# ONETS: Online Network Slice Broker From Theory to Practice

Vincenzo Sciancalepore®, *Senior Member, IEEE*, Lanfranco Zanzi®, *Member, IEEE*,
Xavier Costa-Pérez®, *Senior Member, IEEE*, and Antonio Capone®, *Fellow, IEEE*

*Abstract*— **Virtualization and network slicing offer an unprecedented opportunity to mobile network operators: open their physical network infrastructure platform to the concurrent deployment of multiple logical self-contained networks, namely** *network slices*. **In this paper, we propose and analyze** *ONETS*, **an Online NETwork Slicing solution that i) builds on the budgeted lock-up multi-armed bandit mathematical model and properties, ii) derives its analytical bounds in our proposed extension for network slicing, iii) seamlessly integrates into the 3GPP architecture, iv) proves its feasibility through a proof-of-concept implementation on commercial hardware considering three network slices and v) allows for the design of a low-complexity online network slice brokering solution that maximizes multiplexing gains.**

*Index Terms*— **5G, network slicing, brokering, virtualization, online algorithm, RAN.**

## I. INTRODUCTION

**W**IRELESS and mobile networks are the key-elements of today's society, enabling communication, access and instantaneous information sharing. However, as cellular networks make a move from being voice-centric to data-centric, operators revenues are not able to keep pace with the upcoming increase of traffic volume resulting in profit reduction. Such pressure on operators return on investment has pushed research efforts towards designing novel mobile network solutions able to open the door for new revenue sources.

The emerging *network slicing* paradigm provides new business opportunities by enabling mobile operators to open their network infrastructure to multiple *tenants*, which may own a single slice with very diverse requirements. This potential

new vertical market multiplies the monetization opportunities of the network infrastructure as $i$) new business players may be involved (e.g., automotive industry, e-health, etc.) bringing in new economic opportunities, and $ii$) a higher infrastructure capacity utilization (i.e., resource efficiency that in turn translates into additional return of investment) can be achieved by admitting as many network slice requests as possible while exploiting stochastic multiplexing gains obtained from such a multi-tenant shared infrastructure. Network slicing unlocks the concurrent running of diverse services: each network slice instance consists of a set of virtual network functions—selected to provide given service requirements—while running on the same infrastructure equipped with a tailored orchestration layer. The novel paradigm comes with new technical challenges that need to be addressed in order to be adopted in practice. When deploying a network-slicing-enabled architecture, the trade-off between a *fully shared* mobile network among tenants (with shared functions and resources) and an *isolated slices* one (with dedicated functions and resources only) calls for new effective solutions. Therefore, a network slice broker component is desirable, which may act as an arbitration entity in charge of satisfying heterogeneous slice requirements as required by active tenants while at the same time guaranteeing the most efficient use of the infrastructure resources, as pointed out by [1]. An architectural proposal of a network slice broker has been previously considered in [2] where the authors directly plug the novel component into the 3GPP-compliant architecture. However, most of the existing works consider *a-posteriori* approaches efficiently orchestrating wireless resources only as a response to unexpected network changes.

The objective of our proposal, namely *ONETS*, is to design an efficient online broker that maximizes future revenues deriving from the satisfaction of upcoming network slice requests by analyzing past network slicing information while exploiting multiplexing gains coming from the overbooking of resources: this would directly translate into an overall system utilization increase. In particular, we provide $i$) a novel decisional model addressing the "exploration vs exploitation" dilemma, dubbed as Budgeted Lock-up Multi Armed Problem (BLMAB), $ii$) a detailed analysis of such a class of problems, including specific exploitable features to design an efficient solution maximizing multiplexing gains, $iii$) multiple variants of the proposed solution accounting for complexity and optimality properties along with performance upper bounds, $iv$) an exhaustive simulation campaign with synthetic traces

Fig. 1. 3GPP network slicing architecture.

to determine the expected benefits in large-scale scenarios and $v$) a proof-of-concept implementation using commercial hardware to prove the feasibility of our solution considering three network slices: enhanced Mobile BroadBand (eMBB) for Guaranteed Bit Rate (GBR), eMBB for Best Effort (BE), and Public Safety.

The reminder of the paper is organized as follows. Sect. II shows the architectural requirements of our solution. Sect. III lists the state-of-the-art. Sect. IV presents the class of decisional problems by deriving tight bounds and insightful properties. Sect. V proposes different online algorithms trading-off between complexity and sub-optimality. Sect. VI shows experimental results to prove our findings with synthetic network scenarios. Sect. VII presents our proof-of-concept built on top of commercial equipment and corresponding results. Finally, Sect. VIII provides our conclusions.

## II. NETWORK SLICING IN 3GPP

3GPP has defined a novel network architecture for network slicing support. In particular, the 3GPP working group SA2 [3] has already defined the basis for building an evolved core network infrastructure managing multiple slices on the same network infrastructure. The envisioned architecture is depicted in Fig. 1 which clearly differentiates between control plane (C-Plane) and user plane (U-Plane). In the control plane, new components are introduced to manage user authentication and registration (AMF), support multiple connection sessions (SMF), and instruct different routing policies (PCF). On the other hand, the traditional core user plane functionalities are decentralized towards multiple dedicated User Plane Functions (UPFs) managing distinct data networks (DNs) through the next-generation Radio Access Network (ngRAN), therefore allowing for packet processing and traffic aggregation at the network edge. This new architecture builds on network functions virtualization and also enables flexible multi-tenant deployments. In fact, RAN resources can be virtualized and dynamically chained to provision end-to-end slices with a dedicated SMF [4]. Interestingly, AMF (and PCF) can be still be shared among multiple slices when presenting service requirements commonalities. Based on this architecture, the Network Exposure Function (NEF) can be used as a direct

interface between the mobile network operator and the network slice tenants to access the virtualized network functions. NEF is envisioned to expose a list of available slice templates defining specific functions to be instantiated for given service requirements. Network slice request coming through the NP8 interface will then indicate the requested slice template based on the available ones. At this point, an arbitration entity is needed to grant (or deny) network slice requests. Once a network slice request is granted, a Network Slice Selection Assistance Information (NSSAI) indicator is propagated through all network components and advertised to incoming UEs through the RAN [5]. Based on the NSSAI, the AMF will select the SMF and a network slice will be successfully installed. Associated UEs might then indicate in the RRC signaling the NSSAI to be used for serving its traffic.

In Fig. 1 we depict the proposed location of the arbitration entity in charge of granting or denying network slice requests, referred in the paper as *Network Slice Broker*. In the following we review the state-of-the-art of network slicing solutions available in the literature to meet the functionality required by an online network slice broker.

## III. RELATED WORK

Network slicing is an emerging topic of 5G (and beyond) research community given its business relevance [6] and the recent definition of the architecture by 3GPP [7]. Related work in the field can be split into two main categories.

*Admission Control:* The goal of a network slice broker is to coordinate slice requests as well as to ensure the dynamic provisioning of networking resources in an efficient way. In [8], the slicing admission control and scheduling tasks are introduced by means of an NP-hard optimization problem formulation. The authors suggest a reinforcement learning-based approach, which exploits resource utilization and traffic predictions to improve the overall admission control decision process. Their analysis details on the achievable benefits regarding the number of admitted network slice requests and the system capacity utilization. [9] suggests a heuristic-based prioritized admission control mechanism. The novelty here stands in the possibility to adapt the algorithm to both inter- and intra-cell admission control problem. The admission procedure of a new UE belonging to the same slice is done taking in account the current traffic load and the available resources. Only if there are enough resources to guarantee and satisfy at least the requirements on a predefined minimum data rate, the procedure will be successfully ended. [10] presents a framework for enabling negotiation, selection and assignment of network slices for requesting applications in future 5G networks. Based on different QoS Class Indicators (QCIs), different virtualized networks or network slices are selected and assigned to users demanding for a specific service. Subsequently, static or dynamic routing mechanisms are used to treat data packets according to the QCI and security requirements and to flexibly select network functions and service function paths through a NS. A similar approach can be found in [11]. In the same context, [12] introduces a practical admission control solution for network slicing pursuing at maximizing the overall network revenues following

a simple pricing model for RAN slices. [13] mathematically analyses the admission control issue for network slicing and proposes an algorithmic solution by applying machine learning concepts. However, it relies on offline approach, which gathers several network slice requests within a fixed time window while selecting (some of) them as so to maximizing the overall network utilization. As pointed out in [14], network slicing implies each slice instance to have a dedicated set of allocated network resource, therefore introducing novel challenges in the management of mobile systems. Indeed, in previous generations of mobile networks, the resources to be assigned were mainly radio resources, while in 5G (and future) networks it is commonly accepted that this will impact also on the core network resources splitting. This easily matches with the research effort in Network Function Virtualization (NFV) and Software Defined Network (SDN) fields toward the deployment of a modular and flexible 5G network, as discussed in [15].

*Resource Orchestration:* Several solutions supporting virtualization of network resources have been already designed to optimize network resource orchestration in slicing-enabled scenarios and improve QoE. The authors of [16] focus on the RAN domain, and propose an efficient sharing of the radio resources which performs adaptive provisioning and isolation of network slices by means of a bi-level scheduling approach. Similarly, in [17], the authors present a reinforcement learning-based RAN orchestration solution focusing on latency constraints. Their framework allows for adaptive resource slicing decisions without prior knowledge of slice-specific traffic demands or channel statistics. A slice selection function is a key element in the future core network architecture. [18] attempts to fill the gap between the emerging new service requirements in terms of performance and efficiency and the possible realization of the concept by proposing a new slice selection mechanism allowing the UE to connect to multiple slices based on service types. The authors of [19] come up with a set of requirements to guide the design of the 5G RAN network and to support service provisioning. In particular, they model the three-fold relationship among end-users, base station and the network slice, and investigate user admissibility, slice association and bandwidth allocation aspects for diversified use cases with various quality of service (QoS) requirements in an offline optimization manner.

