ONETS: Online Network Slice Broker
From Theory to Practice

Vincenzo Sciancalepore, *Member, IEEE,* Lanfranco Zanzi, *Student Member, IEEE,*
Xavier Costa-Perez, *Member, IEEE,* and Antonio Capone, *Senior Member, IEEE*

Abstract—Network slicing allows mobile network operators to open their physical network infrastructure platform to the concurrent deployment of multiple logical self-contained networks, i.e., *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.

◆

1 INTRODUCTION

MOBILE networks are a key element of today's society, enabling communication, access and information sharing. However, as cellular networks move from being voice-centric to data-centric, operators revenues are not able to keep pace with the predicted increase in traffic volume. 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*, i.e., slice owners, with very diverse requirements. The availability of this vertical market multiplies the monetization opportunities of the network infrastructure as (i) new players may come into play (e.g., automotive industry, e-health, ...), and (ii) a higher infrastructure capacity utilization can be achieved by admitting network slice requests and exploiting multiplexing gains. With network slicing, different services can be provided by different network slice instances. Each of these instances consists of a set of virtual network functions that run on the same infrastructure with a tailored orchestration.

However, network slicing introduces new challenges that need to be addressed in order to be adopted in practice. A trade-off has to be considered between a *fully shared* mobile network among tenants (with shared functions and resources) and an *isolated slices* one (with dedicated functions and resources only). In this context, a network slice broker solution is desirable, acting as an arbitration entity in charge of satisfying heterogeneous slice requirements from tenants while at the same time guaranteeing the most efficient use of the infrastructure resources. The network slice broker concept has been previously considered in [1]. In this

paper we build on this concept and design an *online* network slice brokering solution that complies with the novel 3GPP Network Slicing architecture development (described in Section 2).

The objective of our *ONETS* solution is to design an efficient online network slice broker that by analyzing past network slicing information maximizes the network slice resources multiplexing gains. 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 a feasible 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 to get an indication of 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, and Public Safety.

2 NETWORK SLICING IN 3GPP

3GPP has defined a novel network architecture for network slicing support. In particular, the 3GPP working group SA2 [2] 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 (i) manage user authentication and registration (AMF), (ii) support multiple connection sessions (SMF) and (iii), instruct different routing policies (PCF). On the other hand, the user plane is unified into a generic function (UPF) managing distinct data networks (DNs) through the next-generation-Radio Access Network (ngRAN). This new architecture allows for an easier network functions virtualization and thus, flexible multi-tenant deployments. RAN nodes (and functions) are virtualized and flexibly chained to provision end-to-end RAN slices with

- *V. Sciancalepore, L. Zanzi, X. Costa-Perez were with NEC Europe Ltd., Kurfrsten-Anlage 36, 69115 Heidelberg.*
  E-mail: {name.surname}@neclab.eu.
- *A. Capone is with Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza L. da Vinci, 32, 20133 Milan, Italy.*
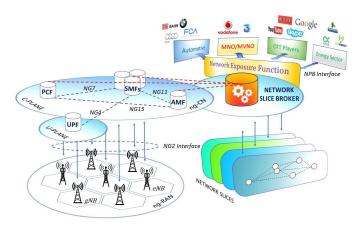 *E-mail: antonio.capone@polimi.it.*

Fig. 1. 3GPP Network Slicing Architecture

a dedicated SMF. Interestingly, AMF (and PCF) can 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. 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.

3 RELATED WORK

Network slicing is currently a very hot topic in the 5G research community given its business relevance [3] and the recent definition by 3GPP of the architecture [4]. Network virtualization though is not a new topic and several solutions supporting virtualization of network resources have been already designed for LTE optimizing network resource utilization and improving QoE.

