the mMTC slices are more flexible w.r.t. the latency requirements and their utility is less impacted by delay in the scheduling decisions.

2) Throughput utility: As show in Fig. 2(b), we assume that for the low-demanding applications, in terms of throughput, the achievement of the minimum guaranteed bit-rate is sufficient to provide a fruitful service, resulting in a relatively high utility. In contrast, for the eMBB slice, we impose a much lower utility value, which indicates a low provided quality. For the second region, the critical applications (cMTC and TI) show almost a step behavior, which is a direct effect of the stringent latency requirements of such applications: an increase in the throughput leads to a decrease in the latency. On the other side, the non latency-critical applications are more demanding in terms of aggregate throughput: in particular, for the mMTC we assume that a minimum guaranteed bit rate must be provided, but given the massive number of devices expected, the aggregate throughput can be high. Similarly, for the eMBB slice we assume a per-user high throughput requirement, but with a relatively low number of users simultaneously active per cell.

The specific values used in the numerical evaluation for both the latency and the throughput utility are detailed in Section IV.

III. Problem Formulation

Based on the utility functions defined in the previous section, the scheduler assigns the physical resources to users in order to fulfill the slice requirements, respectively translated into utility functions (and values). Furthermore, as mentioned in the introduction, we introduce slice-specific parameters, $\underline{\alpha}_s$, to enable a dynamic mechanism in which the tenants can decide how to weight their utility components. In the following, we detail the problem formulation and provide insights on the effects of the slice parameters $\underline{\alpha}_s$.

A. *Slice-aware resource allocation problem*

We define the resource allocation problem by means of the optimization formulation described by Eq. (9a)-(9f):

$$\max \sum_{s \in S} \underline{U}_s \cdot \underline{\alpha}_s \tag{9a}$$

$$\text{s.t.} \sum_{k \in K} x_k[n] \leq 1, \quad \forall n \in N, \tag{9b}$$

$$z_k[n] \leq D_k[n-1], \quad \forall k \in K, \ \forall n \in N, \tag{9c}$$

$$\sum_{i=1}^{n} x_k[i] \cdot r_k[i] \leq D_k[n] \cdot b_k, \quad \forall k \in K, \ \forall n \in N, \tag{9d}$$

$$\sum_{i=1}^{n} x_k[i] \cdot r_k[i] \geq z_k[n] \cdot b_k, \quad \forall k \in K, \ \forall n \in N, \tag{9e}$$

$$Q_s[n] = \begin{cases} 0, & \text{if } D_k[n] = z_k[n] \\ 1, & \text{otherwise} \end{cases}, \ \forall n \in N, \forall k \in K_s, \forall s \in S, \tag{9f}$$

where the optimal solution is obtained maximizing the weighted sum of the slices utilities. Without loss of generality, we assume in this work $\underline{U}_s = \{U^L_s, U^T_s\}$ and $\underline{\alpha}_s = \{\alpha^L_s, \alpha^T_s\}$, which can be however extended to include several utility functions and corresponding weights.

Constraint (9b) ensures that at each time slot the scheduler does not assign more resources than the ones available in the network. With constraint (9c), we assume that a packet can be considered successfully transmitted only at the end of a time slot, or equivalently at the beginning of the next one. In (9d), we ensure that the number of transmitted bits cannot be greater than the total number of bits arrived in the system at that time slot. Based on (9c) and (9d), constraint (9e) updates the cumulative number of received packets, at each time slot n, based on the total number of bits received. Finally, equation (9f) defines the state evolution of the buffers[3].

B. *Customization and differentiation of slices*

The proposed formulation always guarantees the maximization of the utilities for all the slices. Therefore, when there is "enough resources for everyone", we can assume that all slices perfectly reach their maximum utility. However, the MNO has to be ready also to deal with situations in which resources are not enough, e.g. due to congestion. For this reason, we assume that tenants are able to monitor real-time their slice performances and can modify their priority metrics in order to rescale their utilities and get different network performances.

[3]For the sake of readability, we write the constraint as non-linear, but it can be easily linearized with standard techniques.