Differently from above-mentioned solutions, to the best of our knowledge, the work presented in this paper is the first of its kind that aims at designing and evaluating an automated online network slice broker that instantaneously decides which set of slices to be accepted while opportunistically pursuing the network slicing multiplexing gain maximization.

## IV. SYSTEM DESIGN

We consider a telecom service provider (hereafter dubbed *Operator*) making available its own infrastructure to external network tenants, for e.g. vertical industries. The operator owns the entire access infrastructure with a fixed transmission capacity $C$.[1] The available tenants set $\mathcal{I}$ is known a-priori, as each external domain must subscribe beforehand to submit network slice requests and get access to the infrastructure. Each tenant $i \in \mathcal{I}$ can request a network slice $s \in \mathcal{S}$ best matching the network characteristics required amongst available network slice templates (NSTs). Such templates are considered as fixed based on the operator's business model [6]. Each network slice request $s = \{R^{(s)}; L^{(s)}\}$ comprises an amount of physical wireless resources, $R^{(s)}$, to be assigned to tenant users within the network slice $s$, and a time duration $L^{(s)}$ expressed in seconds. Such parameters might be tailored to particular services and might be modified according to new service requirements. Indeed, the network slice requirements define the Service Level Agreement (SLA) between the operator and the infrastructure tenant. Any failure to comply with such SLA might result in undesired penalties from the operator side. Moreover, we assume a two-layer scheduling approach as commonly adopted within the network slicing context [16], [17]. On the one side, an inter-slice scheduler is in charge of defining the best radio allocation strategy pursuing the satisfaction of networking and isolation requirements. On the other side, a lower layer intra-slice scheduler enforces the assignment of the pre-allocated radio resources to connected slice end-users. A correct sizing of the allocated resources would benefit slice isolation and energy consumption. Nevertheless, its dimensioning depends on unknown traffic statistics and real-time stochastic variables, such as wireless channel quality and end-user mobility. Therefore, we tackle this challenging scenario by focusing our work on the higher-level inter-slice resource allocation, leaving the definition of standard intra-slice scheduling strategies open to address tenant-specific requirements.

In the next subsection, we first introduce the objective of the operator to increase its revenue opportunities while keeping SLA violations negligible, and then we detail the system model behind such problem as well as our proposed solution.

### A. System Model

Let us consider a set of $T$ time instants denoted by $\mathcal{T} = \{1, \ldots, T\}$. Each tenant $i \in \mathcal{I}$ asks for a network slice template $s$ at time instant $t \in \mathcal{T}$, modeled with the variable $r_i^{(s)}(t)$. The requests distribution is obtained from an i.i.d. random variable, namely inter-arrival time $\Delta t$, exponentially distributed with rate $\phi_i$. Inter-arrival rates, $\phi_i$, are drawn from a Pareto distribution with mean $\rho$ and standard deviation $\zeta$, determining the level of heterogeneity.[2]

The tenant selects a network slice template $s = \{R^{(s)}; L^{(s)}\}$ to be issued with the network slice request. In our analysis, this choice is taken based on tenants network slice requirements to efficiently drive the slice selection process. We also assume that each tenant can only ask for a single network slice at a given time and, tenants can be granted only a single network

---

[1] In our work, we consider the network slicing paradigm uniformly applied across the network, as currently stated by 3GPP [7].

[2] Note that this assumption relies on the traffic flow behavior, as suggested in [20]. However, as shown in Section V-A.2, it might be relaxed to bring interesting findings.

slice request, i.e., multiple network slices assigned to the same tenant cannot overlap in time, $t \geq t_{-1} + L^{(s)}$, where $t_{-1}$ denotes the previous decision interval, if the slice template $s$ has been granted in the previous request $r_i^{(s)}(t_{-1})$.[3] However, it can be easily extended by assuming tenants asking for multiple slices at the same time as distinct virtual tenants.

Without loss of generality, we express network slice request as $r_{i,t}^{(s)} = \{R_{i,t}^{(s)}; L_{i,t}^{(s)}\}$ and the problem as follows: Upon receiving a network slice request, *the operator decides whether to accept or reject it in an online fashion, pursuing the objective of network slicing multiplexing maximization while still honouring the agreed guarantees (SLAs) for previously granted network slice requests.*

### B. Online Decisions: Exploration vs Exploitation Dilemma

Once a network slice request is received, the operator might decide (at runtime) to accept or reject it, based on different factors: $i)$ deterministic aspects and $ii)$ stochastic components. The former group includes the set of requirements for network slices currently running and the total available system capacity. The latter comprises random tenant choices when issuing the slice request, upcoming network slice requests and real network utilization within an allocated network slice. The operator can decide to allocate an incoming network slice request fitting into available network resources. However, this might prevent future network slice requests from being accepted, though they can further boosting the network utilization. This may negatively affect the overall online process, as the current selection decision is strongly tied to future admissions. While deterministic considerations can be efficiently taken into account, stochastic features need advanced mechanisms to drive the system towards a near-optimal system behavior.

*Multi-Armed Bandit Model:* When dealing with an online decision process, a plethora of mathematical tools and practical schemes helps to bound the space of solutions and provides affordable and sub-optimal results [21]–[23]. We focus in this dissertation on a subset of online algorithms considering sequential decisions with limited information. In particular, we envisage a gambler facing diverse game options to play, resulting in different gains. The player must sequentially select the best option (i.e., the tenant slice request) in order to maximize the profit (i.e., overall system utilization). This results in the fundamental *exploration vs exploitation* lemma during sequential experiments: The gambler needs to balance the exploitation of known tenant slice requests that paid well in the past and the exploration of upcoming tenant slice requests that might pay even more.

Sequential allocation problems fully match Multi-Armed Bandit models (MABs) [24]. In particular, the fundamental problem formulation is obtained from a casino use-case, where a gambler faces with multiple slot machines (i.e., bandits). Slot machine return unpredictable revenues obtained through unknown statistical functions. The gambler can play one coin at once to $i)$ observe the profit behavior of unexplored slot machines or $ii)$ keep playing with the one providing (in the

previous rounds) the best profit. The final objective is to maximize the overall profit after playing a finite number of rounds. While this model has been fully investigated, our problem needs substantial improvements to be treated as a novel variation of MAB.

### C. A Budgeted Lock-Up Multi-Armed Bandit Problem (BLMAB)

We build on top of the basic MAB our problem formulation by introducing three fundamental MAB variations: $i)$ multi-plays, $ii)$ limited budget and $iii)$ lock-up periods.

Let us consider each tenant $i$ as a bandit that, if pulled at round $t$, returns a certain reward $\eta_{i,t}$. Multiple tenants can ask (simultaneously) a network slice request, hence the gambler may play multiple bandits at the same round, i.e., she may select multiple tenants to be granted at the same time. To avoid a trivial solution pulling down all arms to maximize the total revenue, our formulation introduces a cost function for selecting those bandits: At every round, the player needs to select a batch ($K \geq 1$) of arms whose cost lies within the available budget.

We define a cost function $\lambda_{i,t}$ as the number of resource blocks (PRBs) used within the round $t$. The total budget available for selecting slice requests is $C$, that is the total available system capacity. This defines our admissibility region, as more than available PRBs cannot be allocated.

Since different network slice requests might come at different times and occupy the resources for fixed time intervals, we need to modify our model accordingly. Basically, we account for the case when a pulled-down arm does not return any payoff, as it directly translates into the case of a tenant selected to be granted at time $t$ but not interested in issuing network slice requests at that time. Although this behavior looks counter-intuitive, it provides the effective means for deeply learning the tenant behaviors (by considering the frequency of network slice requests and the real slice utilization) and predict future requests.

On the other side, a network slice request $r_{i,t}^{(s)}$ granted for tenant $i$ at time $t$ must be considered active for the next rounds $t$ until $L^{(s)}$ expires. Both features are taken into consideration by introducing the concept of lock-up periods: On each round $t$, if a lock-up period is running, the gambler must select the same arm as in the previous round (however, the gambler can still select multiple arms in that round).

A self-explained example is depicted in Fig. 2, where three different rounds are highlighted ($t_k$, $t_m$ and $t_n$). In the first case, the selection policy decides to grant tenant 1 but no network slices are issued from this tenant, returning no rewards. In the second case, the selection policy grants tenant 1 asking for a network slice (and hence returning a reward). It also reselects the previous tenant (5) as its lock-up is still running. In $t_n$ the selection policy can select tenant 4 and 1 getting rewards, as they ask for new network slices. The case of a traffic request coming from a tenant and not pulled by the broker, will simply force the tenant to perform a second request in a subsequent time instant.

---

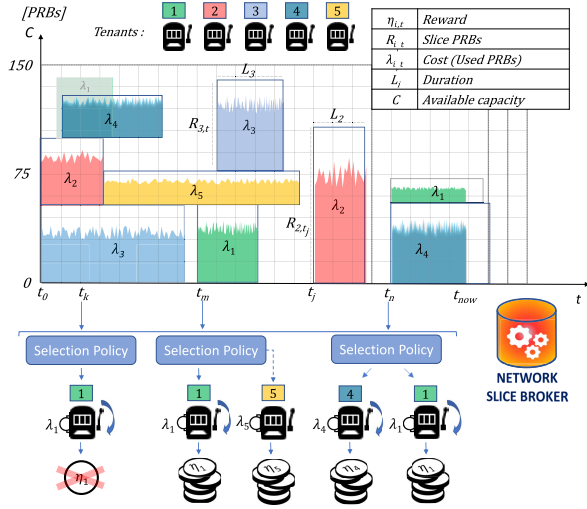[3]This assumption makes tractable the analysis in Section IV-C.

Fig. 2. BLMAB problem example applied to an online network slicing brokering process.

