PREPRINT VERSION

Anticipatory Resource Allocation and Trading in a Sliced Network

Özgür Umut Akgül*†, Ilaria Malanchini†, and Antonio Capone*
*Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, Milano, Italy
E-mail: oezguerumut.akguel, antonio.capone@polimi.it
†Nokia Bell Labs, Stuttgart, Germany
E-mail: ilaria.malanchini@nokia-bell-labs.com

***Abstract*—Dynamically sharing network resources in a sliced multi-tenant network can provide cost efficient solutions that are able to guarantee specific service requirements for 5G networks and beyond. By automatizing the negotiations between tenants and infrastructure providers over the shared resources, it is possible to maximize the flexibility of the network in a very short time frame, thus increase efficiency. However, negotiating resources in a reactive manner can bring risks to the tenants due to traffic variations, and can also limit the gain in terms of spectral efficiency for the infrastructure provider. In this paper, we focus on how to exploit anticipatory strategies relying on predicted information on users' conditions in order to improve the efficiency of the proposed dynamic network slicing and trading framework. In particular, we analyze how to integrate a prediction algorithm into our scheme and analyze the techno-economic impacts of the anticipatory approach. Finally, we introduce a novel filtering algorithm to limit the impacts of prediction errors. Our results prove that using anticipatory strategies in dynamic negotiations and resource allocation increases tenants' utilities, while allows the infrastructure provider to accommodate more requests.**

I. Introduction

The saturation in the consumer market of mobile broadband services and the decreasing profitability of network providers make cost efficiency the dominant factor in the transition towards the next generation wireless networks. As cost reduction becomes a key challenge, network operators begin searching for alternative revenue sources. Recent findings reveal that focusing on specialized industry segments can boost revenues as high as to 36% [1]. On the other hand, in order to accommodate these new markets, the available network infrastructure has to support a multitude of vertical applications, each one with diverse set of requirements, thus challenging the current networks' technical capabilities. A relatively new idea to address heterogeneous requirements of different services is to couple connectivity services with storage and processing resources [2]. This naturally leads to the logical slicing of the network resources and to optimizing each slice individually according to the requirements of the specific service. Therefore, by means of *network slicing*, network resources can be customized to achieve the target quality of service (QoS) per service type [3].

The possibility of vertically slicing the network resources to accommodate services with heterogeneous requirements can at the same time be combined with the idea of sharing the infrastructure resources among multiple mobile virtual operators. This way the total cost can efficiently be reduced. *Infrastructure sharing* can reshape the wireless market, allowing virtual mobile operators to act as *tenants* of network resources, while favoring mobile operators to act as *infrastructure providers* [4]. In this evolving wireless market, mutual relationships rely on well-defined service level agreements (SLAs). This requires tenants to have a very clear understanding of their resource needs as well as of the evolution of the traffic demand within the validity period of the agreed SLAs [5] [6]. However, a static allocation of resources to slices cannot provide the required flexibility and efficiency for the envisioned 5G networks and beyond. To exploit the full potential of infrastructure sharing, *dynamic network slicing* is proposed. Dynamic slicing frameworks as proposed in the literature still require well defined SLAs (e.g. [7])) in this new context. These frameworks thus are not able to dynamically adapt the resource prices in order to fully use the network resources according to the demand (e.g. [8]). Therefore, they cannot provide the efficiency and flexibility required by a dynamic wireless resource market. With these objectives in mind, we proposed in [9] a dynamic slicing and resource trading platform. In this platform, the tenants determine a set of high-level SLA policies in line with their business strategies, which are automatically translated into sharing parameters, to be used for the real time resource allocation. Moreover, tenants are able to update these high-level parameters over time.