	TI	cMTC	eMBB	mMTC
$[\tau_{\text{tar}} - \tau_{\text{max}}]$ [ms]	$[1-4]$	$[1-2]$	$[10-20]$	$[20-40]$
U^L_{tar}	1	1	1	1
U^L_{min}	0	0	0	0
δ_s	0.1	0.1	0.1	0.1
$[\rho_{\text{min}} - \rho_{\text{tar}}]$ [Mbps]	$[3-20]$	$[0.1-1]$	$[5-50]$	$[0.01-1]$
U^T_{tar}	1	1	1	1
U^T_{min}	0.5	0.6	0.7	0.2
λ_s [pkt/ms]	1	4	0.4	6
b_s [kbit]	20	1	250	20
$\lvert K_s\rvert$	1	4	2	110

TABLE I: Simulation parameters.

By introducing the slice-specific parameters $\underline{\alpha}_s$, we enable tenants to achieve slice customization and differentiation. These features become particularly interesting in case of congestion of the network, since the MNO has to take decisions how to handle slices, when it is known a priori that meeting all requirements might not be possible. In this sense, tuning $\underline{\alpha}_s$ allows the tenants:

- To define priorities among the set of network performances (that has to be satisfied) within a single slice (*slice customization*). According to this, whenever the MNO is not be able to meet all the slice requirements, the resources will be assigned so as to maximize the utilities of the metrics with higher priority.
- To define priorities among slices (*slice differentiation*). In this case, when the MNO is not able to provide the maximum utility values for all the slices, these parameters give indication on the most critical slices that require higher priority.

IV. Numerical Results

A. Simulation setup

The simulation scenario adopted for our numerical analysis is based on the numerology 0 of the 5G frame structure [17]. We consider $M = 60$ sub-windows, each one of length equal to $N = 100$ time slots, where the scheduler takes decision on every time slot of length 1 ms. Namely, at every sub-window[4], we solve the optimization problem defined in Eq. (9a)-(9f), by means of the Gurobi Mixed Integer Linear Programming (MILP) solver [18]. Numerical results are evaluated at the end of the simulation as average values over all the sub-windows.

We consider $|S| = 4$ slices, one per type defined in Section II. The simulation parameters for each slice have been derived according to [15] and are reported in Table I. We also assume for all slices $\rho_{\text{zero}} = \frac{\rho_{\text{min}}}{2}$. The λ_s parameter defines the aggregated message arrival rate of users in slice s. We assume that both mMTC and cMTC have high rates but small size messages. While for eMBB, we consider relatively large messages, to map the high bandwidth requirements of this slice, and for TI, both low rate and small messages.

[4]Note that we do not solve the problem at every time slot, but evaluate the optimal allocation policy that would satisfy the tenants' requirements at every sub-window, assuming knowledge of upcoming users' channel states and packet arrivalss.

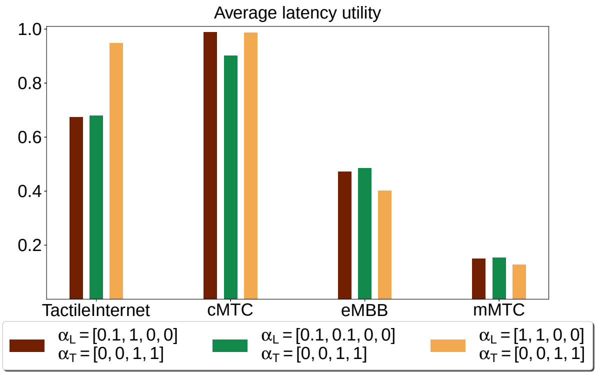


Fig. 3: Average latency utilities when varying the slice-specific latency parameters.

Users are uniformly distributed throughout the coverage area of the base station (with a radius of 250 m), and during the simulation period they are active and do not move. We model the wireless channel by a frequency-flat block fading channel with i.i.d. Rayleigh coefficients, resulting in exponentially distributed random channel gains $|h_k[n]|^2$. Then, we assume that the power received by a generic user is calculated through the Okumura-Hata model, indicating the base station transmitted power (in Watts [W]) by P, and the path-loss exponent by $\beta = 3.5$. Hence, the average Signal-to-Interference-plus-Noise-Ratio (SINR) of user k, SINR$_k$, can be computed as:

$$\text{SINR}_k = \frac{P \cdot d_k^{-\beta}}{\sigma^2 + I_0}, \tag{10}$$

where d_k is the user's distance from the base station (in meters [m]), σ^2 is the thermal noise, I_0 is the average interference power of the neighboring base stations, with $\frac{P}{\sigma^2+I_0}$ = 90 dB. Due to the fast fading components, the instantaneous SINR at time slot n is equal to