All in all, we can express the *reward* $\eta_{i,t}$ as the following

$$\eta_{i,t} = \alpha \frac{R_{i,t}^{(s)}}{C} + (1-\alpha) \frac{R_{i,t}^{(s)} - \lambda_{i,t}}{R_{i,t}^{(s)}}, \qquad (1)$$

where $\alpha \in [0,1]$ is a weight parameter, and it holds

$$\lambda_{i,t} \le R_{i,t}^{(s)} \le C, \qquad (2)$$

so that no negative values are obtained, i.e., $\eta_{i,t} \in [0,1]$. Specifically, the reward accounts for the global amount of resources asked within the slice request (left side of Eq. (1)) as well as for the multiplexing gain, i.e., the ratio between what has been really used and what is being asked (right side). The rationale behind relies on the concept of discovering bargains. Tenants underutilizing assigned resources are preferred with respect to the ones fully using them.[4]

Operators may reuse spare resources to allocate additional network slice requests so as to increase the network utilization (and, in turn, to increase overall system revenues). Additionally, we also take into consideration the total amount of PRBs, as operator might prefer to assign resources to tenants asking (and paying) for more resources, again pursuing the system utilization maximization. $\alpha$ provides a trade-off between those different metrics. However, in case of monitoring information not available, the second term will be null and the reward is equal to $\eta_{i,t} = \alpha \frac{R_{i,t}^{(s)}}{C}$. Please note that, when tenant(s) is(are) selected to be granted with no pending slice requests, the total amount of PRBs asked is $R_{i,t}^{(s)} = 0$ resulting in reward $\eta_{i,t} = 0$. Notably, as explained before, the reward expressed in Eq. (1) indirectly accounts also for the tenants behavior, such as the inter-arrival time between consecutive slice requests. Every round $t$, the operator selects tenants to be granted for being allocated through a set of binary actions $\mathcal{A}_t$. We can now formulate our problem as the following.

---

[4]In this paper we assume uniform pricing for slice resources.

---

**Problem** ONLINE-SLICING:

$$\max \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}} \eta_{i,t} a_{i,t}$$

$$\text{s.t.} \sum_{i \in \mathcal{I}} \lambda_{i,t} \le C, \quad \forall t \in \mathcal{T}; \text{budget}$$

$$a_{i,t} \in \mathcal{A}_t \subseteq \{0,1\}^{|\mathcal{I}|}, \quad \forall t \in \mathcal{T}; \text{multi-plays}$$

$$a_{i,t} \ge a_{i,t-1} \mathbb{1}(t - t_i^{\text{START}} \le L_i^{(s)}), \quad \forall t \in \mathcal{T}; \text{lock-up}$$

where the last constraint (:lock-up) imposes to select the same arm as in the previous turn, if the lock-up period is still running. For the sake of completeness, we wish to highlight that we assume $a_{i,-1} = 0$ as initial condition of the problem. With some abuse of notation, we denote $t_i^{\text{START}}$ as the round when slice request has been allocated for tenant $i$ and $\mathbb{1}(\cdot)$ is an indicator function providing value 1, if the condition in brackets is satisfied.

*Lemma 1:* A network slice online brokering can be mapped onto a multi-armed bandit (MAB) model with novel variations, such as $i$) multi-plays, $ii$) limited budget and $iii$) lock-up periods. Therefore, Problem ONLINE-SLICING falls into a new class of MAB problems, namely Budgeted Lock-up Multi-armed Bandit problems (BLMAB).

*Sketch of Proof:* Problem ONLINE-SLICING is a specific instance of MAB, as stated in Lemma 1. We apply a reduction to Problem ONLINE-SLICING by assuming that only one tenant $i$ can be selected every round $t$, i.e., $\sum_{i \in \mathcal{I}} a_{i,t} \le 1$, $\forall t \in \mathcal{T}$. Additionally, we assume that each network slice request lasts a single round, i.e., $L^{(s)} = t - t_{-1}$. This implies that the first constraint (:budget) is always satisfied due to Eq. (2). Therefore, we can state that Problem ONLINE-SLICING can be easily reduced with polynomial reductions to a MAB problem. □

Lemma 1 states that every advanced algorithm solving BLMAB also provides solutions to our network slicing online brokering problem, as we will show in Section V. Additionally, Lemma 1 clarifies that such a new class of problem, namely BLMAB exhibits higher complexity w.r.t. the classical MAB problem due to additional constraints. Therefore, we propose the following:

*Proposition 1:* The BLMAB problem is more complex than the stochastic multi-armed bandit problem, where the forecaster can switch arms at every round. Therefore, a regret lower bound of the stochastic MAB is also a regret lower bound of the BLMAB problem, considering the same number of rounds.

Proposition 1 suggests that we can use the lower bound suggested for MAB to provide a reference point to our mechanism. As we show in the next section, we build on top of such a lower bound to further provide findings on tight bounds for this given class of decision policies.

*D. Regret Lower Bound*

The performance of MAB algorithms can be measured by given metrics, namely regret. The regret denotes the difference in terms of rewards between actions played according to an arbitrary selection policy and the optimal selection policy

aware of all reward distributions [25]. Let us consider a player selecting a set of arms $\mathcal{K} \subseteq \mathcal{I}$ every round, such that the budget constraint is fulfilled (refer to constraint:budget of Problem ONLINE-SLICING). Each arm $i \in \mathcal{K}$ is associated with a univariate known probability density function $f(x, \theta_i)$, where $x$ is the random variable, all $\theta_i \in \Theta$ are unknown parameters and $\Theta$ represents the set of the reward distributions. Every time arm $i$ is pulled, it returns a reward $v_i$ drawn from $f(x, \theta_i)$ such that $\mu(\theta_i)$ is the mean of $v_i$. Let us now consider $\pi = \{\pi(t)\}_{t=1}^T$ as an arbitrary selection policy. The optimal cumulative reward is provided by selecting $|\mathcal{K}|$ arms with the highest reward, i.e., $\{i | i \in \{\sigma_1, \sigma_2, \cdots, \sigma_{|\mathcal{K}|}\}\}$, where $\sigma$ is a permutation vector of $\mathcal{I}$ following a reward decreasing order. Mathematically, it holds that the regret is obtained as

$$R_T^\pi(\Theta) = T \sum_{i=1}^{|\mathcal{K}|} \mu(\theta_{\sigma(i)}) - \mathbb{E}_\pi[\sum_{t=1}^T v_{\pi(t)}(t)]$$

$$= T \sum_{i=1}^{|\mathcal{K}|} \mu(\theta_{\sigma(i)}) - \sum_{i=1}^{\mathcal{I}} \mu(\theta_i)\mathbb{E}[W_i(T)]; \quad (3)$$

where $W_i(t)$ is the number of rounds arm $i$ has been pulled down till time instant $t$, and $T$ represents the last instant of the time period $\mathcal{T}$. More in details, the first term of Eq. (3) represents the optimal cumulative reward at time $T$, ideally achievable only by selecting the best $|\mathcal{K}|$ arms in all the decision intervals, while the second term identifies the expected reward of the experiment, defined as function of the arms' mean reward $\mu(\theta_i)$ and algorithm's arm selection rates $\mathbb{E}[W_i(T)]$. If the selection policy is *uniformly good*, then $R_T^\pi(\Theta) = o(T^b), \forall b > 0$, where $o(\cdot)$ refers to the little-o notation, and, in turn, it holds that

$$\lim_{T \to \infty} \sum_{i=1}^{\mathcal{I}} T^{-1} \mu(\theta_i) \mathbb{E}[W_i(T)] = \sum_{i=1}^{|\mathcal{K}|} \mu(\theta_{\sigma(i)}). \quad (4)$$

Therefore, we can express the lower bound of the regret for any uniformly good policy as the following

$$\lim_{T \to \infty} \inf \frac{R_T^\pi(\Theta)}{\log T} \geq \sum_{i: \mu(\theta_i) < \mu(\theta_{\sigma(|\mathcal{K}|)})} \frac{\mu(\theta_{\sigma(|\mathcal{K}|)}) - \mu(\theta_i)}{H(\theta_i, \theta_{\sigma(|\mathcal{K}|)})},$$

$$(5)$$

where $H(\theta_u, \theta_v) = \mathbb{E} \log(\frac{f(x, \theta_u)}{f(x, \theta_v)})$ is the relative entropy of one statistical distribution with respect to the other, characterized by $\theta_u$ and $\theta_v$, respectively.[5]

## V. ONLINE NETWORK SLICE BROKER

Although online network slicing brokering solutions can benefit from being fully customized and not requiring human interventions, a proper design needs advanced algorithms to achieve near-optimal performance. We focus on different classes of solutions, which are explained and analyzed next.

Algorithms proposed for classical multi-armed bandit (MAB) problems trade off the exploitation of good payoffs with exploration of unknown rewards. This makes such solutions robust and practical resulting in $O(\log T)$ as expected

[5]We refer the reader to state-of-the-art literature such as [26] and [27] for a complete list of the derivation steps.

---

**Algorithm 1 eUCB:** Selection Algorithm $\pi$ to Select the Next Batch of Arms to Pull Down While Guaranteeing a Fixed Budget

---

**Input:** $\mathcal{I}, T, C$
**Initialization:** $i = 0, W_i = 0, \bar{\theta}_i(0) = 0 \in \bar{\Theta}, \forall i \in \mathcal{I}, \mathcal{L} \leftarrow \emptyset$
**Procedure**
1: **for all** $i \in \mathcal{I}$ **do**
2:     GET $v_i$
3:     UPDATE $\bar{\theta}_i(0)$
4:     $W_i = W_i + 1$
5: **end for**
6: **for all** $t \in T$ **do**
7:     **for all** $i \in \mathcal{I}$ **do**
8:         $\hat{\theta}_i(t) = \bar{\theta}_i(t) + \sqrt{\frac{2 \log t}{W_i}}$
9:     **end for**
10:     $\mathcal{R} = \{i\} \leftarrow$ Problem D-ONLINE-SLICING($C, \mathcal{L}(t), \hat{\Theta}(t)$)
11:     **for all** $i \in \mathcal{R}$ **do**
12:         GET $v_i$
13:         UPDATE $\bar{\theta}_i(t)$
14:         $W_i = W_i + 1$
15:     **end for**
16:     UPDATE $\mathcal{L}(t)$
17: **end for**

---

cumulative regret. When important variations are considered, the overall complexity might be perturbed by additional factors, e.g., the lock-up time periods.