Despite the level of autonomy and flexibility that we achieved in [9], high-level renegotiations still occur in a reactive manner, that results in tenants' taking business risks, while at the same time it limits the efficiency of resource allocation for the infrastructure provider. Therefore, in this work, we focus on exploiting anticipatory information [10] during the negotiation period to maximize the efficiency of network slicing. More specifically, we investigate how to exploit anticipation in a dynamic network slicing framework. We also study the techno-economic implications of negotiations when the key stakeholders (i.e. tenants and infrastructure provider) can predict their users' upcoming achievable rates. In the context of anticipatory network slicing, [11] tackles the resource allocation problem using predictions on the upcoming resource demand. However, this work relies on static SLAs, which cannot fully exploit the flexibility and the efficiency of

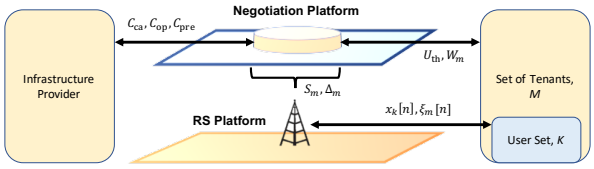


Fig. 1. Proposed negotiation and resource scheduling platform

dynamic network slicing. To the best of our knowledge, our work is the first to focus on using anticipatory strategies with the goal of providing both flexibility with recurring negotiations within a short time frame and efficiency of dynamic resource allocation.

The contributions of this work can be summarized as follows.

- We analyze how to use anticipatory strategies in a real time network slicing and trading problem,
- We propose a techno-economic model that exploits anticipated information for the real-time resource scheduling and trading,
- We design a novel filter to limit the impact of prediction errors.

The remainder of the paper is organized as follows. The outline of the proposed system model and the considered prediction algorithms are presented in Section II. Section III introduces the proposed anticipatory network slicing and trading framework and shows how to exploit predicted information. The numerical analysis of the proposed approach is presented in Section IV, while Section V concludes the paper.

II. System Model

Extending our previous work [9], we investigate the dynamic negotiation platform provided in Fig. 1. This figure summarizes the interaction between the key stakeholders in our model, namely an infrastructure provider, a set of tenants (M) and a set of users (K). We use indexes k and m to indicate a specific user and tenant, respectively. For the sake of simplicity, we assume that each user requires only one service type and the total number of users $|K|$ is distributed evenly among the tenants and shown as $|K_m|$, where $\cup_{m \in M} K_m = K$. Following the general approach in resource allocation literature, the time horizon, N, is discretized and divided into time slots that are indexed with n. We assume that each time slot n spans over a number of transmission time intervals (TTIs) that can be determined according to the complexity of the negotiation algorithm and the computational capacity of the base station.

The SLAs that control the resource sharing between the key stakeholders are modeled using three parameters, i.e. S_m, Δ_m and W_m. The guaranteed resource share, indicated by $S_m \in [0, 1]$, is defined as the average resource share that the tenant m receives on average. In order to exploit the dynamic nature of the wireless environment, the maximum deviation from SLA for a given time window length W_m is limited by Δ_m. Therefore, the delay constraint per tenant is indirectly taken into account by using W_m. The tenant-specific sharing parameters, S_m and Δ_m, are updated at each renegotiation interval (RI).

As shown in Fig. 1, tenants set their utility targets, U_{th}, and their respective budgets, B_m. The total cost of wireless resources is modeled as the sum of the operational costs (C_{op}), capital costs (C_{ca}) and the pressure cost (C_{pre}). Pressure cost is used to ensure efficient resource usage and to provide extra revenue for the infrastructure provider to expand the network capacity where congestion is often experienced. More specifically, in line with any demand-based market, the pressure cost scales the unit cost according to the instantaneous demand; if there are insufficient resources to satisfy all the users, pressure cost becomes greater than zero, making it more expensive to buy resources. Otherwise, if there are sufficient resources to satisfy all the users, it will be equal to zero. Thus, the accumulated pressure cost measures the cost of necessary additional capacity to fully satisfy all the users.

Based on the decided sharing parameters per tenant and the achievable rate of each user k, i.e. $r_k[n]$, the real time scheduler assigns resources to each user k, i.e. $x_k[n]$. The actual achieved rate of k at any time slot n is evaluated as $r_k[n]x_k[n]$ and is used to calculate the utility of each user $U_k[n]$. We assumed that each user has an equivalent weight in the tenant's utility target, meaning that the total achieved utility of the tenant is $\sum_{k \in K_m} \frac{U_k[n]}{K_m}$.