In [5] the authors propose a wireless network virtualization for LTE systems, which includes a slicing scheme to maximize the resource utilization and physical resource blocks allocation to different Service Providers (SPs). The scheme is dynamic and flexible, allowing to reach arbitrary fairness in the resource allocation among the different SPs. [6] proposes a framework for wireless resource virtualization for LTE networks. In this case, they mostly investigate the customization of scheduling policies in order to fit the service requirements and business model of different Mobile Network Operators (MNOs), providing considerations

on isolation, complexity and resource utilization. [7] introduces a downlink mechanism for allocating network resources. This mechanism decides to accept a new service only if the provisioning does not affect the throughput of the other services in the cell. This work does not consider the dynamic change in terms of QoS experienced by moving UEs, and it does not compensate during resource allocation strategy definition. [8] suggests a new heuristic-based prioritized admission control mechanism. The novelty here is 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. [9] 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.

The authors of [10] come up with a set of requirements to guide the design of the 5G RAN network and to support its development. They first focus on the impact that network slicing could bring to the protocol architecture, i.e., how the split could be defined between functions among the layers of the protocol stack and how different functions may or may not be relevant to different slices. Hence, they investigate the design of network functions (NFs) as common or dedicated resources for a given slice, ending their analysis on the management of shared infrastructure and the management of different slices that may be associated to different business purposes. Similarly, [11] attempts to solve the network slice selection problem by proposing a new architecture and discussing the integration of next generation RRC with NAS protocol. As pointed out in [12] slicing implies that each 5G slice needs to have its own set of allocated resources and this aspect introduces a novelty in the management of network resources in mobile systems. Indeed, in previous generations of mobile networks, the resources to be assigned were mainly radio resources, while in 5G 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 [13], [14]. [15] 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. In the same context, [16] introduces a practical admission control solution for network slicing pursuing at maximizing the overall network revenues following a simple pricing model for RAN slices. A slice selection function is a key element in the future core network architecture. [17] tries 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.

To the best of our knowledge, the work presented in this paper

is the first designing and evaluating an automated online network slice broker with the optimization goal of maximizing network slicing multiplexing gains.

4 SYSTEM DESIGN

We consider a telecom service provider (hereafter Operator) opening its infrastructure to external tenants (vertical industries). The operator owns the infrastructure which has a capacity C[1]. The infrastructure tenant list $\mathcal{I}$ is known a-priori, as each external domain must subscribe beforehand to send 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 network slice characteristics amongst available network slice templates. Such templates are fixed and decided beforehand by the operator [3]. Each network slice $s = \{R^{(s)}; L^{(s)}\}$ comprises an amount of resources, $R^{(s)}$, to be explicitly assigned to users served within the network slice s, and a time duration $L^{(s)}$ expressed in terms of seconds. Such parameters are tailored to particular services and might be modified according to new service requirements. The Network Slice requirements define the Service Level Agreement (SLA) between the operator and the tenant.

4.1 System Model

Each tenant i asks for a network slice template s at time t, $r_i^{(s)}(t)$. Such time instant is obtained from an i.i.d. random variable, namely inter-arrival time Δt, exponentially distributed with rate ϕ_i. Inter-arrival rates, ϕ_i, are drawn from a Pareto distribution with mean ρ and standard deviation ζ, 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 slice request, i.e., multiple network slices assigned to the same tenant cannot overlap in time, $t \geq t_{-1} + L^{(s)}$, if the slice template s has been granted in the previous request $r_i^{(s)}(t_{-1})$[3].

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.*

4.2 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 [19], [20], [21]. 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) [22]. 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.

4.3 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.

---

1. We consider an uniform network slicing solution across the whole network, as currently considered by 3GPP [4]

2. Note that this assumption relies on the traffic flow behavior, as suggested in [18]. However, as shown in Section 5.1.3, it might be relaxed to bring interesting findings.

3. This assumption makes tractable the analysis in Section 4.3. However, it can be easily extended by assuming tenants asking for multiple slices at the same time as distinct virtual tenants.
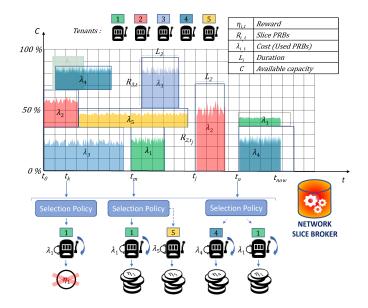
Fig. 2. BLMAB problem example applied to an online network slicing brokering process.

