Mobile Traffic Forecasting for Maximizing 5G Network Slicing Resource Utilization

Vincenzo Sciancalepore*, Konstantinos Samdanis†, Xavier Costa-Perez*,
Dario Bega‡§, Marco Gramaglia‡§, Albert Banchs‡§
* NEC Europe Ltd. † Huawei Europe ‡ IMDEA Networks Institute § Universidad Carlos III de Madrid

***Abstract*—The emerging network slicing paradigm for 5G provides new business opportunities by enabling multi-tenancy support. At the same time, new technical challenges are introduced, as novel resource allocation algorithms are required accommodating different business models. In particular, infrastructure providers need to implement radically new admission control policies to decide on network slices requests according to their Service Level Agreements (SLA). When implementing such admission control policies, infrastructure providers may apply forecasting techniques in order to adjust the allocated slice resources so as to significantly improve the network utilization while meeting network slices' SLAs.**

This paper focuses on the design of three key network slicing building blocks in charge of (i) traffic analysis and prediction per network slice, (ii) admission control decisions for network slice requests, and (iii) adaptive correction of the forecasting solution based on measured deviations. Our results illustrate the very substantial potential gains in terms of system utilization as well as the trade-off between conservative forecasting configurations versus more aggressive ones (higher gains, SLA risk).

I. Introduction

In addition to the clear advantages in terms of, among others, enhanced bandwidth, reduced latency or extended coverage, the introduction of future 5G networks will have a significant impact on how operators manage their infrastructure. In contrast to the relatively monolithic architectures of 3G and 4G, by building on the recent advances in network *softwarization*, 5G networks will be highly modular and designed to be *future-proof*.

5G networks will hence allow higher flexibility: network virtualization can boost the introduction of very diverse services to be deployed on-demand using shared infrastructure. This feature enables new business opportunities for Mobile Network Operators (MNO); indeed, hosting different services with possibly conflicting requirements on the same infrastructure is currently not achievable with the current *one-size-fits-all* architectures. However, it also introduces new critical challenges. To this end, the network slicing concept [1] is expected to be one of the technical solutions to these challenges.

Network slicing allows MNOs to open their physical network infrastructure platform to the concurrent deployment of multiple logical self-contained networks, orchestrated in different ways according to their specific service requirements; such network slices are then (temporarily) owned by tenants. 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. However, the technical enablers for network slicing admission control need to be investigated.

This work has been partially funded by the European Union's Horizon 2020 research and innovation programme under the grant agreement No. 671584 5GNORMA.

The 5G Network Slice Broker [2] is a novel network element that builds on the capacity broker functional block considered by 3GPP for advanced RAN sharing [3]. It maps incoming Service Level Agreement (SLA) requirements associated to network slice requests into physical resources. Tenants hence obtain a "slice" of the appropriate Radio Access Network (RAN) elements. The architectural specifications for this new network paradigm are currently under definition and the necessary algorithms yet to be devised.

Although very conservative mappings may be considered for mission critical services that need ultra-high availability, enhanced admission control algorithms that leverage multiplexing gains of traffic among slices are the key to the optimization of network utilization and monetization. To this end, the ability to predict the actual *footprint* of a particular network slice is essential to increase the maximum number of slices that might be run on the same infrastructure.

Building on this idea, in this paper we design three key network slicing building blocks: (i) a *forecasting* module predicting network slices traffic based on past traffic and user mobility, (ii) a network slicing *admission control* algorithm and (iii) a network slicing *scheduler* algorithm in charge of meeting the agreed SLAs and report deviations to the forecasting module.

The remaining of the paper is organized as follows. In Section II we review the state-of-the-art solutions before presenting our framework building blocks in Section III. In Section IV we give the premises of our forecasting model, whereas in Section V we formulate the admission control problem as a geometric knapsack, providing also its NP-Hardness proof. In Section VI we explain the scheduling process and how its feedback is used to adjust the forecasting process. In Section VII we discuss the simulation results and, finally, we conclude the paper in Section VIII.

II. Related Work

The support of multi-tenancy in 3GPP LTE networks was originated from active RAN sharing, which facilitated network sharing based on contractual agreements. A study on virtualization for wireless and mobile networks considering preliminary proposals such as the GENI project as well as early LTE base station virtualization is elaborated in [4]. Two

active network sharing architectures are specified in 3GPP, the Multi-Operator Core Network (MOCN) allowing each operator to share eNBs connected on a separate core network and the Gateway Core Network (GWCN) where operators share additionally the Mobility Management Entity (MME) [5]. A complementary network sharing management, which enables MVNOs to control the allocated resources is defined in [6]. Our proposal exploits the experience of early deployments, while being compatible with the 3GPP specifications.

A RAN sharing solution applying proportional fairness criterion is depicted in [7]. For sharing resources among different operators while considering radio conditions [8] introduced the Network Virtualization Substrate (NVS), a two-step process that allows the infrastructure provider to control the resource allocation towards each virtual instance of an eNB before each tenant customizes scheduling within the allocated resources [9]. In our work we adopt a similar two-step process allocating slices via a broker entity that provides admission control based on the requested SLAs.

We build on the concept of a signaling-based network slicing broker solution by elaborating the capacity forecasting problem considering guaranteed and best-effort traffic in addition to user mobility. A study that explores different options of network sharing based on a centralized broker is detailed in [10] considering mobility means, spectrum transfer policies and resource virtualization enhancing MNO's limited resources. Such a study unlike our proposal introduces new 3GPP interfaces to accommodate the broker functionality. A scheme that integrates the capacity broker with a minimum set of enhancements on the 3GPP architecture is documented in [11]. Such capacity broker forecasts the network capacity allocating guaranteed and best-effort slices, considering the desired SLA. This study enhances available solutions by introducing algorithms that dynamically evaluate network slices SLA requests, while maximizing the infrastructure resources utilization.

III. System Design

This paper builds on the concept of a 5G network slice broker in the context of the 3GPP network sharing management architecture [6] for establishing network slices through signaling. The 5G network slice broker is introduced at the network management system of the infrastructure provider being able to exploit 3GPP conventional monitoring procedures for gathering global network load measurements. Such information can assist the forecasting process, facilitating admission control and considering the specified network slice SLAs. To support a signaling-based slice allocation certain 3GPP interfaces need to be enhanced (Type 5 and Itf-N) for network slice instantiation and configuration while specifying the time duration, the required resource amount, and additional requirements, i.e., the traffic SLA. We refer the reader to [2] for further architectural details.