A. Anticipatory strategies

In the proposed system model, we assume that the parameters that are anticipated (i.e. predicted) by the tenants are the achievable rates of users. Therefore, our framework optimizes the network parameters in response to the network conditions. Furthermore, the negotiation algorithm runs in real-time, thus, the prediction algorithm is also required to make predictions within the same time frame. When selecting a suitable prediction algorithm for a real time system, the primary focus shall be on its computational complexity. The second factor to be taken into account is prediction accuracy. For this reason, we have selected two well-known prediction methods that are proved to be able to reach high prediction accuracy at the cost of reasonable computational complexity. Those are the Auto-Regressive Integrated Moving Average (ARIMA) and Feed Forward Neural Networks (FFNN)[1].

1) ARIMA: ARIMA is widely applied for time series prediction, due to its low complexity and high performance. Unlike most of the deep learning mechanisms (e.g. [12]), ARIMA does not require a long history of the observed function [13]. With a relatively small set of samples, it sets a few parameters that model the function behavior and anticipates possible future values. ARIMA contains five major parameters, namely, the prediction window W_p, the learning window W_l, the number of auto-regressive terms p, the number of non-seasonal differences d and the number of moving average

[1]Note that the proposed framework could also work with different prediction schemes.

$$\min_{x_k[n],S_m,\Delta_m} \quad \sum_{m\in M} \xi_m[n] \tag{1a}$$

$$\text{s.t.} \;\; U_{th,m} - \sum_{k\in K_m} U_k(R_k[n]) \leq \xi_m, \;\; \forall m \in M, \tag{1b}$$

$$\epsilon_m[n] = \left(\frac{1}{(a_m+1)} \sum_{i=n-a_m}^{n} \sum_{k\in K_m} x_k[i]\right) - S_m, \;\; \forall m \in M, \tag{1c}$$

$$|\epsilon_m[n]| \leq \Delta_m, \quad \forall m \in M, \quad \forall n \in N, \tag{1d}$$

$$\sum_{i=n-a_m}^{n} (S_m(C_{\text{ca}} + C_{\text{op}}) + \epsilon_m[i]C_{\text{op}} + f_{\text{pre}}(C_{\text{pre}}, \; \xi_m)) \leq B_m(a_m+1), \forall m \in M, \tag{1e}$$

$$0 \leq \Delta_m \leq \frac{1}{a_m+1} \sum_{i=n-a_m}^{n} \sum_{k\in K_{m,elastic}} x_k[i], \quad \forall m \in M, \tag{1f}$$

$$\sum_{k\in K} x_k[n] \leq 1, \quad x_k[n] \geq 0, \quad \forall k \in K, \tag{1g}$$

$$\sum_{m\in M} S_m \leq 1 \;, \quad S_m \geq 0, \quad \forall m \in M, \tag{1h}$$

terms q. At first, the algorithm, taken the past observations over W_l, estimates the correlation between the past and current time slots and finds the optimal set of (p, d, q) parameters. Afterwards, the algorithm, using the ARIMA parameters (p, d, q) found, predicts the future values of the series over W_p.

2) FFNN: FFNN is widely applied due to its high precision and capability to approximate complex functions [10]. Unlike ARIMA, FFNN requires a relatively long learning period, during which it learns the correlation among the samples and updates the weights of the neural network accordingly. FFNN is defined using five major parameters, namely, the learning window (W_l), prediction window (W_p), the number of nodes in the hidden layer (D_N), the number of hidden layers (H) and finally the number of delays (E). In our implementation, each observed $r_k[n] \; \forall k \in K$ is fed to one node. Thus, D_N also indicates the number of past samples that are used to predict the future values. Note that the prediction of the FFNN is based on a sliding window. More specifically, using previous D_N samples, the algorithm predicts $D_N + 1^{\text{th}}$ values. After feeding the predicted values as an input, it predicts $D_N + 2^{\text{th}}$ values. This continues until it reaches $D_N + W_p$.