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 counterintuitive, it provides the effective means for deeply learning the tenant behaviors (by considering the frequency of network slice requests and the real slice utilization average) 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.

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

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

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. α 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.

Problem ONLINE-SLICING:

$$\begin{aligned}
\text{maximize} \quad & \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{A}_t} \eta_{i,t} \\
\text{subject to} \quad & \sum_{i \in \mathcal{A}_t} \lambda_{i,t} \leq C, \quad \forall t \in \mathcal{T}; \qquad \text{:budget} \\
& a_{i,t} \in \mathcal{A}_t \subseteq \{0,1\}^{|\mathcal{I}|}, \quad \forall t \in \mathcal{T}; \qquad \text{:multi-plays} \\
& a_{i,t} \geq a_{i,t_{-1}} \mathbb{1}(t - t_i^{\text{START}} \leq L_i^{(s)}); \quad \text{:lock-up}
\end{aligned}$$

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. With some abuse of notation, we denote t_i^{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., $\sum_{i \in \mathcal{I}} a_{i,t} \leq 1, \forall t \in \mathcal{T}$. 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. $\qquad \square$

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 5.

Proposition 1. *The BLMAB problem is more complex than the stochastic multi-armed bandit problem. Therefore, a lower bound of the stochastic MAB is also a lower bound of Problem* ONLINE-SLICING, *considering the same number of rounds.*

---

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

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.

4.4 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 [23]. 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 density function $f(x, \theta_i)$, where all $\theta_i \in \Theta$ are unknown parameters. Every time arm i is pulled, it returns a reward υ_i drawn from $f(x, \theta_i)$ such that $\mu(\theta_i)$ is the mean of υ_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 σ is a permutation vector of $\mathcal{I}$ following a reward decreasing order. Mathematically, it holds that the regret is obtained as

$$
\begin{aligned}
R_T^{\pi}(\Theta) \quad &= T \sum_{i=1}^{|\mathcal{K}|} \mu(\theta_{\sigma(i)}) - \mathbb{E}_{\pi}[\sum_{t=1}^{T} \upsilon_{\pi(t)}(t)] \\
&= T \sum_{i=1}^{|\mathcal{K}|} \mu(\theta_{\sigma(i)}) - \sum_{i=1}^{\mathcal{I}} \mu(\theta_i)\mathbb{E}[W_i(T)];
\end{aligned}
\tag{3}
$$

where $W_i(T)$ is the number of rounds within T arm i is pulled down. If the selection policy is *uniformly good*, then $R_T^{\pi}(\Theta) = o(T^a), \forall a > 0$, and, in turn, it holds that

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

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

$$
\lim_{T \to \infty} \inf \frac{R_T^{\pi}(\Theta)}{logT} \geq \sum_{i:\ \mu(\theta_i) < \mu(\theta_{\sigma(|\mathcal{K}|)})} \frac{\mu(\theta_{\sigma(|\mathcal{K}|)}) - \mu(\theta_i)}{H(\theta_i, \theta_{\sigma(|\mathcal{K}|)})},
\tag{5}
$$

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 θ_u and θ_v, respectively.

5 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 $O(logT)$ as expected cumulative regret. When important variations are considered, the overall complexity might be perturbed by additional factors, e.g., the lock-up time periods.

Algorithm 1 eUCB: Selection algorithm π to select the next batch of arms to pull down while guaranteeing a fixed budget.

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

5.1 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 4.4), which might optimally bound the performance. We consider the classical Upper Confidence Bound (UCB) solution, enhanced to address our BLMAB problem.

5.1.1 Enhanced-UCB (eUCB)

Several works address the UCB solution considering different variations. We focus on the classical one proposed in [23] 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_{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.

Our problem introduces additional features, which must be carefully taken into account. In particular, multiple arms can be selected while guaranteeing a maximum budget C. Also, if an arm has been selected in the previous rounds and its lock-up window is still running, the algorithm must select again such an arm for the next round. An enhanced version of UCB (*eUCB*) algorithm is described in Alg. 1. Lines $1 - 5$ require that all arms are pulled once at the beginning. 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 $\upsilon_i \sim f(x, \theta_i)$. Line 8 optimally solves an instantaneous version of the problem, defined 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

Algorithm 2 ONETS: Selection algorithm π to select K arms to pull down while guaranteeing a fixed budget.