Fig. 1 depicts the 5G Network Slice Broker building blocks addressed in this paper. Network slice requests are collected within a fixed negotiation time window. When the time window is closed, network slice requests are processed and

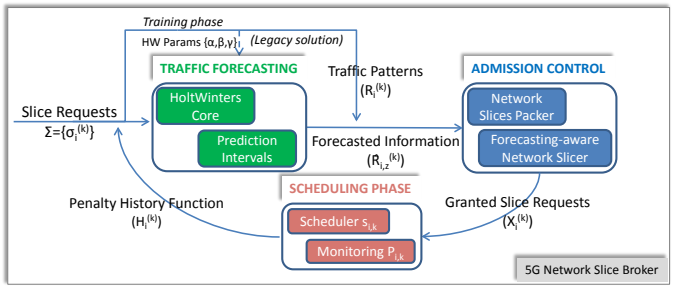


Fig. 1: Block diagram of the 5G Network Slice Broker.

evaluated. A key aspect for an efficient network slice admission control mechanism is to accurately predict the tenants' traffic near-future evolution. This is achieved through a *Traffic Forecasting* phase in charge of analyzing the network slices traffic patterns and providing forecasting information to the *Admission Control* module, as explained in Section IV. When no forecasting solution is applied (w/o forecasting) or during the *training period* (for adjusting the forecasting algorithm parameters), network slice SLA request information is used. Based on such information, Admission Control policies are applied in order to select the network slice requests granted for the next time window. Two different algorithms (with different performance and complexity features) are devised based on the traffic information provided as input, as explained in Section V. A list of granted slice requests is sent to the *Scheduling* phase that allocates network slice physical resources and monitors with a penalty history function the served traffic levels and potential SLA violations. Such a function is used to provide feedback to the forecasting module and to adaptively adjust the system, as explained in Section VI.

IV. Traffic Forecasting

Forecasted traffic pattern information is used to derive actual traffic shapes and maximize the system resource utilization. A key property lies on the accuracy of the forecasting algorithm: the more accurate, the more flexibility in leasing available resources, the less the probability of violating traffic SLAs. While the first aspect is deeply assessed hereafter, we refer the reader to Section VI for more details on SLA violations and dynamic forecasting parameters adjustments.

A. Tenant traffic analysis: characterization and forecasting

Traffic predictions are computed on an aggregate basis for every tenant. Each tenant i might ask for a different network slice request tailored for specific service requirements. Indeed, the forecasting process can easily categorize the traffic requests based on related service requirements, thereby performing a prediction separately per slice. In our analysis, we first assume traffic requests uniformly distributed within the whole network. However, in Section IV-B we extend this assumption for multi-cellular environments where tenant traffic requests are significantly affected by the user mobility.

We assume different classes of traffic based on specific SLAs, as shown in Table I. We denote the traffic volumes of

tenant i for traffic class k, e.g., satisfying particular service requirements, as a realization of a point process, $\zeta_i^{(k)} = \sum_{t=0}^{T} \delta_t \, r_i^{(k)}(t)$, where δ_t denotes the Dirac measure for sample t. We express traffic requests $r_i^{(k)}(t)$ in terms of required resources but they can be easily translated into different metrics, such as latency or throughput demands. Given the periodic nature of traffic requests, the traffic forecasting is based on an observed time window, namely T_{OBS} as defined by the vector $\mathbf{r_i^{(k)}} = (r_i^{(k)}(t - T_{\text{OBS}}), r_i^{(k)}(t - (T_{\text{OBS}} + 1)), \cdots, r_i^{(k)}(t))$. Thus, the forecasting function f_{HW} provides forecasted traffic volumes for time period $[t + 1, t + T_{\text{WINDOW}}]$ defined as $\hat{\mathbf{r}}_\mathbf{i}^{(\mathbf{k})} = (\hat{r}_i^{(k)}(t+1), \hat{r}_i^{(k)}(t+2), \cdots, \hat{r}_i^{(k)}(t+T_{\text{WINDOW}}))$. This periodicity translates into seasons of length W_S, which is repeated based on fixed traffic patterns. Within a single season we assume that process $\zeta_i^{(k)}$ is stationary and ergodic. Therefore, we use the Holt-Winters (HW) forecasting procedure to analyze and predict future traffic requests associated to a particular network slice. Without loss of generality, we denote a specific predicted traffic request $\hat{r}_i^{(k)}(t)$ as $\hat{r}_{i,t}^{(k)}$ and, based on the level l_t, trend b_t and seasonal s_t factors, we obtain the following traffic prediction

$$\begin{aligned} l_t &= \alpha(r_{i,t}^{(k)} - s_{t-W}) + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta(L_t - l_{t-1}) + (1-\beta) b_{t-1} \\ s_t &= \gamma(r_{i,t}^{(k)} - l_{t-1} - b_{t-1}) + (1-\gamma) s_{t-W} \\ \hat{r}_{i,t+T_{\text{WINDOW}}}^{(k)} &= l_t + b_t h + s_{t+T_{\text{WINDOW}}-W}. \end{aligned} \tag{1}$$

We rely on the additive version of the HW forecasting problem as the seasonal effect does not depend on the mean traffic level of the observed time window but instead it is added considering values predicted through level and trend effects.

While the set of optimal HW parameters α, β and γ can be obtained during a training period through existing techniques [12], we focus on the forecasting errors and how the forecasting inaccuracy may affect our network slicing solution. We define the one-step training forecasting error $e_{i,t}^{(k)}$ as follows

$$e_{i,t}^{(k)} = r_{i,t}^{(k)} - \hat{r}_{i,t}^{(k)} = r_{i,t}^{(k)} - (l_{t-1} + b_{t-1} + s_{t-1}), \tag{2}$$

which can be obtained during the training period of our forecasting algorithm, i.e., when predicted values are compared with the observed ones. Given that our process $\zeta_i^{(k)}$ is ergodic, assuming an optimal HW parameter set, for any predicted value at time z we can derive the prediction interval $\left[\hat{ll}_{i,z}^{(k,\chi)}, \hat{hh}_{i,z}^{(k,\chi)}\right]$ wherein future traffic requests lie for that particular network slice with a certain probability $\chi_i^{(k)}$. Analytically, it holds that

$$Pr\left\{\hat{ll}_{i,z}^{(k,\chi)} \leq \hat{r}_{i,z}^{(k)} \leq \hat{hh}_{i,z}^{(k,\chi)}\right\} = \chi_i^{(k)}, \forall z \in [t{+}1, t{+}T_{\text{WINDOW}}] \tag{3}$$

where $\hat{hh}_{i,z}^{(k,\chi)}$ (or $\hat{ll}_{i,z}^{(k,\chi)}$) $= \hat{r}_{i,z}^{(k)} + (-)\Omega_\chi \sqrt{Var(e_{i,z}^{(k)})}$ and

$$Var(e_{i,z}^{(k)}) \approx \left((1 + (z-1)\alpha^2[1 + z\beta + \frac{z(2z-1)}{6}\beta^2]\right)\sigma_e^2.$$