III. Anticipatory Resource Scheduling

A. Mathematical programming formulation

To model the anticipatory resource scheduling problem, we propose a mathematical programming formulation as in (1a)-(1h). The model distributes the network resources in real time, while enables a market-driven pricing mechanism according to the QoS requirements of services, achievable rates of the users, tenants' budgets and utility goals. The continuous objective function (1a) minimizes the total gap of the tenants ξ_m, thus maximizes the resource efficiency. The gap is defined according to constraint (1b) as the difference between the tenant's expected utility $U_{th,m}$ and the actual achieved utility, $U_k(R_k[n])$. The tenants' utility function can be defined according to the specific service type and requirements. In general, we assume that it is a function of the average achieved rate. Namely, at any time slot n, we evaluate the average achieved rate over a time window a_m+1, where $a_m \equiv n-1 \mod W_m$. Formally, the average achieved rate over such time window is

$$R_k[n] = \frac{1}{(1+a_m)}\left(\sum_{i=n-a_m}^{n} x_k[i]r_k[i]\right). \tag{2}$$

The maximum instantaneous deviation per tenant m at time slot n, $\epsilon_m[n]$, is defined in (1c) and is constrained to be less than or equal to Δ_m in (1d). Constraint (1e) binds the economical aspects of sharing to the scheduling decisions. Namely, the left-hand side of Equation (1e) evaluates the total cost for a tenant as the sum of capital expenditures (CapEx), operational expenditures (OpEx), and pressure cost. The first expression, i.e. $S_m(C_{\text{Ca}} + C_{\text{Op}})$, reflects the fact that tenants are required to pay both CapEx and OpEx according to their resource shares in the network. On the other hand, flexibility is provided by the second expression, i.e. $\epsilon_m[i]C_{\text{op}}$, that is scaled according to the tenants' actual resource usage. More specifically, this expression indicates that the tenants are only obliged to pay OpEx for the resources that are obtained from the resource pool. Consequently, it gives an economic incentive to share resources rather than to have exclusive ownership. The third expression, i.e. $f_{\text{pre}}(C_{\text{pre}}, \xi_m)$, is the pressure cost. By definition, the pressure cost is a means for the infrastructure provider to regulate the resource price. In this paper, we assume that the infrastructure provider has no profit targets, but reinvests all the obtained revenue for network expansion. Therefore, the pressure cost is modeled as the product of the tenant's gap and the unit pressure cost, i.e. $\xi_m C_{\text{pre}}$. Note that since ξ_m is actually a variable of our problem, but the pressure cost is not, we assume that the average gap of the previous renegotiation interval is used to evaluate the pressure cost for the next interval. The right-hand side of Equation (1e) is the budget of tenant m, B_m, defined per time slot. Therefore, by multiplying it by the time window length, a_m+1, and summing up the total costs on the left-hand side, the constraint allows tenants to utilize the unused budget from the previous time slots in the next time slots, within the same time window.

The upper bound for Δ_m is set to be the total amount of assigned resources to the elastic services (cf. (1f)), implying that tenants are not willing to trade resources assigned to non-elastic services[2]. Constraints (1g) and (1h) impose the total assigned resources and the total resource shares, respectively, to be less than or equal to the available ones.

[2]As explained in Section IV-A, by elastic services we refer to services without strict requirements on throughput, in contrast to inelastic services.

B. Exploiting the prediction data

In Section II-A, we clarified that the main objective of this work is not to introduce a new prediction algorithm, but to analyze how to exploit anticipatory information with recurring negotiations within a short time frame to increase the resource usage efficiency. Therefore, for the sake of generality, in this section we assume the prediction algorithm to be a "black box", while we focus on how to exploit the predicted information.

One way to do this is to directly incorporate the predicted information into the proposed model (1a)-(1h), and assign the predicted resource allocations that minimize the objective function. However, this approach, which would perfectly work with perfect (i.e. error-free) prediction, is not robust to errors, which have direct impact on the slice configuration. Therefore, since guaranteeing high prediction accuracy in real time is quite challenging, we tackle the problem by splitting our original formulation into two subproblems, namely to P_1 and P_2, as in our previous work [9]. However, our aim with the two-step algorithm in this case is different from what we had in [9]. In this work, we aim to achieve robustness against prediction errors.