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

---

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.

5.1.2 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 [24]. 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 6).

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).

5.1.3 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 $\hat{\theta}_i = \bar{\theta}_i + G_{\text{ALG}}$ $\left(\text{in case of } eUCB, G_{\text{ALG}} = \sqrt{\frac{2 \log T}{W_i(T)}} \right)$. We can calculate the ex-

pected number of times arm i is pulled down based on *ONETS* algorithm as follows.

$$
\mathbb{E}[W_i(T)] \leq
$$
$$
\int_0^\infty f(x, \hat{\theta}_i) \Pr(x \geq \max_{p \neq i} \upsilon_p) \mathrm{d}x +
$$
$$
\int_0^\infty f(x, \hat{\theta}_i) \Pr_{\forall j \in \mathcal{I}}(\text{single } \upsilon_j \geq x) \Pr(x \geq \max_{p \neq \{i,j\}} \upsilon_p) \mathrm{d}x +
$$
$$
\int_0^\infty f(x, \hat{\theta}_i) \Pr_{\forall j \in \mathcal{I}}(\text{multiple } \upsilon_j \geq x) \Pr(x \geq \max_{p \neq i} \upsilon_p) \mathrm{d}x \leq
$$
$$
\sum_{|\mathcal{H}|=0}^{|\mathcal{K}|} \sum_{\sigma=1}^{\binom{|\mathcal{I}|}{|\mathcal{H}|}} \int_0^\infty f(x, \hat{\theta}_i) \left(\prod_{\substack{j \in \mathcal{H}(\sigma)}} \int_x^\infty f(y, \hat{\theta}_j) \mathrm{d}y \prod_{\substack{k \in \mathcal{I}, \\ \mathcal{I} \backslash \{i\} \\ \mathcal{I} \backslash \mathcal{H}(\sigma)}} \int_0^x f(y, \hat{\theta}_k) \mathrm{d}y \right) \mathrm{d}x.
$$
$$
(6)
$$

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

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

where σ 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 $L_i << T$ 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 $L^* = \max\limits_{i \in \mathcal{I}} L_i$, as also confirmed in [25].

5.2 ϵ-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 ϵ value. A linear dependency of ϵ 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 6. The algorithm (listed in Alg. 3) will select the best arms maximizing the reward (line 9) with probability ϵ, whereas it will select arms randomly (line 11) satisfying (in both cases) the limited budget constraint (lines $14 - 17$). Please note that ϵ-greedy algorithm does not require an initial training phase.

Algorithm 3 ϵ-**greedy:** Selection algorithm π to select the next batch of arms based on ϵ-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 υ_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**

Regret Upper Bound for ϵ-greedy

Greedy solutions are proved to have a sub-linear regret. In particular, as shown in [26] the upper bound regret for $T \rightarrow \infty$ 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:

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

where i^* represents the optimal arm.

6 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 4.3. Network slice requests are generated following exponential distributions, as explained in Section 4, with given ρ, ζ parameters. 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% confidence degree. The default system parameters are listed in Table 1.
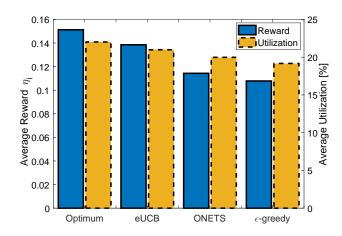

Fig. 3. Performance evaluation of BLMAB Heuristics versus the *Optimum* and *eUCB* solutions.

TABLE 1
Simulation parameters

System Parameters	Values
$|\mathcal{I}|$	10 tenants
$|\mathcal{S}|$	10 slice templates
Capacity (C)	150 RBs (15 Mhz) [27]
$\rho; \zeta$	100; 0.1
α	0.5
Time horizon (T)	10000 rounds
ϵ-greedy ($b; d$)	10; 0.01
K (ONETS)	6

6.1 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 2 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.