While Ω_χ denotes the one-tailed value of a standard normal distribution such that we obtain $\chi_i^{(k)}$ probability, σ_e^2 is the variance of one-step training forecasting error, i.e., $\sigma_e^2 = Var(e_{i,t}^{(k)})$, over the observed time window. Due to the penalties imposed by traffic SLAs, we focus only on the upper bound of the prediction interval as it provides the "worst-case" of a forecasted traffic level. From Eq. (3), a larger prediction time window T_{WINDOW}, e.g., a higher number of predicted values z, leads to a less accuracy and behaves closer to the real network slice demand (limited network slice resources utilization). Conversely, an accurate forecasting with a lower error probability can result in severe penalties for not having guaranteed the traffic SLAs. Therefore, we adjust the forecasting error probability $\chi_i^{(k)}$ according to the service requirements and to the number of prediction points the forecasting process needs to perform. For instance, best-effort traffic requests having no stringent requirements can tolerate a prediction with a longer time pace resulting in imprecise values. This makes the upper bound $\hat{hh}_{i,z}^{(k,\chi)}$ very close to the real (future) values $r_{i,z}^{(k,\chi)}$ regardless the error probability $\chi_i^{(k)}$ as the number of z values to predict is limited. Hence, we might select for this service type a low forecasting error probability $\chi_i^{(k)}$. On the other hand, when guaranteed bit rate traffic is considered, the corresponding SLA must be fulfilled in a shorter time basis, which makes our forecasting process much more complex requiring significantly more predicted values z. Therefore, our system models such a type of traffic with a higher forecasting error probability $\chi_i^{(k)}$.
Mathematically, traffic class $k = 0$ provides a forecasted horizon longer than the other traffic classes, i.e., a higher number of values z must be predicted. Hence, we can derive an upper bound for the forecasting probability error per tenant i. We calculate the maximum potential gain between the slice request and the forecasted traffic requests as $\hat{d}_i^{(k)} = \max_{z \in \text{T}_{\text{WINDOW}}} \left(R_i^{(k)} - \hat{r}_{i,z}^{(k)}\right)$. We then compute the forecasting error probability as the following

$$\chi_i^{(k=0)} : \Omega_\chi \sqrt{Var(e_{i,z}^{(k=0)})} = \hat{d}_i^{(k=0)}. \tag{4}$$

As soon as the potential gain $\hat{d}_i^{(k=0)}$ becomes very large, we cap the one-tailed value Ω_χ to 3.49 resulting in $\chi_i^{(k=0)} = 99.9\%$. Conversely, we compute the forecasting error probability $\chi_i^{(k=|\mathcal{K}|)} = 50\%$ for the best-effort traffic, which is supposed to have relaxed traffic requirements. Intermediate forecasting error probabilities $\chi_i^{(k)}$ for the other traffic classes k are calculated from (4) by linearly deriving $\hat{d}_i^{(k)}$ values from the upper and the lower bound values. However, forecasting error probability values are dynamically assessed and adjusted based on the SLA violations experienced during the scheduling process, as explained in detail in Section VI-B.

B. User mobility and traffic model periodicity

We extend our forecasting model for dynamic scenarios where user mobility is considered and the traffic periodicity

TABLE I: Network slice traffic requirements [13]

k	$T^{(k)}$	Type and QCI
0	10 ms	GBR - 65
1	50 ms	GBR - 3
2	100 ms	GBR - 1
3	150 ms	GBR - 2
4	300 ms	non-GBR - 6
5	1000 ms	non-GBR - 9

assumption no longer holds. We assume a multi-cellular environment covering the whole area. For that reason, we use human-based mobility patterns for advancing the forecasting process accuracy. We consider the well-accepted SLAW mobility model [14] for user motions. In particular, users move among a number of waypoints, which are distributed over the network area according to self-similarity rules forming a given number of clusters. Clusters with more waypoints are considered as hotspots attracting more users. While performing a flight (a movement from one waypoint to the other within the same trip), based on the gravity probability, users choose a set of clusters which are dynamically and randomly replaced during the flight. Then, users start moving between a subset of waypoints residing within the selected clusters according to a least-action trip planning (LATP) with $\alpha_{\text{SLAW}} = 3$. Traffic requests come randomly during the user trip. Assuming that users stop when reaching a waypoint for a pause-time, we can model the value of the flight-time (x_L) and pause-time (x_P) as a random value drawn from a heavy-tailed distribution function defined in terms of Fourier transformations as

$$f_L(x) = f_P(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-iu\,x - |\rho u|^{\alpha_{\text{DISTR}}}} \mathrm{d}u \qquad (5)$$

where ρ is the scale factor and α^{DISTR} depends on the distribution considered (pause-time or flight-time). Given a uniform user speed distribution, the traffic model of considered users is dominated by a heavy-tailed distribution, whose components, as showed in Eq. (5), can be decoupled to obtain again a periodic behaviour. Without loss of generality, assuming a period M and a generic traffic vector $\mathbf{r_i^{(k)}} = \{r_t\}$, the forecasting process applies a Discrete Fourier Transform (DFT) to retrieve the M-periodic samples $R_w = \sum_{n=0}^{M-1} r_t e^{-iw\frac{2\pi}{N}t}$, where $w = 0, \cdots, M-1$. Please note that R_w is a complex number translating the sinusoidal component of r_t. Thus, the forecasting process can obtain all single time-series components derived by each of those frequency samples by applying the Inverse Discrete Fourier Transform (IDFT), e.g., $r_n = \frac{1}{N}\sum_{w=0}^{M-1} R_w e^{\frac{2\pi i}{N}wn}$, where $n = 0, \cdots, N-1$, finally resulting in a periodic traffic vector $\mathbf{r_i^{(k)}} = (r_i^{(k)}(n), r_i^{(k)}(n+1), \cdots, r_i^{(k)}(n+M))$.

V. Admission Control: Design and Validation

A 5G Network Slice Broker might decide on the network slice requests to be granted for the subsequent time window T_{WINDOW} based solely on the current resource availability. However, if forecasting information is considered, network slice requests might be accurately reshaped (Fig. 2) to fit additional slice requests into the system.

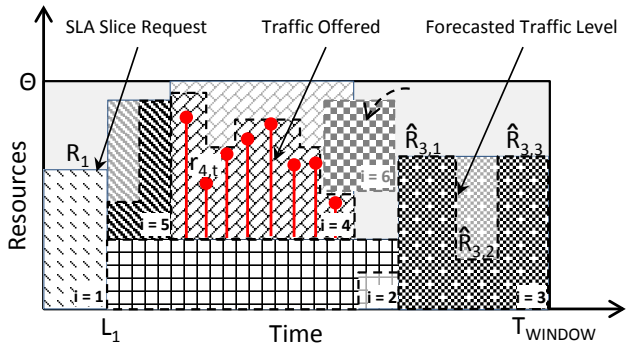


Fig. 2: Admission control problem as geometric knapsack problem.