Formally, P_2 is defined by the complete mathematical model as in (1a)-(1h), and the predicted data is used to determine the predicted minimum gap, ξ_m^{pre}, sharing parameters, S_m^{pre}, Δ_m^{pre}, and resource distribution among users, x_k^{pre}. In case of perfect prediction, x_k^{pre} is the optimum resource distribution, while ξ_m^{pre} is the minimum achievable gap in the given network.

To limit the effects of prediction errors, the update process of the sharing parameters is built on the following weighted approach. At the beginning of each renegotiation interval, the sharing parameters of the tenants are updated using a scaling coefficient α_m, as follows:

$$S_m^{\text{new}} = (1-\alpha_m)S_m^{\text{pre}} + \alpha_m S_m^{\text{old}}, \tag{3}$$

$$\Delta_m^{\text{new}} = (1-\alpha_m)\Delta_m^{\text{pre}} + \alpha_m \Delta^{\text{old}}, \tag{4}$$

where α_m is defined as:

$$\alpha_m = \frac{|\xi_m - \xi_m^{\text{pre}}|}{\xi_m + \xi_m^{\text{pre}}}. \tag{5}$$

In case of perfect prediction, the achieved gap is equal to the predicted gap ($\xi_m = \xi_m^{\text{pre}}$), pushing α_m to zero. In this case, the sharing parameters are set to be equal to the predicted sharing parameters. On the other hand, if the predicted values are far from the real ones, the measured gap is higher than the predicted gap ($\xi_m >> \xi_m^{\text{pre}}$), pushing α_m to one. In this case, the prediction is assumed not to be reliable and the predicted sharing parameters are not considered in the updating process, thus the scheduler maintains the previous sharing parameters[3].

Consequently, P_1 receives ξ_m^{pre}, x_k^{pre}, S_m^{new}, Δ_m^{new} and r_k^{pre} as input from P_2 and using (1a), (1b), (1c), (1d), (1e) and (1g), determines the real time resource allocations ($x_k[n]$) according to the actual achievable rates per time slot ($r_k[n]$). The predicted resource distribution x_k^{pre} is used as an upper-bound to the real time resource scheduling, i.e.

$$x_k^{\text{pre}} \geq x_k[n]. \tag{6}$$

In this way, the real time scheduler can make small adjustments on the resource allocations, by taking into account possible prediction errors.

[3]Note that the evaluation of the accuracy of the prediction is done a posteriori, and the information is then used to update the parameters of the next renegotiation interval.

C. Active filtering

The proposed two-step algorithm is designed to cope with prediction errors, which can however limit the achieved performance. In particular, when prediction accuracy is low, Equation (6) can prevent some users from obtaining resources, while forcing the scheduler to assign resources to others. Thus, in this section, a simple, yet efficient filter is proposed to limit the impacts of prediction errors, while exploiting the anticipatory information of users' future achievable rates.

The proposed filter is defined as

$$F(x_k^{\text{pre}}[n], E_k[n]) = x_k^{\text{pre}}[n] + \frac{E_k[n]}{1 + e^{-a_{1,k}(E_k[n]-a_{2,k})}} \tag{7}$$

where $x_k^{\text{pre}}[n]$ is the calculated optimum resources for the predicted rates, $E_k[n]$ is the prediction error, $a_{1,k}$ and $a_{2,k}$ are filter parameters. In order to calculate the prediction error, we used the Euclidean distance between the predicted achievable rate and the measured achievable rate, i.e. $E_k[n] = |r_k^{\text{pre}}[n] - r_k[n]|$. Note that in case of perfect prediction, the output of the proposed filter mechanism is equal to $x_k^{\text{pre}}[n]$.

The filter's sensitivity to the prediction errors depends on $a_{1,k}$ and $a_{2,k}$. Since the effect of prediction errors on resource distribution is influenced by the traffic mix and the network state, $a_{1,k}$ and $a_{2,k}$ are chosen to be dynamic. More specifically, these values are calculated as

$$a_{1,k} = \mu_{n \in W_m}(E_k), \tag{8}$$

$$a_{2,k} = \frac{10}{\sigma_{n \in W_m}(E_k)}, \tag{9}$$

where μ and σ represent the average and the standard deviation of the error, respectively. These parameters are updated at the end of every RI based on the prediction errors observed during the entire RI.