TABLE 2
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	-	-
ϵ-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**

(a) Cumulative Rewards Distribution.

(b) Tenant Selection Ratio
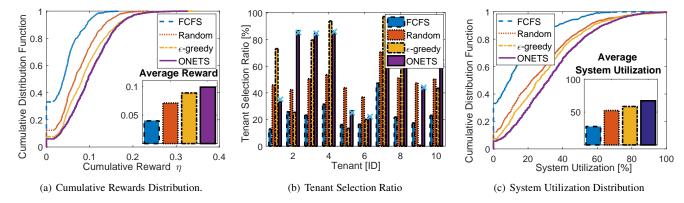
(c) System Utilization Distribution

Fig. 4. Performance evaluation of different BLMAB selection policies for a 10 tenants scenario.

6.2 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 Δt of slice requests per tenant are exponentially distributed with rate ϕ_i (see Section 4). 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 5. 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 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 while increasing the 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 (α)

In this section, we study our reward solution expressed in Eq. (1) in order to illustrate the impact of the configurable parameter α. In
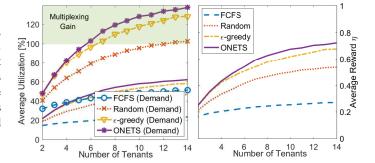

Fig. 5. Performance evaluation of different BLMAB selection policies when increasing the number of tenants.

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 η_i, which validates our reward design: the larger the reward, the higher the multiplexing gains.

However, a proper tuning of the weight α might strongly influence and lead the system toward near-optimal steady states. When α 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* is the best scheme efficiently translating 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 degradations 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 ($< 0.015\%$).
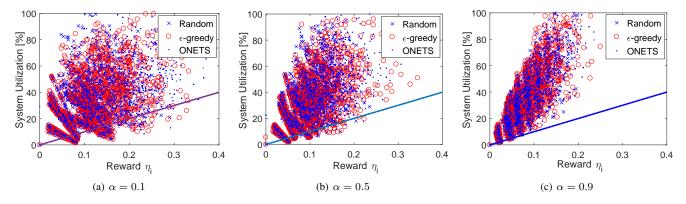
Fig. 6. Scatterplot of Reward versus System Utilization for different α configurations
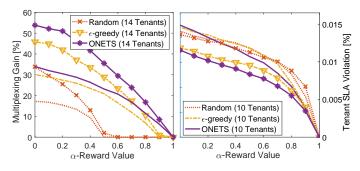

Fig. 7. Performance evaluation of different BLMAB selection policies by varying the α configuration.

This trade-off is optimally driven by a proper tuning of the reward $\eta_{i,t}$, expressed in Eq. (1)[5]. Specifically, different α 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 α 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.

7 PROOF-OF-CONCEPT

7.1 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 3.

The OpenEPC software [28] 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.

5. 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.

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 [34], they use 15MHz bandwidth, i.e., 150 PRBs per 1ms subframe (cfr. [27]). 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.
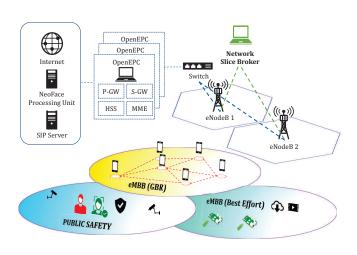
To support the network slicing concept, we build on RAN equipment supporting RAN Sharing: This enables us to use the same RAN infrastructure for different Public Land Mobile Networks (PLMNs). We apply the Multiple Operator Radio Access Network (MORAN) approach to have dedicated network core domains sharing 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).
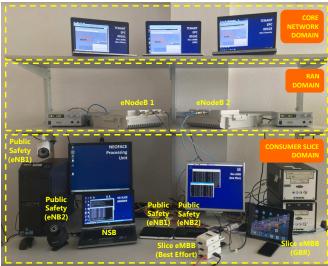
We assume three different tenants registered for issuing network slice requests with different traffic characteristics: (i) enhanced Mobile BroadBand (eMBB BE) slice generating FTP file transfers best-effort traffic, (ii) eMBB GBR generating multiple Voice over LTE (VoLTE) traffic streams emulating an audio conference system, (iii) a Public Safety slice for video surveillance.