A mathematical approach is proposed next to fully understand the admission control problem for both cases. Next, we prove its NP-Hardness and suggest a baseline algorithm for allocating network slice requests when no forecasting information is available. Then, we detail the Forecasting-aware Network Slicer algorithm to efficiently perform the admission control phase exploiting accurate traffic pattern information.

A. Problem Formulation

We initially assume a constant amount of resources required for a network slice instantiation. However, we show that relaxing such an assumption, by considering different forecasted traffic levels, makes the problem more complex, but still tractable for our admission control process.

Let us define a network slice request as $\sigma_i^{(k)} = \{R, L, i, k\}$ where i identifies the tenant, R is the amount of resources required, L is the time duration of the slice and k is the traffic class. Without loss of generality, we simply refer to a tenant request as $R_i^{(k)}(L_i)$. Recalling the main objective as the network slice requests accommodation while maximizing the network resource utilization within a fixed time window T_{WINDOW}, we next derive our model.

Let us assume a rectangular box with fixed width W and height H representing the resource availability within a fixed time window. In particular, the box width corresponds to T_{WINDOW} and box height corresponds to the total amount of resources Θ. Let us assume a set of items $\mathcal{I}$, where each item $i \in \mathcal{I}$ corresponds to a network slice request having width w_i corresponding to slice duration L_i and height h_i corresponding to the amount of resources R_i. In addition, each item is provided with a profit c_i corresponding, in our case, to the amount of resources needed. This assumption relies on the fact that every slice request pays the same amount of money proportional to the number of resources granted [1]. The objective of our admission control problem is to find a subset of items $\mathcal{I}' \subseteq \mathcal{I}$ which maximizes the total profit $\sum_{i \in \mathcal{I}'} c_i$, e.g., the total amount of used resources, as shown in Fig. 2.

Lemma 1. *Let the overall system resource availability be a box with height Θ and width T, and let each item $i \in \mathcal{I}$ be the network slice request σ_i with height R_i and width L_i. Then,*

[1]This assumption could be relaxed to reflect a different economic model within the multi-tenancy framework, which is out of the scope of the paper.

the admission control problem is mapped onto a Geometric Two-dimensional knapsack problem with the objective of filling up our system capacity with network slice requests while maximizing the network resources utilization.

Let us now assume tenant requests characterized by a set $\hat{R}_{i,z}^{(k)} = \hat{hh}_{i,z}^{(k,\chi)}$ representing the predicted amount of needed resources per time z for a traffic type k (i.e., given a forecasting error probability $\chi_i^{(k)}$) based on the forecasting phase. This results in time-variant resource requests where shapes are no longer rectangular.

Lemma 1a. *Let the overall system capacity be a box with height Θ and width T, and let each item $i \in \mathcal{I}$ be the network slice request σ_i with irregular shapes, identified by different height values $R_{i,z}^{(k)}$ and width L_i. Then, the admission control problem is mapped onto a Flexibile Geometric Two-dimensional knapsack problem, with the objective of maximizing the network resources utilization whilst accommodating network slice requests, assuring traffic class time constraints.*

An illustrative example is provided in Fig. 2 wherein different amounts of resource values are forecasted for a single network slice request. It may be observed that when the forecasting is accurate, i.e. real traffic (red points) are bounded within new slice values (slice $i = 4$), more room can accommodate more slices ($i = 6$). Please note that in our case the (flexible) geometric two-dimensional knapsack problem is constrained by the orientation law of the considered items. In particular, each item i has a fixed orientation, which can not be changed to fit in the box. Although some state-of-the-art work calls such a problem constrained geometrical knapsack problem, we prefer to omit the "constrained" word as it may refer to additional constraints on the relationship between items stored in the box, which are out of the scope of this work. We can formulate our admission control problem as follows

Problem ADM−CONTROL:

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in \mathcal{I}} c_i \cdot x_i \\ \text{subject to} \quad & \sum_{i \in \mathcal{I}} w_i \cdot x_i \leq W; \qquad \text{(relaxed)} \\ & \mathcal{S}(x_i) \cap \mathcal{S}(x_k) = \emptyset, \quad \forall i \neq k; \\ & \mathcal{S}(x_i) \subset \mathbb{S}, \qquad \forall i \in \mathcal{I}; \\ & x_i \in [0,1], \qquad \forall i \in \mathcal{I}; \end{aligned}$$

where $\mathcal{S}(x_i)$ depicts the geometrical area of the item i (either rectangular or irregular defined) whereas $\mathbb{S}$ is the area of the box, i.e., $|\mathbb{S}| = T \cdot \Theta$. The first constraint refers to the weight of each item. For the sake of simplicity we consider the weight capacity of our box as infinite $W = \infty$ to neglect the item weight. The next two constraints state that items cannot overlap with each other and must be contained within the total space of the box. The solution of such a problem provides a set of x_i, which is a binary value indicating whether the item i is admitted into the system or rejected for the next time window T_{WINDOW}, e.g., a list of granted slice requests in Fig. 1.

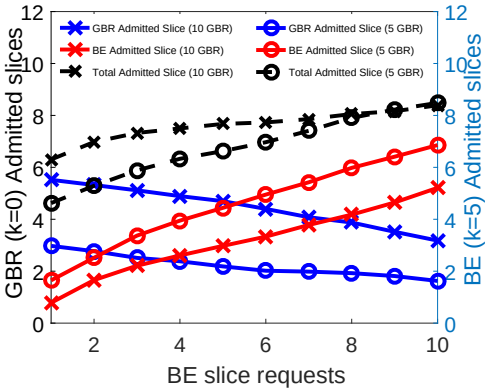


Fig. 3: Admitted slice requests within a time windows while collecting slice requests with GBR traffic requirements ($k = 0$) and best-effort traffic requirements ($k = 5$).