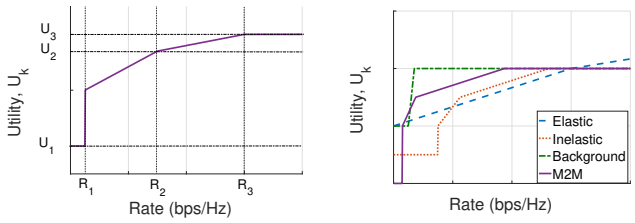
The output of the filter function $F(x_k^{\text{pre}}[n], E_k[n])$ sets an upper limit to the assignable resources in P_1, namely,

$$F(x_k^{\text{pre}}[n], E_k[n], \beta_k[n]) \geq x_k[n]. \tag{10}$$

Depending on the prediction error $E_k[n]$, the assignable resources vary within $x_k^{\text{pre}}[n] \leq x_k[n] \leq 1$.

IV. Simulation Results

We consider a single base station with a coverage radius of 500 m. $|K| = 12$ users share the downlink of this base station and are distributed uniformly within the coverage area. Users are assigned to $|M| = 3$ tenants evenly, hence $|K_m| = 4$. The presented results are averaged over 50 independent instances and each of these instances covers a simulation horizon of



(a) Generic utility function (b) Exemplary utility functions

Fig. 2. Generic utility function (left) and exemplary utilities (right)

5000 TTIs, i.e. $N = 5000$ TTIs. Moreover, the simulation horizon is discretized into time slots with a length of 1 TTI.

The simulations are run by using Matlab 2017a, while the proposed mathematical formulation is solved by the Gurobi solver [14]. Users are assumed to move along a straight line towards a random direction and with a walking speed $v = 1.5$ m/s. The users' achievable rate is calculated using the Shannon-Hartley theorem, i.e. $r_k[n] = \log 2(1 + SINR_k[n])$. For each user k, the SINR is calculated under constant transmission power, P_{Tx}, and constant inter-cell interference, I_0, using the equation $SINR_k[n] = |h_k[n]|^2 P_{Tx} d_k^{-\alpha}/\sigma^2 + I_0$, where d_k indicates the distance of user k, α is the path loss exponent and σ^2 is the sum of the thermal noise. Moreover, a frequency-flat fading channel with Rayleigh coefficients, i.e. $h_k[n]$, is assumed to be between the user and the base station. The maximum Doppler spread is calculated using $F_d = v f_c/c$, where f_c indicates the carrier frequency of 2 GHz, c is the speed of light and v is the walking speed.

A. Utility functions

The utility of each user is evaluated according to the average achieved rate within the considered time window. In order to model the utility, we designed a piece-wise linear function (cf. Fig. 2(a)) that is defined by six parameters, i.e. R_1, R_2, R_3, U_1, U_2 and U_3. We opt for such utility functions for the sake of mathematical tractability, however, the proposed anticipatory slicing framework can use more complicated utility functions.

In the proposed utility function, when the actual achieved rate is smaller than the minimum rate requirement, R_1, we assume the service not to be active and the achieved utility value is $U_1 \leq 0$. R_2 is set to be the minimum rate for the service to receive standard quality and corresponds to the utility value U_2. The region between $R_1 - R_2$ is designed to have a steep slope due to the user's sensitivity to changes in the achieved rate, thus to QoS. Finally, R_3 indicates the achieved rate that produces the maximum utility, U_3. Note that any further increase in the achieved rate after R_3 does not affect the utility.