$$\gamma_k[n] = |h_k[n]|^2 \cdot \text{SINR}_k, \tag{11}$$

and then the reference spectral efficiency of a user k (in bit/s/Hz) at any time instant n is

$$r_k[n] = \log_2(1 + \gamma_k[n]). \tag{12}$$

B. Numerical evaluation

The scheduling behavior depends mainly on two factors: the utility functions and how they are defined for the different slice types, and how the tenants weight them together by setting $\underline{\alpha}_s$. In presenting results, the slice-specific weights are grouped and shown as follow

$$\alpha_T = [\alpha^T_{TI}, \alpha^T_{cMTC}, \alpha^T_{eMBB}, \alpha^T_{mMTC}],$$

for the throughput parameters, whereas the slice-specific latency parameters as

$$\alpha_L = [\alpha^L_{TI}, \alpha^L_{cMTC}, \alpha^L_{eMBB}, \alpha^L_{mMTC}].$$

Slice parameters	% of resources
$\alpha_L = [0.1, 0.1, 0, 0]$ $\alpha_T = [0, 0, 1, 1]$	$[15.2, 4.9, 64.0, 15.7]$
$\alpha_L = [0.1, 1, 0, 0]$ $\alpha_T = [0, 0, 1, 1]$	$[15.2, 5.6, 63.6, 15.4]$
$\alpha_L = [1, 1, 0, 0]$ $\alpha_T = [0, 0, 1, 1]$	$[17.6, 5.6, 62.0, 14.6]$

TABLE II: Percentage of resources allocated to each slice, for different slice-specific latency parameters, according to the tuple [TI, cMTC, eMBB, cMTC].

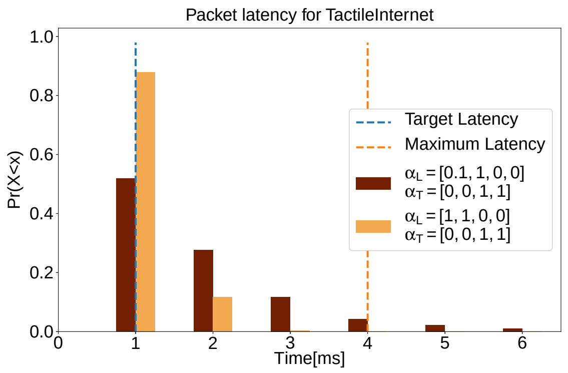


Fig. 4: Cumulative distribution function of packet latency for Tactile Internet slice.

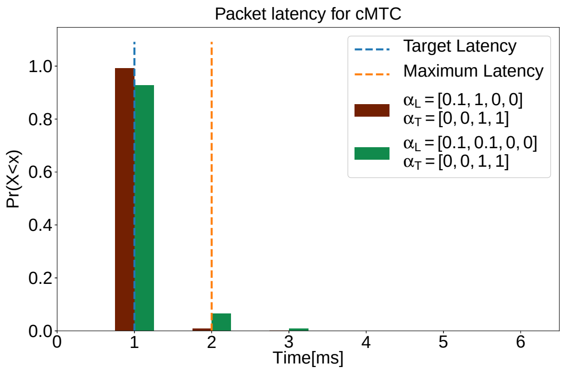


Fig. 5: Cumulative distribution function of packet latency for cMTC slice.

Slice parameters	% of resources
$\alpha_L = [1, 1, 0, 0]$ $\alpha_T = [0, 0, 1, 0.1]$	$[17.7, 5.6, 65.1, 11.3]$
$\alpha_L = [1, 1, 0, 0]$ $\alpha_T = [0, 0, 0.1, 1]$	$[18.2, 5.7, 12.3, 63.6]$
$\alpha_L = [1, 1, 0, 0]$ $\alpha_T = [0, 0, 1, 1]$	$[17.6, 5.6, 62.0, 14.6]$

TABLE III: Percentage of resources allocated to each slice, for different slice-specific throughput parameters, according to the tuple [TI, cMTC, eMBB, cMTC].