### A. Index-Based Policy Algorithms

The first class of selection algorithms computes an index per arm. Such an index is updated based on the set of arms already selected in the past, including information regarding the total time elapsed. By doing that, it  guarantees the uniformly goodness property (as explained in Section IV-D), which might optimally bound the performance. We consider the classical Upper Confidence Bound (UCB) solution, enhanced to address our BLMAB problem.

*1) Enhanced-UCB (eUCB):* Several works address the UCB solution considering different variations. We focus on the classical one proposed in [25] where we collect at every "attempt" the reward obtained from each arm and infer the mean of the statistical distribution $\bar{\theta}_i$. Without loss of generality, we assume that the density function is parameterized with its mean, i.e., $\mu(\theta_i) = \theta_i$. Clearly, the larger the number of attempts, the more accuracy on the distribution information. To avoid the negative influence of random effects, the authors include an additional term $G_{\text{ALG}}$ to give more weight to empirical distribution means obtained in longer time windows, than averages obtained in shorter time windows. Also, to address starvation issues, each empirical distribution is weighted with the number of times that an arm has been selected. Index per arm $i$ is formulated as $\hat{\theta}_i(t) = \bar{\theta}_i(t) + G_{\text{ALG}}$, where $G_{\text{ALG}} = \sqrt{\frac{2 \log t}{W_i(t)}}$, and $\bar{\theta}_i(t)$ is the empirical distribution mean until time $t$.

This could be envisioned as a training session where tenants subscribing for network slicing operations, may express their

interests on given slice template and, in turn, result in different initial fictitious rewards $v_i \sim f(x, \theta_i)$. It can be noticed that line 8 optimally solves an instantaneous version of the problem, dubbed as `D-ONLINE-SLICING`, assuming as input only one time instant $t$, the total budget, the lock-up time windows currently running and the empirical mean of reward distributions retrieved until that time. Please note that the instantaneous version significantly reduces the complexity of our problem, pursuing only the punctual reward maximization at time $t$ rather than the cumulative reward over time window $T$. Output of this problem is a set of arms indexes that are promptly selected in lines $9 - 13$. Line 14 updates the status of current lock-up periods for next selections.

*2) ONETS: Online NETwork Slice Broker:* While the *eUCB* algorithm achieves outstanding performance, it requires solving an Integer Linear Programming (ILP) problem at every round and thus, it is not a feasible solution in practice. UCB applied to the Budget-Limited Multi-Armed Bandit problem has been proven to be NP-HARD in [28]. Therefore we designed a lower complexity version of *eUCB*, namely **ONETS**, where exactly $K$ arms are selected every round while meeting the budget constraint. Clearly, this modification drastically reduces the computational time to $O(K)$ (see Alg.2) but at the performance cost of a sub-optimal solution (both effects are studied in Section VI).

The pseudocode is listed in Alg. 2. The idea is to substitute Problem `D-ONLINE-SLICING` with a practical procedure: Indexes $\hat{i}$ with active lock-up sessions are included in the next round (Line $9 - 10$), as the budget constraint has been already fulfilled in the previous round. If those indexes are not enough (less than $K$), we search the index among the remaining ones such that the empirical distribution mean is maximized while fitting into the budget left (line 14).

*Regret Upper Bound for ONETS:* We provide an upper bound analysis for the *ONETS* scheme. Let us consider set $\hat{\Theta} = \{\hat{\theta}_i\}$ of empirical distribution means per arm $i$ obtained within time window $T$, where $\hat{\theta}_i = \bar{\theta}_i + G_{\text{ALG}}$ (in case of *eUCB*, $G_{\text{ALG}} = \sqrt{\frac{2 \log T}{W_i(T)}}$). We can calculate the expected number of times arm $i$ is pulled down based on *ONETS* algorithm as follows.

$$
\mathbb{E}[W_i(T)]
$$
$$
= \int_0^\infty f(x, \hat{\theta}_i) \Pr(x \geq \max_{p \neq i} v_p) \mathrm{d}x
$$
$$
+ \int_0^\infty f(x, \hat{\theta}_i) \Pr_{\forall j \in \mathcal{I}}(\text{single } v_j \geq x) \Pr(x \geq \max_{p \neq \{i,j\}} v_p) \mathrm{d}x
$$
$$
+ \int_0^\infty f(x, \hat{\theta}_i) \Pr_{\forall j \in \mathcal{I}}(\text{multiple } v_j \geq x) \Pr(x \geq \max_{p \neq i} v_p) \mathrm{d}x
$$
$$
\leq \sum_{|\mathcal{H}|=0}^{|\mathcal{K}|} \sum_{\sigma=1}^{\binom{|\mathcal{I}|}{|\mathcal{H}|}} \int_0^\infty f(x, \hat{\theta}_i)
$$

**Algorithm 2 ONETS:** Selection Algorithm $\pi$ to Select $K$ Arms to Pull Down While Guaranteeing a Fixed Budget

**Input:** $K, \mathcal{I}, T, C$
**Initialization:** $i = 0; B = 0; n = 0; W_i = 0, \bar{\theta}_i(0) = 0 \in \bar{\Theta}, \forall i \in \mathcal{I}; \mathcal{L} \leftarrow \emptyset$
**Procedure**
1: **for all** $i \in \mathcal{I}$ **do**
2:   GET $v_i$
3:   UPDATE $\bar{\theta}_i(0)$
4:   $W_i = W_i + 1$
5: **end for**
6: **for all** $t \in T$ **do**
7:   **for all** $i \in \mathcal{I}$ **do**
8:     $\hat{\theta}_i(t) = \bar{\theta}_i(t) + \sqrt{\frac{2 \log t}{W_i}}$
9:   **end for**
10:   **while** $n \leq K \wedge B \leq C$ **do**
11:     **if** $\mathcal{L}(t) \neq \emptyset$ **then**
12:       $\hat{i} \leftarrow \mathcal{L}(t)$
13:     **else**
14:       $\hat{i} : \arg\max_{\mathcal{I} \backslash \mathcal{L}} \hat{\theta}_i(t)$
15:     **end if**
16:     **if** $B + \lambda_{\hat{i}} \leq C$ **then**
17:       $\mathcal{R} \leftarrow \mathcal{R} \cup \hat{i}$
18:       $B = B + \lambda_{\hat{i}}$
19:       $n = n + 1$
20:     **end if**
21:   **end while**
22:   **for all** $i \in \mathcal{R}$ **do**
23:     GET $v_i$
24:     UPDATE $\bar{\theta}_i(t)$
25:     $W_i = W_i + 1$
26:   **end for**
27:   UPDATE $\mathcal{L}(t)$
28:   $B = 0; n = 0$
29: **end for**

$$
\times \left( \prod_{j \in \mathcal{H}(\sigma)} \int_x^\infty f(y, \hat{\theta}_j) \mathrm{d}y \prod_{\substack{k \in \mathcal{I}, \\ \mathcal{I} \backslash \{i\} \\ \mathcal{I} \backslash \mathcal{H}(\sigma)}} \int_0^x f(y, \hat{\theta}_k) \mathrm{d}y \right) \mathrm{d}x. \quad (6)
$$

Considering a negative exponential distribution $f(x, \hat{\theta}_i) = \frac{e^{-\frac{x}{\theta_i}}}{\theta_i}$ and recalling that $\int_0^x f(x, \hat{\theta}_i) \mathrm{d}x = 1 - e^{-\frac{x}{\theta_i}}$ we can obtain the following upper bound for $\mathbb{E}[W_i(T)]$

$$
\sum_{|\mathcal{H}|=0}^{|\mathcal{K}|-1} \sum_{\sigma=1}^{\binom{|\mathcal{I}|}{|\mathcal{H}|}} \sum_{\substack{\phi \in \\ \wp(\mathcal{I} \backslash \{\mathcal{H}(\sigma)\}, \\ \mathcal{I} \backslash \{i\})}} (-1)^{|\phi|} \frac{1}{\hat{\theta}_i \left( \frac{1}{\theta_i} + \sum_{j \in \mathcal{H}(\sigma)} \frac{1}{\theta_j} + \sum_{p \in \phi} \frac{1}{\theta_p} \right)};
$$
$$
(7)
$$

where $\sigma$ is the permutation (index) of all elements included in $\mathcal{H}$, while $\wp(\mathcal{I})$ is the power set of all elements included in $\mathcal{I}$. Note that, in this paper, the reward distribution function can be approximated to a negative exponential distribution as we

assume an exponential distribution for the arrival time of slice requests. However, complex distributions can be used to derive advanced upper bounds. For the sake of brevity, we leave to the reader the derivation from Eq. (6) to Eq. (7). Also note that, the first two summations (on the left of Eq. (7)) can be also expressed as $\wp(K)$, however we prefer to explicitly keep them to provide the computational effort for solving that equation. By substituting the empirical distribution mean $\hat{\theta}_i$ with the value provided by the algorithm, assuming $L_i \ll T$ and using Eq. (3), we can obtain the upper bound of the regret, i.e., $\mathbb{E}[R_T^\pi(\Theta)] = O(log(T))$ if we apply *ONETS*. When $T \sim L_i$, $\mathbb{E}[R_T^\pi(\Theta)] = O(log(T) + L^*)$, where $L^* = \max_{i \in \mathcal{I}} L_i$, as also confirmed in [29].