Raspberry Pis [33] with LTE USB dongles [31] are used to generate eMBB BE traffic. Tenants might ask for a new network

TABLE 3
PoC Hardware Specifications

Equipment	Description	Ref.
Virtualized EPC	OpenEPC Rel. 6	[28]
Face Recognition SW	NeoFace Facial Recognition	[29]
Surveillance Camera	HD, Motion Detection	[30]
UE	LTE USB Stick	[31]
Smart phones	iPhone (SE) and LG Nexus (5)	-
Router LTE	LTE with SIM-card slot	[32]
Smart devices	Raspberry Pi 2	[33]
LTE Small cell	15 MHz channel, LTE Band 3	[34]
	DL 1775 MHz, UL 1870 MHz (FDD)	

(a) Testbed blocks overview

(b) Real deployment

Fig. 8. Network Slicing Proof-of-Concept with 3 Slices: eMMB (BE),Public Safety and eMBB(GBR).

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 [32], which are, in turn, connected to our eNBs. The cameras [30] are provided with motion detection features. This introduces a bursty traffic source for our experiments. In addition, an advanced face recognition software [29] 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 target, a red 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.

7.2 PoC Performance evaluation

We evaluate our *ONETS* online network slicing solution in the proof-of-concept setup previously described. Our goal is to an-

Fig. 9. Sample of video surveillance face detection: successful (left side) and failed (right side).

alyze 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
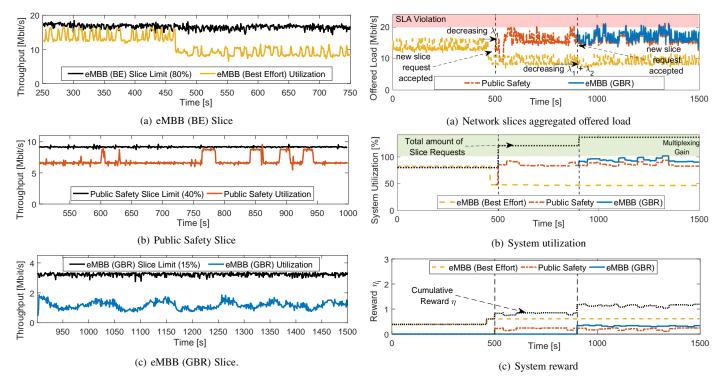
(a) eMBB (BE) Slice

(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.

is fully using its resource allocation. During this period, 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 (λ_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 (λ_1, λ_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 illustrative 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.

8 CONCLUSIONS

One of the key novel concepts of 5G networks is Network Slicing, 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 accomodation 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, ϵ-greedy) for the considered scenarios, (iv) its computational complexity is in the same order of magnitude of a simple greedy solution (ϵ-greedy), (v) *ONETS* aggressivity for achieving network slicing multiplexing gains is a configurable parameter (α) that can be freely tuned by operators according to proprietary policies and desired SLA protection levels.

REFERENCES

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

[2] Third Generation Partnership Project (3GPP), "System Architecture for the 5G System," 3GPP TS 23.501 v0.3.1, Mar. 2017.

[3] N. Alliance, "Description of network slicing concept," *NGMN 5G P1*, Jan. 2016.

[4] Third Generation Partnership Project (3GPP), "Study on architecture for next generation system," 3GPP TS 23.799 V2.0.0, Dec. 2016.

[5] M. I. Kamel, L. B. Le, and A. Girard, "LTE wireless network virtualization: Dynamic slicing via flexible scheduling," in *Vehicular Technology Conference (VTC Fall)*. IEEE, Sep. 2014, pp. 1–5.

[6] M. Kalil, A. Shami, and Y. Ye, "Wireless resources virtualization in LTE systems," in *IEEE Infocom - Conference on Computer Communications Workshops*, Apr. 2014, pp. 363–368.