In Fig. 3, we consider three different scenarios, where we fix the α_T values while modifying the α_L values for the latency-critical slices. As expected, when tenants decrease the value of α_s^L from 1 to 0.1, their utility values also decrease. According to the definition of average latency (Eq. (4)) and the corresponding slice utility (Fig. 2(a)), the utility values decrease when one slice experiences an increasing delay in the transmission of the packets in its buffer. This means that, in order to maximize the utility function of latency-critical slices, users need to be scheduled as soon as a packet arrives in the buffer, regardless of their channel conditions. This approach assigns the maximum priority to such users and does not allow the scheduler to adopt more efficient scheduling decisions. This is shown in Table II, where we can see that the higher the values of α_s^L, the higher the percentage of allocated resources. On the other side, for lower values of α_s^L, the critical services (cMTC and TI) get less resources and observe a degradation in latency performance, which is balanced by an increase in resource utilization for the other slices (mMTC and eMBB). This degradation is reported for the latency-critical slices in Fig. 4 and Fig. 5. Between the two, the TI slice experiences higher packet delay because it requires more bandwidth and then competes on a larger amount of resources with the other slices.

In Fig. 6, we consider the scenario where we keep fixed α_s^L values, and change the values of α_s^T for eMBB and mMTC. Since the critical slices have their α_s^L parameter set to 1, the scheduler always assigns maximum priority to the latency-critical slices, and this indirectly maximizes their throughput utilities (equal to 1). After having satisfied the requirements for the latency-critical slices, the scheduler optimizes the allocation of the available resources, privileging the slice for which the utility has the steeper slope. As a matter of fact, we can observe that the scheduler is always able to guarantee the minimum bit-rate to all slices, namely providing the minimum slice utility value.

Moreover, as shown in Fig. 7, the performance of the eMBB slice does not experience significant degradation due to changes in the α_s^T of the mMTC (compare red and green curves). Due to steeper slope in the second linearity region (Fig. 2(b)), the eMBB users get higher priority w.r.t. to the mMTC ones, once the minimum bit rate has been guaranteed. Therefore, only when the mMTC slice requires higher priority w.r.t. the eMBB slice (yellow curve in Fig. 7 and Fig. 8), the mMTC users can achieve higher throughput.

Furthermore, it is worth noting that, differently from latency-critical services, for which the channel conditions of the users do not affect the scheduling priority, for throughput-oriented slices, the increase in utility can also be caused by good channel conditions of the users (which explains also what observed in Fig. 6, where values can also be above the minimum utility).

Finally, we can observe from Table III that, on one side, the tuning of α_s^T in the utility function allows the competing slices (eMBB and mMTC) to get higher amount of resources (to the detriment of the competitor). On the other side, however, the competition among these slices does not affect the resource distribution for the critical slices (TI and cMTC).

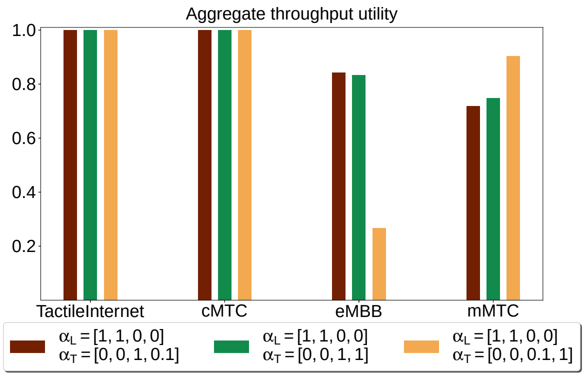


Fig. 6: Aggregate throughput utilities when varying the slice-specific throughput parameters.

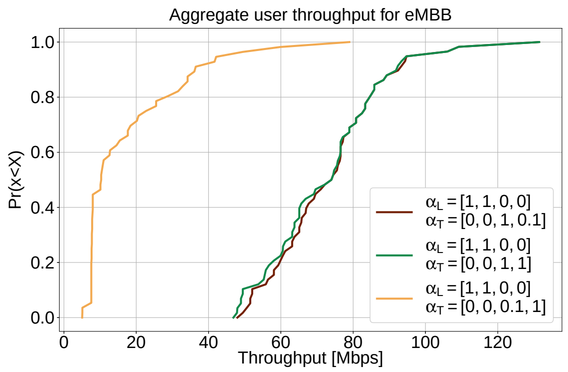


Fig. 7: Cumulative distribution function of aggregate throughput for eMBB slice.