### B. $\epsilon$-Greedy Algorithm

A greedy solution is the simplest algorithm for approaching BLMAB problems, where $K$ is not needed. It implies that the balance between exploitation and exploration is driven by a random $\epsilon$ value. A linear dependency of $\epsilon$ with the elapsed time $t$ helps the selection policy to explore "more" neighboring solutions during the first rounds (as inferred distribution means might not be accurate) while "trusting" more on known distributions along the evolution of the experiments. We set $\epsilon = \frac{b|\mathcal{I}|}{d^2 t}$ where $d \in \{0, 1\}$ and $b > 0$ are arbitrary values, as shown in Section VI. The algorithm (listed in Alg. 3) will select the best arms maximizing the reward (line 9) with probability $\epsilon$, whereas it will select arms randomly (line 11) satisfying (in both cases) the limited budget constraint (lines $14 - 17$). Please note that $\epsilon$-greedy algorithm does not require an initial training phase.

*Regret Upper Bound for $\epsilon$-Greedy:* Greedy solutions are proved to have a sub-linear regret. In particular, as shown in [26] the upper bound regret for $T \to \infty$ is expressed as $b/(d^2 T) + o(1/T) + O(1/T^{1+\epsilon})$. Since our lock-up period constraint might only affect the number of times sub-optimal arms are randomly selected, i.e., $L^* = \max_{i \in \mathcal{I}} L_i$, such an upper bound works also for our BLMAB considering that the probability to select a sub-optimal arm is at most the following:

$$
\begin{aligned}
P_{i \neq i^*}\{a_{i,t} = 1\} = &\frac{b}{d^2 t} + 2\left(\frac{b}{d^2} \log \frac{(t-1)d^2 e^{1/2}}{b|\mathcal{I}|}\right) \\
&\times \left(\frac{b|\mathcal{I}|}{(t-1)d^2 e^{1/2}}\right)^{\frac{b}{(5d^2)}} \\
&+ \frac{4e}{d^2}\left(\frac{b|\mathcal{I}|}{(t-1)d^2 e^{1/2}}\right)^{b/2},
\end{aligned}
\tag{8}
$$

where $i^*$ represents the optimal arm.

### VI. NUMERICAL RESULTS

In this section, we validate our findings through numerical simulations carried out using a commercial tool, MATLAB®. In particular, we deploy a budgeted lock-up multi-arm bandit (BLMAB) problem as discussed in Section IV-C. Network slice requests are generated following exponential distributions, as explained in Section IV, with given $\rho, \zeta$ parameters.

---

**Algorithm 3** $\epsilon$-**Greedy:** Selection Algorithm $\pi$ to Select the Next Batch of Arms Based on $\epsilon$-Exploration Probability

**Input:** $\mathcal{I}, T, C, b, d$
**Initialization:** $W_i = 0, \bar{\theta}_i(0) = 0 \in \bar{\Theta}; \forall i \in \mathcal{I}, \mathcal{L} \leftarrow \emptyset$
**Procedure**

1: **for all** $t \in T$ **do**
2:    $\epsilon = \min\{1, \frac{b|\mathcal{I}|}{d^2 t}\}$
3:    **while** $(C - B \geq 0) or (\mathcal{I} \neq \emptyset)$ **do**
4:      **if** $\mathcal{L}(t) \neq \emptyset$ **then**
5:        $\hat{i} \leftarrow \mathcal{L}$
6:      **else**
7:        GET $z \in [0, 1]$ (uniformly distributed)
8:        **if** $z > \epsilon$ **then**
9:          $\hat{i} : \arg\max_{\mathcal{I} \setminus \mathcal{L}} \bar{\theta}_i(t)$
10:       **else**
11:         $\hat{i} : rand(\mathcal{I} \setminus \mathcal{L})$
12:       **end if**
13:      **end if**
14:      **if** $B + \lambda_i \leq C$ **then**
15:        $\mathcal{R} \leftarrow \mathcal{R} \cup \hat{i}$
16:        $B = B + \lambda_i$
17:      **end if**
18:    **end while**
19:    **for all** $i \in \mathcal{R}$ **do**
20:      GET $v_i$
21:      UPDATE $\bar{\theta}_i(t)$
22:      $W_i = W_i + 1$
23:    **end for**
24:    UPDATE $\mathcal{L}(t)$
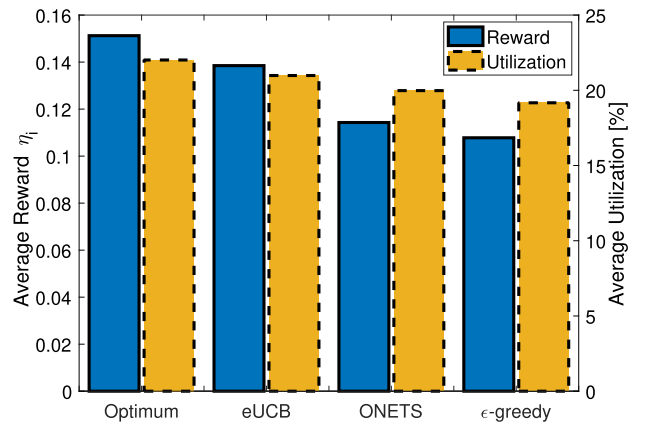25:    $B = 0$
26: **end for**

---



Fig. 3. Performance evaluation of BLMAB Heuristics versus the *Optimum* and *eUCB* solutions.

Every round, a selection policy is invoked to select a batch of tenants $K \geq 1$ to be granted. When a tenant is granted, it can (randomly) choose the slice template $s \in \mathcal{S}$ for its own traffic. Network traffic utilization is obtained in terms of used PRBs, as a sequence of i.i.d. random variables upper-bounded to the available number of slice template resources. All our simulations are run and results averaged over 1000 random seeds to cope with randomness effects, providing a 95%
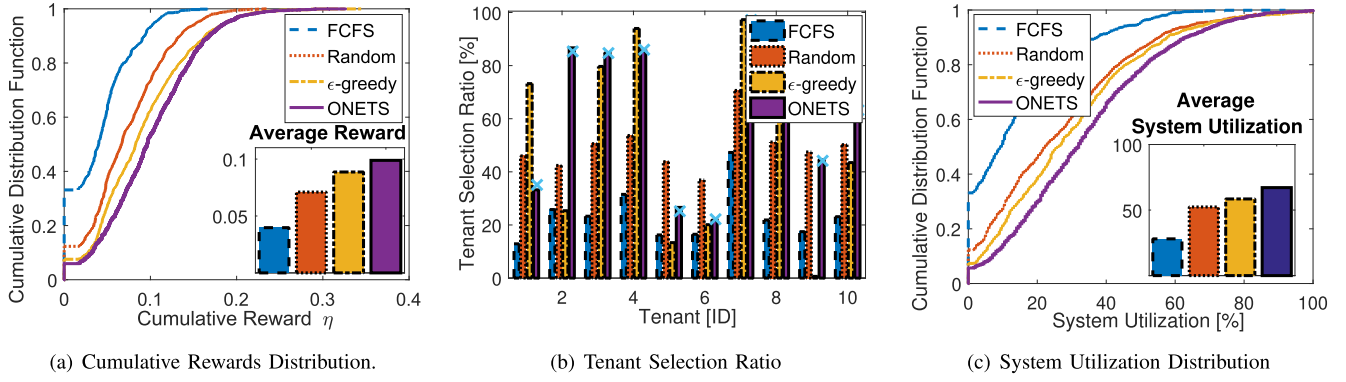
Fig. 4. Performance evaluation of different BLMAB selection policies for a 10 tenants scenario.

TABLE I

SIMULATION PARAMETERS

| System Parameters | Values |
|---|---|
| $|\mathcal{I}|$ | 10 tenants |
| $|\mathcal{S}|$ | 10 slice templates |
| Capacity ($C$) | 150 RBs (15 MHz + MIMO)[30] |
| $\rho; \zeta$ | 100; 0.1 |
| $\alpha$ | 0.5 |
| Time horizon ($T$) | 10000 rounds |
| $\epsilon$-greedy ($b; d$) | 10; 0.01 |
| K (ONETS) | 6 |

TABLE II

COMPUTATIONAL LOAD

| Solutions | 5 Tenants | | 10 Tenants | | 15 Tenants | |
|---|---|---|---|---|---|---|
| | Sim. | Inst. | Sim. | Inst. | Sim. | Inst. |
| Optimum | 24109 s | | - | - | - | - |
| eUCB | 4708 s | 131 s | 11512 s | 543 s | - | - |
| $\epsilon$-greedy | 219 s | 0.3 s | 398 s | 0.5 s | 502 s | 0.6 s |
| **ONETS** | **322 s** | **1.2 s** | **501 s** | **1.9 s** | **847 s** | **2.6 s** |

confidence degree. The default system parameters are listed in Table I.

## A. BLMAB Solutions: Optimal vs Heuristics

Fig. 3 shows a comparison of all solutions previously described for a network scenario with 5 tenants, in terms of average reward $\bar{\eta}_i$ and average system utilization. As expected *eUCB* is the closest solution to the *Optimum* with *ONETS* closely following it.

*Optimum* results have been obtained through the commercial tool IBM CPLEX OPL® solver, used to solve Problem ONLINE-SLICING, whereas *eUCB* results are retrieved after solving Problem D-ONLINE-SLICING for any single time $t$. *ONETS* is run with $K$ empirically set to $K = 3$.

Table II shows the computational counterpart to the average reward and utilization results. We show the measured computational time for running every solutions for a single instance (referred as Inst.) as well as for the whole simulation period of 1000 rounds (referred as Sim.). As it can be observed, the performance gains previously observed from the *Optimum* and *eUCB* solutions come at a poor scalability with the number of tenants deeming them as unfeasible in practice. In the rest of the experiments we will not consider them anymore for feasibility reasons.



Fig. 5. Performance evaluation of different BLMAB selection policies when increasing the number of tenants.

## B. BLMAB Heuristics Benchmarking

Given the lack of existing solutions in this particular context, we consider for benchmarking purposes two baseline approaches. First, we consider a trivial selection policy, First Come First Served (*FCFS*), to accept all incoming network slice requests as far as there are enough resources in our network. Second, we consider a random selection policy process (*Random*) that chooses tenants (or subset of tenants) based on a uniform distributed random variable, while satisfying budget constraints.