B. Complexity Analysis

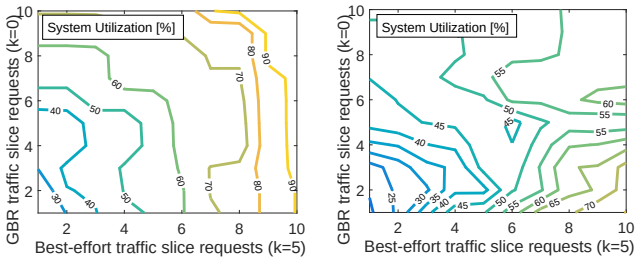
We analyze the complexity issues of the admission control problem providing practical algorithms in the next section. We can formulate the following decision problem DEC−ADM−CONTROL: *given an arbitrary value V, n items with a value c_i and a given area a_i enclosed within a two-dimensional shape identified by $R_{i,z}^{(k)}$ and L_i, and a box with capacity $\mathbb{S}$ delimited by Θ and T, is there a subset $\mathcal{I} \in \{1, 2, \cdots, n\}$ such that items do not overlap and $\sum_{i \in \mathcal{I}} c_i \geq V$?*

Lemma 2. *Considering all items with full flexible dimensions, we can identify one single weight w_i per item representing the area required. Then, if the utility value $c_i = w_i$ the decision problem DEC−ADM−CONTROL reduces to a "Subset Sum Problem".*

Theorem 1. *The decision problem DEC−ADM−CONTROL is NP-Complete and Problem ADM−CONTROL is NP-Hard for any type of traffic k along the network slice request.*

Sketch of Proof: We use a reduction from the subset sum problem based on Lemma 2. We apply a polynomial reduction to the decision problem DEC−ADM−CONTROL considering only items with full flexible dimensions collapsed into a weight w_i and utility value equal to the weights $c_i = w_i$. This reduces the problem to a Subset-Sum problem, known to be NP-COMPLETE. When considering items with fixed resource provisioning, e.g., items with constrained shape values, it is even more difficult to find a solution to Problem DEC−ADM−CONTROL, which proves the NP-Completeness. Based on that, for all $\epsilon > 0$, approximating the solution for Problem ADM−CONTROL, $|\mathcal{I}| = n$ within $n^{1-\epsilon}$, is NP-Hard. This proves that our Problem ADM−CONTROL is NP-Hard. □

Theorem 1 suggests that no optimal poly-time algorithm solves our admission control problem. Interestingly, we remark that the admission control problem is easier when only best-effort slice requests are processed (still NP-Hard). This could negatively drive the infrastructure provider to have a particular tendency for best-effort, or less-demanding, requirements as depicted in Fig. 3. In particular, assuming only stringent traffic



(a) $\eta = 0$; no inter-class priority (b) $\eta = 1$; inter-class priority

Fig. 4: System utilization with different utility functions.

class requirements GBR ($k = 0$) and best-effort (BE) class ($k = 5$), we show the number of admitted slice requests considering different resource demands to the 5G Network Slice Broker. The total number of admitted slices increases with the number of best-effort slice requests showing that best-effort slice requests are preferred due to the higher flexibility. This is further supported by Fig. 4(a), where we show the contour of the total system utilization when different number of slice requests reach the network. Although a disparity between GBR and BE slice requests appears, the utilization of the system is maximized providing outstanding results (more than 90%) in the best case.

Along these lines, we provide a smart mechanism which ensures no traffic inter-class prioritization. We define the utility value in Problem ADM−CONTROL for each slice requests as $c_i = \frac{L_i R_i}{(T^{(k)})^\eta}$, with $\eta \in \{0, 1\}$. For $\eta = 0$, the utility value is exactly the amount of data required within the slice whereas $\eta = 1$ leads to more priority for strict service requirements slices. In Fig. 4(b) we show the contour of the system utilization when $\eta = 1$. While the inter-class fairness is guaranteed (as shown in the top-right part of the picture), the overall utilization degrades exhibiting values around 55% in the best case.

C. Heuristic algorithm design

The admission control needs to cope with different network slice requests and optimize the total utility function, as shown in Problem ADM−CONTROL. Network slice requests can be (i) regularly shaped, i.e., no forecasted information is considered, but with different flexibility degrees due to the traffic class considered, (ii) irregularly shaped exhibiting a different degree of freedom. The first class of network slice requests is handled through a Network Slices Packer algorithm, a revised and improved version of [15]. The second class of network slice problem admits at least the same solution of the first class but, if properly explored, it could provide much more flexibility and resources utilization.

Network Slices Packer. We assume rectangular shapes for network slice requests with different traffic requirements. When traffic class $k = 0$, the regular shape of the network slice is hardly defined and no flexibility is allowed for allocating the traffic requests. Conversely, when less-demanding slice requests $k > 0$ are considered, the slice might be reshaped, delaying the slice traffic, to efficiently fit into the network.

We rely on the assumption that each tenant is not allowed to ask more than the half of the resource availability of the infrastructure provider, i.e., $R_i \leq \frac{\Theta}{2}$. This implies that at least 2 network slices can be accommodated. The algorithm is listed in Algorithm 1. Among all possible pairs of network slice requests, only those fitting the available system capacity are taken into account (line 2). For each 2-slice set (C_l), we formulate a 0-1 knapsack problem to maximize the total profit assuming a single weight (the area of the slice) per item ($R_i \cdot L_i$). The item set to evaluate for the knapsack problem includes the 2-slice set (C_l) and all the other slices (B_l), while considering C_l as already allocated slices. Based on the FPTAS proposed in [16], we retrieve the best solution, i.e., a set of network slice requests ($C_{l*} \cup B_{l*}$) among all knapsack problems (line 9). If the total profit $v(\cdot)$ assigned to the 2-slice set requests C_{l*} is greater than the half of the best profit retrieved after running all knapsack problems, we keep C_{l*} as the best feasible set (line 10-11). Otherwise, we split the optimal set into two subsets $\mathcal{F}_1$ and $\mathcal{F}_2$ (line 13-14). If the total space covered by the items in $\mathcal{F}_1$ is greater than the half of the total system capacity area (line 15), the second subset $\mathcal{F}_2$ will cover less then the half of the available system capacity [2], and a total profit greater than the half of the optimal solution. Therefore, the subset $\mathcal{F}_2 = B_{l*}$ could be easily (in polynomial time) packed into the system capacity (line 16). Otherwise, we

Algorithm 1 Network Slices Packer: Algorithm to admit network slice requests $\sigma_i^{(k)}$ within the system capacity Θ for the next time window T_{WINDOW}.