V. Conclusion

In this work, we propose a dynamic resource sharing algorithm among slices. We provide a slice utility characterization, that enables customization of behaviors of different slice types. Furthermore, differentiation among tenants is achieved, by modifying slice-specific parameters that dynamically reshape their slice utility functions. However, numerical results show that performance of latency-critical services are not affected by changes in other slices. Moreover, the algorithm always guarantees the minimum slice utility value, which corresponds to a minimum guaranteed bit-rate for each slice. Finally, the interactions among tenants and MNO on the renegotiation of such parameters has not been investigated in this work, but the technical and business implications of such agreements will be studied in future works.

Acknowledgement

The work has been supported in part by the Bell Labs Internship program and by the European Union Horizon 2020 Marie Sklodowska-Curie ITN/EID project ACT5G (Grant Agreement No. 643002).

References

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021," Feb. 2017. [Online]. Available: http://cs404620.vk.me/u11927290/docs/c94e7ac85fc9/1_McKinsey_Monetizing_mobile_2014-06.pdf
[2] McKinsey and Company Inc., "Monetizing mobile: making data pay," 2014. [Online]. Available: http://cs404620.vk.me/u11927290/docs/c94e7ac85fc9/1_McKinsey_Monetizing_mobile_2014-06.pdf
[3] NGMN, "5G White Paper," 2015. [Online]. Available: http://www.ngmn.de/5gwhite-paper.html
[4] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32–39, July 2016.
[5] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, July 2016.
[6] C. Liang and F. R. Yu, "Wireless network virtualization: A survey, some research issues and challenges," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 358–380, Firstquarter 2015.
[7] X. de Foy and A. Rahman, "Network slicing : 3GPP use case," October 2017. [Online]. Available: https://tools.ietf.org/id/draft-defoy-netslices-3gpp-network-slicing-02.html
[8] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, and X. Costa-Perez, "Optimising 5G infrastructure markets: The business of network slicing," in *IEEE Conference on Computer Communications (INFOCOM)*, May 2017.
[9] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in *IEEE Conference on Computer Communications (INFOCOM)*, May 2017.
[10] A. Baumgartner, T. Bauschert, A. M. C. A. Koster, and V. S. Reddy, "Optimisation models for robust and survivable network slice design: A comparative analysis," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2017.
[11] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 3044–3058, Oct 2017.
[12] I. Malanchini, S. Valentin, and O. Aydin, "Wireless resource sharing for multiple operators: Generalization, fairness, and the value of prediction," *Computer Networks*, vol. 100, pp. 110 – 123, 2016.
[13] A. A. Gebremariam, M. Chowdhury, A. Goldsmith, and F. Granelli, "Resource pooling via dynamic spectrum-level slicing across heterogeneous networks," in *14th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan 2017.
[14] O. U. Akguel, I. Malanchini, V. Suryaprakash, and A. Capone, "Service-aware network slice trading in a shared multi-tenant infrastructure," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2017.
[15] "Air interface framework and specification of system level simulations," in *FANTASTIC 5G Deliverable D2.1*, May 2016.
[16] A. S. Shafigh, S. Glisic, and B. Lorenzo, "Dynamic network slicing for flexible radio access in tactile internet," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2017.
[17] 3GPP TS 38.211, "NR; Physical channels and modulation."
[18] Gurobi, in *Gurobi Solver*. [Online]. Available: http://www.gurobi.com

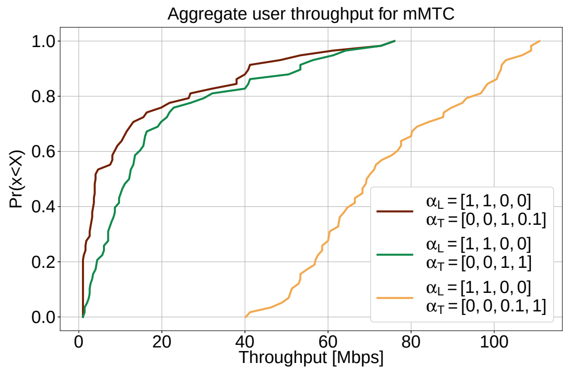


Fig. 8: Cumulative distribution function of aggregate throughput for mMTC slice.