*Arms Selection Policies:* In Fig. 4 we depict three key performance figures for a 10 tenants scenario. Fig. 4(a) shows the differences obtained by the different approaches in terms of reward. Fig. 4(b) shows the number of times each tenant $i$ is selected by the different approaches. Note that inter-arrival times $\Delta t$ of slice requests per tenant are exponentially distributed with rate $\phi_i$ (see Section IV). On the same figure, we also plot with cross signs the expected number of times each tenant is selected based on Eq. (6) as described in Section V. This result supports our model, as our analysis accurately predicts the number of times each tenant is selected by *ONETS*. Fig. 4(c) shows the system utilization percentage achieved with the different approaches. Based on these results *ONETS* fulfills its design objectives outperforming at different levels the different alternative approaches.

*Number of Tenants:* In Fig. 5, we study two performance metrics as the number of tenants is increased. On the left picture, we show the average utilization for all mechanisms
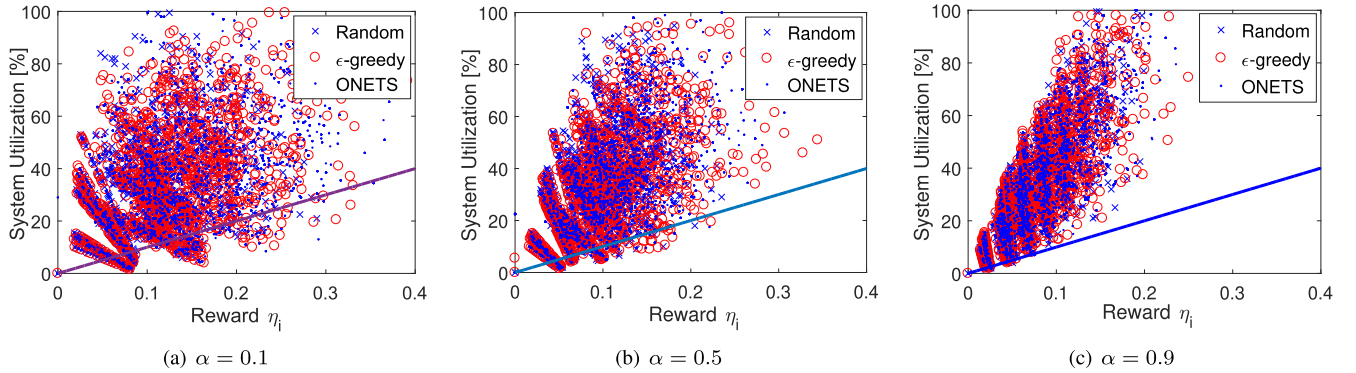
Fig. 6.    Scatterplot of reward versus system utilization for different $\alpha$ configurations.

compared with the total slice resource demand. While increasing the number of tenants intuitively leads to a higher average utilization, it also benefits the potential multiplexing gain as shown in the green area. On the right side, we show the average reward for an increasing number of tenants. *ONETS* outperforms the other solutions showing consistent gains in terms of reward, average utilization and multiplexing gain. These results suggest that operators would benefit of "opening" their networks to external tenants through network slicing given the potential gains in increasing the overall system utilization and corresponding profit.

*Network Slicing Multiplexing Control ($\alpha$):* In this section, we study our reward solution expressed in Eq. (1) in order to illustrate the impact of the configurable parameter $\alpha$. In particular, we show its behaviour in relation to system utilization. Fig. 6 shows a scatter plot of reward obtained per round compared with the system utilization achieved at that particular time, when different solutions are applied. The overall behavior suggests a strong dependency of system utilization from the game reward $\eta_i$, which validates our reward design: the larger the reward, the higher the multiplexing gains.

However, a proper tuning of the weight $\alpha$ might strongly influence and lead the system toward near-optimal steady states. When $\alpha$ equally distributes the weight (Fig. 6(b)), the system efficiency is equally distributed around the linear dependency line (dashed line). When $\alpha = 0.9$ in Fig. 6(c), the strong dependency results in a strong perturbation of results: a small variation of reward may cause a significant increase of utilization resulting in an unstable behavior when performed in an online fashion. All results show that *ONETS* translates higher rewards to higher multiplexing gains.

*SLA Protection vs Multiplexing Gains: ONETS* relies on the BLMAB framework ability to predict traffic behaviour based on past observations. However, outliers (i.e., traffic bursts) might lead to performance degradation and, in the worst case, to SLA violations.

In Fig. 7, we show the network slicing multiplexing gains versus the average SLA violation per tenant computed as the percentage of the number of times slice resources were not fully provided to tenants divided by the total number of slices granted. As it can be observed, our solution achieves high multiplexing gains ($> 30\%$) at limited SLA violation risk
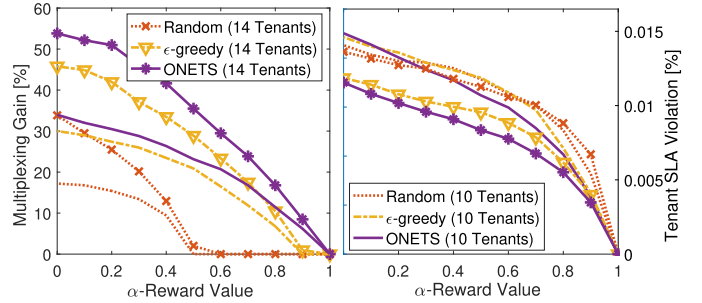


Fig. 7.    Performance evaluation of different BLMAB selection policies by varying the $\alpha$ configuration.
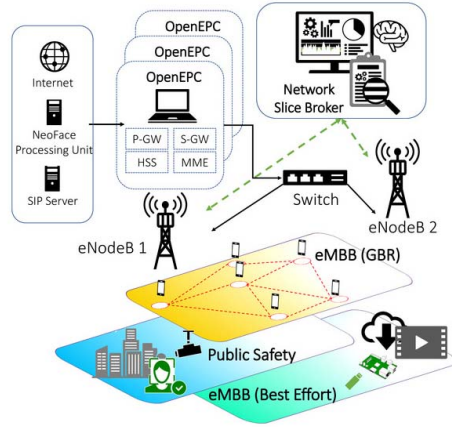
($< 0.015\%$). This trade-off is optimally driven by a proper tuning of the reward $\eta_{i,t}$, expressed in Eq. (1) .[6] Specifically, different $\alpha$ values might strongly influence the overall system behavior, as shown in Fig. 7. In case of $\alpha = 1$, only network slice request information is considered without any past information on the real slice utilization. This ensures no SLA violation but also no multiplexing gains. As $\alpha$ increasingly approaches to 0, past traffic information is considered in our model, allowing for resource over-provisioning and thus, multiplexing gains. However, this comes at the cost of an increasing SLA violation risk.
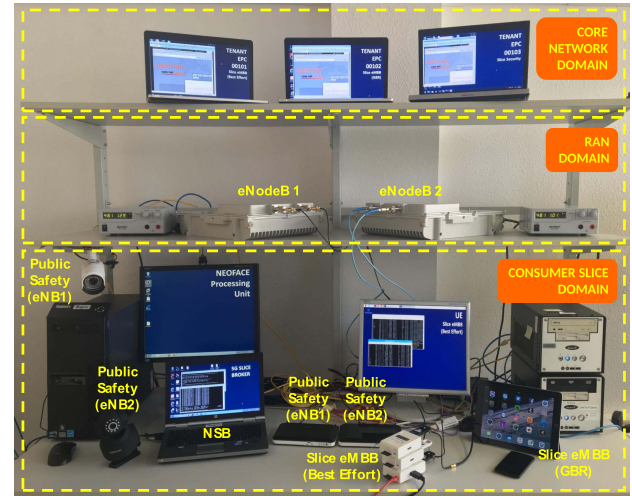
## VII. PROOF-OF-CONCEPT

### A. PoC Setup

In this section we describe the proof-of-concept implementation of our proposed *ONETS* solution. We built on available commercial hardware to setup a testbed comprising: $i$) three virtualized Evolved-Packet-Core (EPC) (one per slice), $ii$) 2 LTE eNBs connected to the EPCs, $iii$) multiple LTE devices generating traffic with different service requirements, such as mobile phones, surveillance cameras and USB dongles, $iv$) our *ONETS* solution implemented as a stand-alone software connected to the Local Maintenance Terminal (LMT) of the RAN environment. All equipment hardware specifications are listed in Table III.

---

[6]We assume the same reward model for the whole system. Advanced reward models differentiating customers classes are out of scope of this paper and might be considered in future extensions.

(a) Testbed blocks overview

(b) Real deployment

Fig. 8.   Network slicing proof-of-concept with 3 slices: eMMB (BE), public safety and eMBB (GBR).

TABLE III

PoC HARDWARE SPECIFICATIONS

| Equipment | Description | Ref. |
|---|---|---|
| Virtualized EPC | OpenEPC Rel. 6 | [31] |
| Face Recognition SW | NeoFace Facial Recognition | [32] |
| Surveillance Camera | HD, Motion Detection | [33] |
| UE | LTE USB Stick | [34] |
| Smartphones | iPhone (SE) and LG Nexus (5) | – |
| Router LTE | LTE with SIM-card slot | [35] |
| Smart devices | Raspberry Pi 2 | [36] |
| LTE Small cell | 15 MHz channel, LTE Band 3 | [37] |
| | DL 1775 MHz, UL 1870 MHz (FDD) | |

The OpenEPC software [31] contains all the functional elements of the 3GPP EPC up to Release 12. We deploy three separated virtual machines running on the same host machine. It automatically builds main LTE core network elements, such as HSS, AAA, S-GW, MME and P-GW. All the interfaces among them are virtualized through a hypervisor, VMWare Workstation. The host machine is provided with two external Gigabit ethernet interfaces: the former is used for an internet gateway connection, the latter is used for establishing the S1 interface with the RAN nodes. Regarding the commercial eNBs [37], they use 15MHz bandwidth and exploit MIMO technology, i.e., 150 PRBs per 1ms subframe (cfr. [38]). For practical reasons we carry out conducted tests: We abate wireless channel uncertainty providing the devices with CRC/SMA cables directly connected to the radio interface of the eNBs.