Input: $\Sigma = \{\sigma_i^{(k)}\}, \Theta, T_{\text{WINDOW}}, \mathbb{S}$
Initialization: $\mathcal{C} \leftarrow \emptyset, \mathcal{F}_1 \leftarrow \emptyset, \mathcal{F}_2 \leftarrow \emptyset, \mathcal{E} \leftarrow \emptyset$
Procedure
1: **for all** $C_l \leftarrow \binom{\Sigma}{2}$ **do**
2: **if** C_l fits into $\mathbb{S}$ **then**
3: $\mathcal{C} \leftarrow \mathcal{C} \cup C_l$
4: **end if**
5: **end for**
6: **for all** $C_l \in \mathcal{C}$ **do**
7: $\{v(C_l \cup B_l), s(C_l \cup B_l)\} \leftarrow$ **Solve** the knapsack problem $P(C_l)$
8: **end for**
9: $l* = \arg\max_{l \in \mathcal{C}} \{v(C_l \cup B_l)\}$
10: **if** $v(C_{l*}) \geq \frac{v(C_{l*} \cup B_{l*})}{2}$ **then**
11: **return** C_{l*}
12: **else**
13: $\mathcal{F}_1 \leftarrow C_{l*}$
14: $\mathcal{F}_2 \leftarrow B_{l*}$
15: **if** $s(\mathcal{F}_1) \geq \frac{|\mathbb{S}|}{2}$ **then**
16: **return** B_{l*}
17: **else**
18: **Sort** $\mathcal{F}_2$ in non-increasing order of their profits and traffic class k
19: **while** $s(\mathcal{F}_1) < \frac{\mathbb{S}}{2}$ **do**
20: e = pop($\mathcal{F}_2$)
21: $\mathcal{F}_1 \leftarrow \{\mathcal{F}_1 \cup e\}$
22: **end while**
23: **if** $v(\mathcal{F}_2) \geq \frac{v(C_{l*} \cup B_{l*})}{2}$ **then**
24: **return** $v(\mathcal{F}_2)$
25: **else**
26: $\mathcal{E} \leftarrow \max\{v(\mathcal{F}_1 \setminus e); v(\mathcal{F}_2)\}$
27: **return** $\mathcal{E}$
28: **end if**
29: **end if**
30: **end if**

[2]Based on the Steinberg's theorem, if the sum of the item areas are less than the half of the box, they can be packed. See A. Steinberg, "A strip-packing algorithm with absolute performance bound 2", *SIAM Journal on Computing*.

move the item with the greatest profit and the highest traffic class k (more flexible) from $\mathcal{F}_2$ to $\mathcal{F}_1$ until the space of $\mathcal{F}_1$ is greater than the half of the system capacity (lines 19-22). Then, if the total profit of $\mathcal{F}_2$ is greater than the half of the optimal one (line 23), the algorithm ends and we keep $\mathcal{F}_2$ as the optimal set. Otherwise, we choose the set providing the best total profit after comparing $\mathcal{F}_2$, without the latest added element, with $\mathcal{F}_1$.

The algorithm provides a performance ratio of at most $\frac{5}{2} + \epsilon$. The first 5 rows of our algorithm are solved within $O(n^2)$ computational time, revealing the number of knapsack problems $P(C_l)$ to be solved. Given that the knapsack problem solution is achieved within a $O(n \log n)$, as the solution is optimal with a moderate number of items, the complexity of the Network Slices Packer is dominated by $O(n^3 \log n)$.

Forecasting-aware Network Slicer. When the forecasted information is available, the admission control gets more room to fit more network slice requests, while still guaranteeing the committed traffic SLAs. The algorithm is reflects the concept of simulated annealing [17]. The additional complexity is due to the feasibility check of a given set of items into the system capacity: packing items in a different order might influence the solution optimality in the next attempts.

We adopt a coding scheme called *sequence pair* [18] to represent candidate solutions of Problem ADM-CONTROL. The solution is represented by a pair of permutations of $|\mathcal{I}|$ items $\{\pi_+, \pi_-\}$. π_+ permutation indicates the spatial relation between items on the horizontal axis, e.g., if i is before j it should be allocated on the left of j. Similarly, π_- refers to the vertical axis. By doing so, the simulated annealing algorithm could easily change the permutations by checking at every step kk whether the new locations are (i) feasible and (ii) provide a greater objective function value, i.e., $\Delta F = F_{kk+1}(x) - F_{kk}(x) > 0$. However, solutions with lower objectives are also accepted according to an admission probability $Pr_a(\Delta F) = \frac{\Delta F}{Tr}$, where Tr is the temperature obtained by the logarithmic cooling function $Tr_k = \frac{Tr_0}{ln(1+kk)}$ and Tr_0 is the initial temperature. The Forecasting-aware Network Slicer algorithm starts by sorting in non-increasing order the slice requests according to their profits (c_i) and traffic class (k). At each step kk, the algorithm decides to shuffle permutations π_+, π_-, add a new item into both sets or remove an existing one. The algorithms stops when the temperature Tr has reached a zero value and no better solutions are found in the next steps. While this algorithm asymptotically finds the global optimal solution, the running time might be not affordable. In Section VII, we provide an empirical complexity analysis with suggestions to improve it.

VI. Scheduling network slice traffic

We design a novel network slice scheduler accomplishing two tasks: (i) serving the tenant traffic of granted network slices, (ii) providing a closed-loop solution for driving the forecasting process to achieve optimal performance (Fig. 1).

A. Multi-class traffic scheduler

We generalize the scheduling model for accounting different traffic SLAs. We assume a traffic request from tenant i for traffic class k as $r_{i,z}^{(k)}$. We consider 6 traffic classes as described in Table I. Each traffic class is characterized by a time window $T^{(k)}$ identifying the offset between two consecutive resource requests $[z; z+1]$, shorter for high-demanding traffic requirements and larger for best-effort class. The scheduler ensures that the whole amount of required resources is served for any given time window $T^{(k)}$.

The key-objective of this novel network slice traffic scheduler is to minimize served resources while guaranteeing the traffic SLAs within a network slice. When forecasted information is available, the scheduler expects slice traffic levels below the predicted traffic bounds such that $r_{i,z}^{(k)} \leq \hat{R}_{i,z}^{(k)}, \forall z \in L_i$. If forecasted traffic bounds are underestimated and the traffic demands exceed the expected values, traffic requests are automatically capped at the original amount of resources agreed during the slice request admission, i.e., $R_i^{(k)}$. Hence, slice allocations may overlap and traffic class requirements might not be fulfilled incurring in slice SLA violations.

We model the scheduler problem as a general minimization problem addressing any traffic class SLA. We introduce the scheduled traffic $s_{i,j}^{(k)}$ representing the real amount of resource served per time j upon the list of admitted slices $x_i^{(k)}$ is available from the admission control phase. The problem is formulated as follows

Problem SLICER-SCHEDULING:

$$\begin{aligned}
\text{minimize} \quad & s_{i,j}^{(k)} \\
\text{subject to} \quad & \left(\sum_{j=zk+\bar{t}}^{zk+\bar{t}+T^{(k)}} s_{i,j}^{(k)} \right) \geq r_{i,z}^{(k)} x_i^{(k)}, \ \forall z \in \left[0, \left\lceil \tfrac{L_i}{T^{(k)}} \right\rceil - 1\right]; \\
& \sum_{i \in \mathcal{N}} s_{i,j}^{(k)} \leq \Theta + P_{i,j}^{(k)}, \quad \forall j \in \mathcal{L}; \\
& s_{i,j}^{(k)} \in \mathbb{R}_+, \qquad \forall i \in \mathcal{N}, j \in \mathcal{L}, k \in \mathcal{K};
\end{aligned}$$

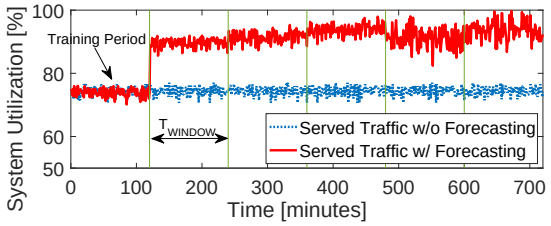
where Θ is the total capacity of the system expressed as the total amount of resource blocks whereas $P_{i,j}^{(k)}$ is the penalty incurred for not having satisfied a particular tenant slice traffic SLA, namely SLA violation. The network slice scheduler keeps track of SLA violations to promptly trigger dynamic forecasting parameters adjustments.