[7] S. B. Rejeb, N. Nasser, and S. Tabbane, "Admission control strategies and QoS evaluation based on mobility in LTE," in *Globecom Workshops*. IEEE, Dec. 2013, pp. 4883–4888.

[8] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing management & prioritization in 5G mobile systems," in *22th European Wireless Conference*, May 2016, pp. 1–6.

[9] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, A. Banchs, "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in *in Proceedings of IEEE INFOCOM*. IEEE, 2017, pp. 4883–4888.

[10] I. da Silva, G. Mildh, A. Kaloxylos, P. Spapis, E. Buracchini, A. Trogolo, G. Zimmermann, and N. Bayer, "Impact of network slicing on 5G radio access networks," in *IEEE European Conference on Networks and Communications (EuCNC)*, Jun. 2016, pp. 153–157.

[11] T. Yoo, "Network slicing architecture for 5G network," in *IEEE International Conference on Information and Communication Technology Convergence (ICTC)*, Oct. 2016, pp. 1010–1014.

[12] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing in 5G: an auction-based model," in *IEEE International Conference on Communications*, 2017.

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

[14] X. Costa-Perez, A. Garcia-Saavedra, and X. Li et al., "5G-Crosshaul: An SDN/NFV Integrated Fronthaul/Backhaul Transport Network Architecture," *IEEE Wireless Communications*, vol. 24, no. 1, pp. 38–45, Feb. 2017.

[15] V. K. Choyi, A. Abdel-Hamid, Y. Shah, S. Ferdi, and A. Brusilovsky, "Network slice selection, assignment and routing within 5G networks," in *Standards for Communications and Networking (CSCN)*. IEEE, Oct. 2016, pp. 1–7.

[16] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, X. Costa-Perez, "Optimising 5G infrastructure markets: The business of network slicing," in *in Proceedings of IEEE INFOCOM*. IEEE, 2017.

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

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

[19] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proceedings of the 16th European Conference on Machine Learning*, 2005, pp. 437–448.

[20] R. Combes, A. Proutiere, D. Yun, J. Ok, and Y. Yi, "Optimal rate sampling in 802.11 systems," in *IEEE INFOCOM 2014 - Conference on Computer Communications*, Apr. 2014, pp. 2760–2767.

[21] R. Kleinberg, A. Slivkins, and E. Upfal, "Multi-armed bandits in metric spaces," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, 2008, pp. 681–690.

[22] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," Foundations and Trends in Machine Learning, 2012.

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

[24] L. Tran-Thanh, A. C. Chapman, A. Rogers, and N. R. Jennings, "Knapsack based optimal policies for budget-limited multi-armed bandits," in *Proceedings of the Twenty-Sixth (AAAI) Conference on Artificial Intelligence*, 2012.

[25] J. Komiyama, I. Sato, and H. Nakagawa, "Multi-armed bandit problem with lock-up periods," *JMLR Workshop and Conference Proceedings: Asian Conference on Machine Learning*, vol. 29, pp. 100–115, Nov. 2013.

[26] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, May 2002.

[27] Third Generation Partnership Project (3GPP), "Technical Specification Group Services and System Aspects; Policy and charging control architecture," 3GPP TS 23.203, Jun. 2016.

[28] OpenEPC 6. http://www.openepc.com/.

[29] NEC NeoFace Facial Recognition. http://au.nec.com/en_AU/solutions/security-and-public-safety/biometrics/neoface-facial-recognition-overview.html.

[30] IP HD-Camera Reolink RLC-410. https://reolink.com/product/rlc-410/.

[31] Huawei E3372 LTE USB Stick. http://consumer.huawei.com/en/mobile-broadband/dongles/tech-specs/e3372.htm.

[32] D-Link DWR-921 4G LTE Router. http://www.dlink.com/de/de/products/dwr-921-4g-lte-router.

[33] Raspberry Pi 2 Model B. https://www.raspberrypi.org/products/raspberry-pi-2-model-b/.

[34] NEC LTE small-cell MB4420. http://www.nec.com/en/global/solutions/nsp/sc2/prod/e-nodeb.html.