To support the network slicing concept, we build on the concept of RAN Sharing: This enables us to use the same RAN infrastructure for different Public Land Mobile Networks (PLMNs). We apply the Multiple Operator Radio Access Network (MORAN) approach to have dedicated network core domains sharing the same RAN facilities. Each UE connects through the same set of eNBs indicating the PLMN-id, i.e., the slice id, for being served. In our experiments, a fixed number of mobile cores is already instantiated (corresponding to the number of tenants in our system). When a network slice request is accepted, eNBs are dynamically configured to activate an additional PLMN-id and to route traffic associated with users under this PLMN-id to its dedicated MME (and virtualized EPC network).

We assume three different tenants registered for issuing network slice requests with different traffic characteristics: $i$) enhanced Mobile BroadBand (eMBB BE) slice generating FTP file transfers best-effort traffic, $ii$) eMBB GBR generating multiple Voice over LTE (VoLTE) traffic streams emulating an audio conference system, $iii$) a Public Safety slice for video surveillance.

Raspberry Pis [36] with LTE USB dongles [34] are used to generate eMBB BE traffic. Tenants might ask for a new network slice only if an own network slice is not already running. Our network slice broker dynamically receives network slice requests and at run-time decides whether to accept the network slice and configure the eNBs accordingly. eNBs provide an LMT interface to properly tune the number of physical resource blocks (PRBs) assigned per PLMN. Once the slice is accepted and correctly instantiated, the network slice broker monitors the slices traffic to retrieve statistical information for future network slicing decisions. An overview of the system is depicted in Fig. 8(a). In the eMBB GBR slice case, commercial cellular phones and tablets are used for generating voice traffic.

As one of the slices in the testbed is tailored to public safety purposes, we deploy surveillance cameras in our testbed. We connect the IP cameras to LTE routers [35], which are, in turn, connected to our eNBs. The cameras [33] are provided with motion detection features. This introduces a bursty traffic source for our experiments. In addition, an advanced face recognition software [32] is fed with video streaming traffic to detect face recognition matches to faces stored in a database. The recognition server is attached to the P-GW through the SGi interface. A target list is already loaded in the recognition server. When the face recognition software matches a known
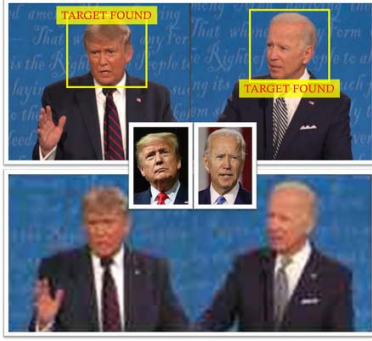
Fig. 9.   Sample of video surveillance face detection: successful (upper side) and failed (lower side).

target, a yellow square appears around the face detected (see Fig. 9). However, a detection threshold parameter may affect the detection process. Indeed, if the quality of the video stream is below a pre-determined threshold, the detection process may fail and the target might not be correctly recognized. The images quality is dynamically adjusted based on the LTE channel condition.

The IP camera stores video streams while a VLC server dynamically encodes the live-video based on the channel quality feedback from the recognition server. In Fig. 8(b) the different proof-of-concept components are depicted, where we highlight the service domain (Consumer slice domain), the RAN infrastructure (RAN domain) and the core network domain where the three EPCs are deployed.

### B. PoC Performance Evaluation

We evaluate our *ONETS* online network slicing solution in the proof-of-concept setup previously described. Our goal is to analyze the feasibility of our approach and get insights on the potentially achievable gains, i.e., exploit the multiplexing gain between the amount of resources assigned to a particular network slice and the actual slice resources utilization in time.

We consider three slices that are sequentially introduced in our system and dimensioned for peak demand. First, we introduce a baseline eMBB (Best-Effort) slice requesting $80\%$ of the system capacity for FTP transfers. Fig. 10(a) depicts the traffic pattern in time of this slice and the difference to the granted slice limit. Second, we introduce a Public Safety slice for video surveillance, see Fig. 10(b), where two surveillance cameras upload video streams for face recognition within a 1500 seconds time window and illustrate the difference to the granted traffic slice limit ($40\%$ of the system capacity). Finally, we introduce an eMBB (GBR) slice for audio conferences. We considered 30 devices generating voice calls. Each device is provided with a custom SIM-card, configured in our core domain to belong to a single PLMN-id, i.e., a single network slice. eNBs are configured to allocate an eMBB (GBR) network slice with $15\%$ of system capacity as demand. Fig. 10(c) shows the measured utilization versus the granted slice limit.

In Fig. 11(a), we show the dynamic system behavior when *ONETS* is applied. The eMBB (Best-Effort) network slice is dimensioned to use $80\%$ of the system capacity. Initially, the slice is fully using its resource allocation. During this period,
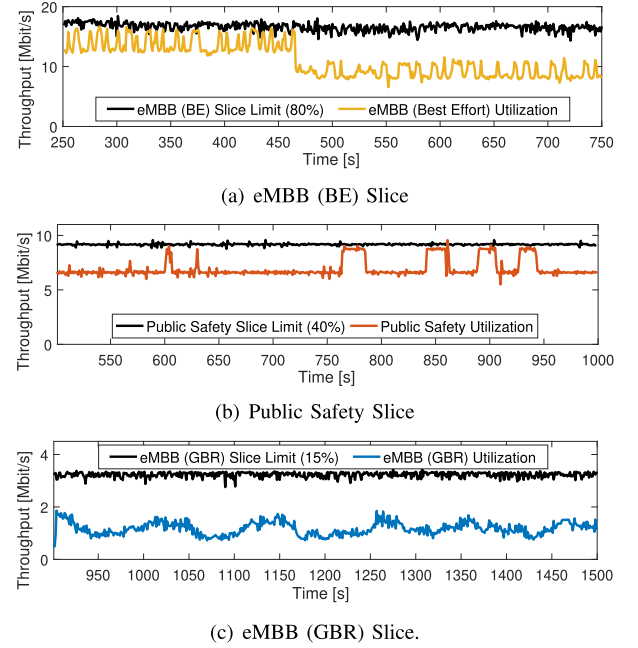


(a) eMBB (BE) Slice



(b) Public Safety Slice



(c) eMBB (GBR) Slice.

Fig. 10.   Network slices traffic: granted vs measured.



(a) Network slices aggregated offered load



(b) System utilization
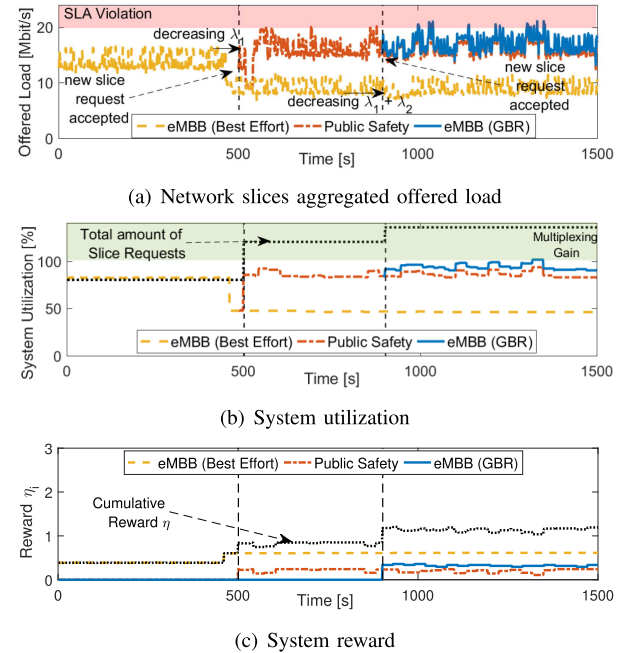


(c) System reward

Fig. 11.   *ONETS* online network slice Broker in action sequentially granting network slices.

if new network slice requests arrive, the system might reject them if they are above the leftover capacity.

After 480 seconds, we reduce the eMBB (BE) slice offered load by reducing the FTP file transfers. Our *ONETS* solution automatically detects a change in the system utilization ($\lambda_1$) and triggers the selection policy to consider admitting new network slice requests. When the Public Safety network slice request arrives, the system checks its feasibility and allocates it resulting in a higher reward ($\eta_1 + \eta_2$), as shown in Fig. 11(c).

After 900 seconds, the traffic associated to the eMBB (BE) slice is decreased again. The system capacity variation is

detected $(\lambda_1, \lambda_2)$ and a new network slice admitted into our system: eMBB (GBR). As its traffic is scheduled, a higher system reward is achieved $(\eta_1 + \eta_2 + \eta_3)$. The multiplexing gain is shown in Fig. 11(b), where the system utilization is compared with the aggregated granted network slice resources (the green area indicates a utilization above 100%).

As it can be observed, the network slicing multiplexing gains achieved with *ONETS* allow for increasing the number of slices that can be accepted in the system. In this example, network slice requests can be admitted up to $\approx 120\%$ of the system capacity, thus *virtually* increasing the *effective* capacity of the system and the achievable profit, accordingly. The cost of this gain is shown in Fig. 11(a) where after admitting in the system the second and third slice request, there are peaks of offered load that hit the maximum available capacity and thus, the SLA protection level could be threatened.