B. Online Reinforcement Learning

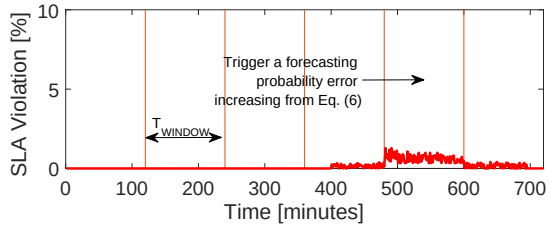
Forecasting process failures may lead the admission control to overbook available network resources and experience SLA violations. A monitoring procedure is designed to keep track of the number of such violations and feed back the forecasting phase through a penalty value $P_{i,j}^{(k)}$ in Problem SLICER-SCHEDULING. We can derive from Eq. (4), the forecasting error probability for a generic traffic class k as follows

$$\chi_i^{(k)} \ : \ h_i^{(k)} \Omega_\chi \sqrt{Var(e_{i,z}^{(k)})} = \hat{d}_i^{(k)} \tag{6}$$

where the penalty history function is defined as $h_i^{(k)} = e^{\frac{nm}{W_S + nm}}$ assuming nm as the number of times the penalty $P_{i,j}^{(k)} = 0, \forall j$, and W_S as the length of the season considered in the forecasting process. The penalty history function drives the system from a conservative behaviour where a higher forecasting error probability is applied to a more aggressive evolution,



(a) System utilization



(b) SLA Violations

Fig. 5: System performance comparison with and without forecasting preprocessing.

when no SLA violation occurs. In case of forecasting failures, a larger forecasting error probability is restored and limited gains are achieved.

VII. Performance Evaluation

We carried out an exhaustive simulation campaign to validate our framework. Our system is evaluated through an ad-hoc simulator developed in MATLAB®with a dual Intel(R) Xeon CPU 2.40GHz 4-cores and 16GB RAM. A summary of the simulation parameters used is provided in Table II. The system includes $|\mathcal{B}| = 7$ base stations ([19]) and $|\mathcal{I}| = 10$ tenants ([1]). The average number of users associated with a tenant is $E[|\mathcal{U}_i|] = 100$, which are distributed uniformly. When they move, a SLAW model is applied [14]. Tenant slice requests may range between the 5% and the 25% of the total system capacity, while their duration ranges between 1000 and 3600s. $P_{i,k}$ defines the probability that a slice request reaches the network within a time window. At the beginning of each T_{WINDOW} the admission control procedure is invoked. Based on the forecasting information, network slice requests are granted for the next time window and the associated slice traffic served.

A. Dynamics and SLA violations

A dynamic analysis of our system is provided here. Since no other work in the literature has proposed a solution for addressing network slice request accommodation, we benchmark our proposal against a legacy solution wherein no forecasted information is available during the admission control phase. The results are shown in Fig. 5(a) for a long simulation period of 720 minutes in terms of system utilization $Ul = \Theta - \sum_{i,k} s_{i,j}^{(k)}, \forall j \in T_{\text{WINDOW}}$, based on Problem SLICER-SCHEDULING. After a prior training period, the forecasting process provides useful information to the admission control block. Based on such information, network slice requests are properly reshaped and more traffic requests are efficiently accommodated into the network capacity. The gain after the second time window is about 20%. While no SLA violation occurs, the forecasting process moves from a conservative behaviour to a more aggressive by reducing the safe margin, i.e., the forecasting error probability from Eq. (6), which visibly brings more gain in terms of system utilization. However, due to the randomness of the traffic requests approaching the system, forecasted information might underestimate the real traffic level resulting in a SLA violation, as shown in Fig. 5(b). This promptly triggers the penalty history function $h_i^{(k)}$ which increases the forecasting error probability in the next time window keeping the SLA violation under control. Interestingly, our solution boosts the system utilization up to 100% at expenses of a small SLA violation per tenant request of about 1.8% in a very short period.

TABLE II: System parameters ([19])

Parameters	Values	Parameters	Values
$\lvert\mathcal{I}\rvert$	10	$\lvert\mathcal{B}\rvert$	7 (21 sectors)
$\lvert\mathcal{K}\rvert$	6	$E[\lvert\mathcal{U}_i\rvert]$	100
Θ	200 RBs	ISD	250m
T_{OBS}	3600 s	η	0
T_{WINDOW}	7200 s	$P_{i,k}$	$\frac{1}{\lvert\mathcal{I}\rvert\lvert\mathcal{K}\rvert}$
L_i	$\{1000; 3600\}$ s	R_i	$\{\Theta * 0.05; \Theta * 0.25\}$
α_{SLAW}	3	Av. Speed	1.5m/s
ρ_{SLAW}	2.5	α_{DISTR}	1.5

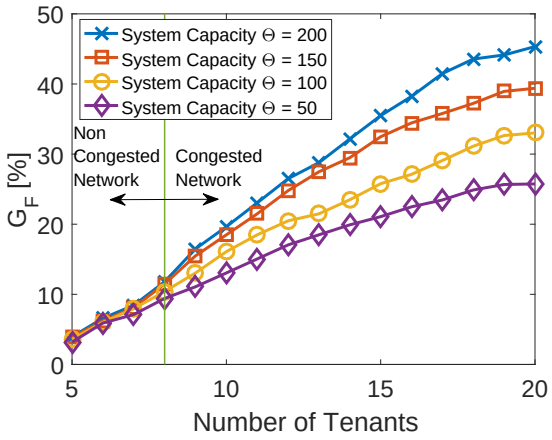


Fig. 6: System utilization gain for different network scenarios.

B. System Capacity Utilization

The helpful forecasted information is evaluated through a number of network scenarios. In Fig. 6, we consider 4 scenarios where the infrastructure provider decides to lease different portions of its own available resources, i.e., Θ from 50 to 200. The results are obtained for the forecasting case and the legacy option, after averaging them over 12 hours time period. We plot the relative gain $G_F = \left(\frac{\bar{Ul}_F}{\bar{Ul}_L} - 1\right)\%$, where $\bar{Ul}_F$ and $\bar{Ul}_L$ are the average utilization value of forecasting and legacy solution, respectively.