## VIII. Conclusion

One of the key novel concepts of 5G (and beyond) networks has been recently identified in the Network Slicing paradigm, driven by use cases which are very diverse and sometimes with extreme requirements, e.g., automated driving, tactile internet, mission-critical. In this paper, we proposed and analyzed *ONETS*: an Online NETwork Slice broker solution that builds on the budgeted lock-up multi-armed bandit theory to design a low-complexity solution that maximizes network slicing multiplexing gains, achieving the accommodation of network slice requests in the system with an aggregated level of demands above the available capacity.

Our results show that *ONETS* $i$) is feasible in practice as it has been successfully implemented and tested on top of a commercial LTE system, $ii$) the achievable multiplexing gains are significant and increase according to the number of slices in the system, $iii$) *ONETS* clearly outperformed naïve or greedy solutions (FCFS, Random, $\epsilon$-greedy) for the considered scenarios, $iv$) its computational complexity is in the same order of magnitude of a simple greedy solution ($\epsilon$-greedy), $v$) *ONETS* aggressivity for achieving network slicing multiplexing gains is a configurable parameter ($\alpha$) that can be freely tuned by operators according to proprietary policies and desired SLA protection levels.

## References

[1] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, and X. Costa-Perez, "Overbooking network slices through yield-driven end-to-end orchestration," in *Proc. 14th Int. Conf. Emerg. Netw. Exp. Technol.*, Dec. 2018, pp. 353–365.

[2] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 32–39, Jul. 2016.

[3] *System Architecture for the 5G System*, Standard 3GPP TS 23.501 v0.3.1, 3rd Generation Partnership Project (3GPP), Mar. 2017.

[4] X. Li, R. Ni, J. Chen, Y. Lyu, Z. Rong, and R. Du, "End-to-end network slicing in radio access network, transport network and core network domains," *IEEE Access*, vol. 8, pp. 29525–29537, 2020.

[5] Y.-J. Liu, G. Feng, Y. Sun, S. Qin, and Y.-C. Liang, "Device association for RAN slicing based on hybrid federated deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 15731–15745, Dec. 2020.

[6] NGMN Alliance, "Description of network slicing concept," Frankfurt, Germany, White Paper, Jan. 2016. [Online]. Available: https://www.ngmn.org/publications/description-of-network-slicing-concept.html

[7] *Study on Architecture for Next Generation System*, Standard 3GPP TS 23.799 V2.0.0, Third Generation Partnership Project (3GPP), Dec. 2016.

[8] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement learning-based 5G network slice broker," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1543–1557, Aug. 2019.

[9] M. Jiang, M. Kondoluci, and T. Mahmoodi, "Network slicing management & prioritization in 5G mobile systems," in *Proc. Eur. Wireless Conf.*, May 2016, pp. 1–6.

[10] V. K. Choyi, A. Abdel-Hamid, Y. Shah, S. Ferdi, and A. Brusilovsky, "Network slice selection, assignment and routing within 5G networks," in *Proc. IEEE Conf. Standards Commun. Netw. (CSCN)*, Nov. 2016, pp. 1–7.

[11] A. Okic, L. Zanzi, V. Sciancalepore, A. Redondi, and X. Costa-Pérez, "π-ROAD: A learn-as-you-go framework for on-demand emergency slices in V2X scenarios," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2021, pp. 1–10.

[12] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, and X. Costa-Perez, "Optimising 5G infrastructure markets: The business of network slicing," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2017, pp. 1–9.

[13] 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 Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 4883–4888.

[14] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing in 5G: An auction-based model," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[15] M. Richart, J. Baliosian, J. Serrat, and J.-L. Gorricho, "Resource slicing in virtual wireless networks: A survey," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 3, pp. 462–476, Sep. 2016.

[16] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 127–140.

[17] L. Zanzi, V. Sciancalepore, A. Garcia-Saavedra, H. D. Schotten, and X. Costa-Perez, "LACO: A latency-driven network slicing orchestration in beyond-5G networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 667–682, Jan. 2021.

[18] M. R. Sama, S. Beker, W. Kiess, and S. Thakolsri, "Service-based slice selection function for 5G," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.

[19] Y. Sun, S. Qin, G. Feng, L. Zhang, and M. Imran, "Service provisioning framework for RAN slicing: User admissibility, slice association and bandwidth allocation," *IEEE Trans. Mobile Comput.*, early access, Jun. 8, 2020.

[20] K. Xu, F. Wang, and L. Gu, "Behavior analysis of Internet traffic via bipartite graphs and one-mode projections," *IEEE/ACM Trans. Netw.*, vol. 22, no. 3, pp. 931–942, Jun. 2014.

[21] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proc. Eur. Conf. Mach. Learn.*, 2005, pp. 437–448.

[22] R. Combes, A. Proutiere, D. Yun, J. Ok, and Y. Yi, "Optimal rate sampling in 802.11 systems," in *IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2014, pp. 2760–2767.

[23] R. Kleinberg, A. Slivkins, and E. Upfal, "Multi-armed bandits in metric spaces," in *Proc. 14th Annu. ACM Symp. Theory Comput.*, May 2008, pp. 681–690.

[24] S. Bubeck, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Found. Trends Mach. Learn.*, vol. 5, no. 1, pp. 1–122, 2012.

[25] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Adv. Appl. Math.*, vol. 6, no. 1, pp. 4–22, Mar. 1985.

[26] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multi-armed bandit problem," *Mach. Learn.*, vol. 47, nos. 2–3, pp. 235–256, May 2002.

[27] R. Agrawal, "Sample mean based index policies by O(log n) regret for the multi-armed bandit problem," *Adv. Appl. Probab.*, vol. 27, no. 4, pp. 1054–1078, Dec. 1995.

[28] L. Tran-Thanh, A. C. Chapman, A. Rogers, and N. R. Jennings, "Knapsack based optimal policies for budget-limited multi-armed bandits," in *Proc. 26th (AAAI) Conf. Artif. Intell.*, Jul. 2012, pp. 1134–1140.

[29] J. Komiyama, I. Sato, and H. Nakagawa, "Multi-armed bandit problem with lock-up periods," in *Proc. Asian Conf. Mach. Learn.*, vol. 29, Nov. 2013, pp. 100–115.

[30] *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and Channel Coding*, Standard 3GPP TS 36.212, TThird Generation Partnership Project (3GPP), Dec. 2020.

[31] (2021). *OpenEPC 6*. Accessed: Apr. 2021. [Online]. Available: https://www.open5gcore.org/

[32] NEC. (2021). *NeoFace Facial Recognition*. Accessed: May 2021. [Online]. Available: https://www.nec.com/en/global/solutions/biometrics/face/index.html

[33] Reolink. (2021). *IP HD-Camera RLC-410*. Accessed: May 2021. [Online]. Available: https://reolink.com/product/rlc-410/

[34] Huawei. (2021). *E3372 LTE USB Stick*. Accessed: May 2021. [Online]. Available: https://consumer.huawei.com/en/routers/e3372/

[35] D-Link. (2021). *DWR-921 4G LTE Router*. Accessed: May 2021. [Online]. Available: http://www.dlink.com/de/de/products/dwr-921-4g-lte-router

[36] (2021). *Raspberry Pi 2 Model B*. Accessed: May 2021. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/

[37] NEC. (2021). *LTE small-cell MB4420*. Accessed: May 2021. [Online]. Available: https://de.nec.com/de_DE/en/global/solutions/nsp/sc2/doc/mb4220.pdf

[38] *Technical Specification Group Services and System Aspects; Policy and Charging Control Architecture*, Standard 3GPP TS 23.203, 3rd Generation Partnership Project (3GPP), Jun. 2016.

**Vincenzo Sciancalepore** (Senior Member, IEEE) received the M.Sc. degree in telecommunications engineering, the M.Sc. degree in telematics engineering in 2011 and 2012, respectively, and the double Ph.D. degree in 2015. He is currently a Senior 5G Researcher with the NEC Laboratories Europe, Heidelberg, focusing his activity on network virtualization and network slicing challenges. He is also involved in the IEEE Emerging Technologies Committee leading the initiatives on SDN and NFV. He is the Chair of the Emerging Technologies Initiative (ETI) on Reconfigurable Intelligent Surfaces (RIS). He was a recipient of the National Award for the Best Ph.D. Thesis in the area of communication technologies (Wireless and Networking) issued by GTTI in 2015.

**Lanfranco Zanzi** (Member, IEEE) received the B.Sc. and M.Sc. degrees in telecommunication engineering from the Politecnico of Milan, Italy, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree with the Technical University of Kaiserslautern. He is working as a Research Scientist with the NEC Laboratories Europe. His research interests include network virtualization, machine learning, blockchain, and their applicability to 5G mobile networks in the context of network slicing.

**Xavier Costa-Pérez** (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in telecommunications from the Polytechnic University of Catalonia (UPC), Barcelona. He is currently the Head of the Beyond 5G Networks Research and Development, NEC Laboratories Europe, the Scientific Director with the i2Cat Research and Development Center, and a Research Professor with ICREA. His team contributes to products roadmap evolution and to European commission research and development collaborative projects and received several awards for successful technology transfers. In addition, the team contributes to related standardization bodies, such as 3GPP, ETSI NFV, ETSI MEC, and IETF. He has been a 5GPPP technology board member. He served on the program committee of several conferences, including IEEE Greencom, WCNC, and INFOCOM, published at top research venues, and holds several patents. He was a recipient of the National Award for his Ph.D. Thesis. He also serves as an Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING and IEEE TRANSACTIONS ON COMMUNICATIONS journals.

**Antonio Capone** (Fellow, IEEE) is currently the Dean of the School of Industrial and Information Engineering, Politecnico di Milano (Technical University of Milan). His main research activities include radio resource management in wireless networks, traffic management in software defined networks, network planning, and optimization. He is author of over 250 publications on these topics. He contributes to major international conferences on networking as a technical program committee member. He is an Editor of the IEEE TRANSACTIONS ON MOBILE COMPUTING, *Computer Networks*, and *Computer Communications* and served as an Editor for the ACM/IEEE TRANSACTIONS ON NETWORKING from 2010 to 2014.