We remark two key-points: (i) the increasing slope while augmenting the number of tenants, (ii) the larger the system capacity, the greater the relative gain. A small number of tenants implies a few network slice requests which are fully accommodated exhibiting a small relative gain. As soon as the network becomes congested, i.e., some network slice requests must be rejected, the utilization of our proposal outperforms the legacy scheme ($G_F \gg 0$) due to a wider distribution of net-

TABLE III: Empirical complexity analysis

No. of tenants	10		20		30	
No. of slice req.	30 - 60		60 - 120		90 - 180	
Algorithms	**slices [no.]**	**time [sec]**	**slices [no.]**	**time [sec]**	**slices [no.]**	**time [sec]**
Network Slices Packer	9.584	78.1	11.337	215.7	13.226	497
Forecasting-aw. Network Slicer	9.607	129.5	**13.061** (11.897)	**714** (600)	**15.81** (13.307)	**3514.4** (600)

work slice request values. On the other hand, when the system capacity is improved, there is more room to accommodate the network slices into the system showing better performances. This intrinsically suggests the infrastructure provider to lease as much as possible network resources portion to improve the overall system utilization.

C. Algorithm Complexity

We finally provide an empirically study of the computational cost for the two admission control algorithms proposed. For a fair comparison, we apply the algorithm to the same instances of the problem and average the results over several instances (100). We only compare the case where regular network slice shapes are considered, as the network slices packer is not designed to handle different problems.

In Table III, we show the results for different number of tenants asking for network slice requests. The results are expressed in terms of number of slices admitted into the system capacity and time elapsed for getting the solution. The average number of network slice requests within a single instance of the problem ranges from 30 (10 tenants) to 90 (30 tenants). Interestingly, the Forecasting-aware Network Slicer algorithm outperforms the Network Slices Packer but experiencing a longer computational time. We apply a time limit $T_Z = 600$s to avoid the process starvation. Given the long-term execution (every 30 minutes) of those admission control algorithms, this time bound is still valid for the overall system implementation. Nonetheless, the Forecasting-aware Network Slicer algorithm shows reasonable results in an affordable computing time.

Conversely, when irregular shape patterns are considered, the time complexity of the Forecasting-aware Network slicer gets even worse. This drawback is significant when the number of network slice requests is greater then 50. We overcome such a problem in the following way. We apply the network slices packer algorithm for the equivalent version with regular shapes. This provides an initial state for the Forecasting-aware Network slicer, which can start exploring the neighbouring solutions while checking whether they fit into the system capacity. In this way, we are able to reduce the complexity time up to 20% for the Forecasting-aware Network slicer making it reasonable for realistic deployments.

VIII. Conclusions

In this paper we have presented our proposed network slicing traffic forecasting, admission control and scheduling solutions for a 5G Network Slice Brokering system. The forecasting solution designed builds on Holt-Winters theory to predict future traffic levels per network slice considering differentiated classes and such that the system utilization can be maximized by an admission control decision engine. The admission control solution has been mapped into a geometric knapsack problem and two low-complexity algorithms designed for regular and irregular network slice requests. The network slice scheduling solution keeps track of SLA violations per slice and feeds them back to the forecasting engine such that it adaptively corrects to deviations. Our main findings can be summarized as follows: i) Holt-Winters theory can be exploited for network slicing traffic forecasting and applied to regular and irregular requests, ii) elastic traffic network slice requests help in increasing the maximum achievable system utilization, iii) the forecasting benefits increase as the number of network slice requests and system capacity increases and iv) low SLA violation risk levels result in very significant system utilization gains.

References

[1] NGMN Alliance, "NGMN 5G White Paper," Tech. Rep. Version 1, February 2015.

[2] K. Samdanis, X. Costa, and V. Sciancalepore, "From Network Sharing to Multi-tenancy: The 5G Network Slice Broker," *IEEE Communications Magazine*, July 2016.

[3] Third Generation Partnership Project (3GPP), "Study on Radio Access Network (RAN) Sharing enhancements," 3GPP TS 22.852, Sept 2014.

[4] H. Wen and P.K. Tiwary and T. Le-Ngoc, "Wireless Virtualization," Springer Briefs in Computer Science, Sept 2013.

[5] Third Generation Partnership Project (3GPP), "Network Sharing; Architecture and Functional Description," 3GPP TS 23.251, June 2016.

[6] ——, "Telecommunication management; Network Sharing; Concepts and requirements," 3GPP TS 32.130, June 2016.

[7] R. Mahindra, M. A. Khojastepour, H. Zhang, and S. Rangarajan, "Radio access network sharing in cellular networks," in *21st IEEE International Conference on Network Protocols (ICNP)*, Oct 2013, pp. 1–10.

[8] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "NVS: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking*, no. 5, pp. 1333–1346, 2012.

[9] X. Costa-Perez, J. Swetina, T. Guo, R. Mahindra, and S. Rangarajan, "Radio access network virtualization for future mobile carrier networks," *IEEE Communications Magazine*, pp. 27–35, July 2013.

[10] J. S. Panchal, R. D. Yates, and M. M. Buddhikot, "Mobile network resource sharing options: Performance comparisons," *IEEE Transactions on Wireless Communications*, vol. 12, pp. 4470–4482, September 2013.

[11] G. Tseliou, K. Samdanis, F. Adelantado, X. Costa, and C. Verikoukis, "A capacity broker architecture and framework for multi-tenant support in LTE-A networks," in *IEEE International Conference on Communications (ICC)*, May 2016.

[12] A. B. Koehler, R. D. Snyder, and J. Ord, "Forecasting models and prediction intervals for the multiplicative holtwinters method," *International Journal of Forecasting*, vol. 17, no. 2, pp. 269 – 286, 2001.

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

[14] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "SLAW: Self-similar least-action human walk," *IEEE/ACM Transactions on Networking*, pp. 515–529, April 2012.

[15] K. Jansen and G. Zhang, "Maximizing the total profit of rectangles packed into a rectangle," *Algorithmica*, pp. 323–342, Mar. 2007.

[16] H. Kellerer and U. Pferschy, "A new fully polynomial time approximation scheme for the knapsack problem," *Journal of Combinatorial Optimization*, vol. 3, pp. 59–71, 1999.

[17] A. Liu, J. Wang, G. Han, S. Wang, and J. Wen, "Improved simulated annealing algorithm solving for 0/1 knapsack problem," in *Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications*. IEEE Computer Society, 2006, pp. 1159–1164.

[18] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "VLSI module placement based on rectangle-packing by the sequence-pair," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, pp. 1518–1524, Dec 1996.

[19] ITU-R, "Guidelines for evaluation of radio interface technologies for IMT-Advanced," Report ITU-R M.2135-1, Dec 2009.