Similar to our previous work, [9], the heterogeneity in envisioned 5G services is captured by considering four major service types, i.e. *elastic services, inelastic services, machine to machine services (M2M)* and *background services*. The service types and their utility functions are designed as given in Fig. 2(b). Namely, elastic services do not have strict delay or rate constraints, thus $R_1 = 0$, $U_1 = 0$. Moreover, it is assumed that they do not have any upper bound on their rate expectations, meaning $R_3 \to \infty$. Inelastic and M2M services are modeled according to the three regions defined by R_1, R_2 and R_3, and U_1 is assumed to be lower than zero (which means that not fulfilling these service requirements decreases the total utility). For M2M services, it is assumed that each piece-wise linear region captures a different type of device group, namely, emergency, low-rate-delay-sensitive and rate sensitive. Finally, background services are assumed to need a very low rate. The utility achieved is mapped directly to U_3 when such requirement is satisfied, consequently, $R_2 = R_3$ and $U_1 = 0$.

TABLE I
COMPARISON BETWEEN ARIMA AND FFNN IN TERMS OF ACCURACY LEVELS AND TIME COMPLEXITIES

	ARIMA	FFNN
Time complexity for training process (sec)	0.428	75.03
Time complexity for prediction process (sec)	0.722	0.598
Prediction error for $\|W_P\| = 10$ms (MAPE)	7.61 %	7.14 %
Prediction error for $\|W_P\| = 50$ms (MAPE)	160.8 %	216.8 %
Adaptability to varying time conditions	Yes	No

B. Comparison between different prediction methods

The applicability of the considered anticipation techniques, i.e. ARIMA and FFNN, is investigated both in terms of prediction accuracy and time complexity. The time complexity of each algorithm is collected from a commercially available computer equipped with i7-4510U CPU and 16 GB RAM. Following the general approach in literature, the total time to run the prediction algorithms is divided into two parts, i.e. the training time and the prediction time (cf. Table I). The training time is the required time for the respective prediction method to build a mathematical model in order to perform predictions. On the other hand, the prediction time is composed of the total time spent on making the prediction for the upcoming renegotiation interval.

In order to evaluate the accuracy of the prediction algorithm, we used the well-known mean average percentage error (MAPE) and the mean square error (MSE), evaluated as

$$\text{MAPE}(\%) = \frac{100}{N \times |K|} \sum_{k \in K} \sum_{n \in N} \frac{|r_k^{\text{pre}}[n] - r_k[n]|}{r_k[n]}, \quad (11)$$

$$\text{MSE} = \frac{1}{N \times |K|} \sum_{k \in K} \sum_{n \in N} (r_k^{\text{pre}}[n] - r_k[n])^2. \quad (12)$$

Table I shows that FFNN has a higher prediction accuracy (in terms of MAPE) for a shorter prediction horizon, while ARIMA is more successful for longer W_P. Moreover, ARIMA can self-adapt to the changing conditions over time, such as users' position or speed, whereas, FFNN requires to be retrained in order to maintain its prediction performance. Consequently, we conclude that ARIMA is more suitable for

TABLE II
PREDICTION ACCURACY FOR DIFFERENT VALUES OF W_P AND W_L

Scenario (W_P, W_L)	MAPE (%)	MSE
(10,10)	7.61	0.101
(10,50)	7.34	2.14
(10,90)	12.81	0.69
(25,25)	76.64	1.10
(25,50)	29.86	0.84
(25,75)	28.30	0.77
(50,50)	165.9	3.70

our problem and for the remainder of the paper, we only used ARIMA as prediction technique.

Table II outlines the accuracy of ARIMA for different W_P and W_L values. Note that in the given simulation scenario, the correlation window of the achievable rates is 100 TTIs, thus, the analysis is limited to the $W_P \leq 100$ TTIs. The results underline the importance of W_L as it has direct effect on both MAPE and MSE. Despite the usual approach of choosing $W_P + W_L = 100$ TTIs (i.e. the correlation window), our analysis showed that for smaller W_P, having a relatively too big W_L results in an overfitting problem and drastically decreases the accuracy. Moreover the first two rows in Table II have similar MAPE values, while they show a clear difference in terms of MSE. Due to the square of the prediction error in (12), bigger prediction errors are more visible in (12) with respect to (11), meaning that if two scenarios have identical MAPE values, in the one with the smaller MSE, the prediction errors are more uniformly distributed. Our simulations show that the proposed model performs approximately 3% better when using $|W_L| = 50$ compared to the case where $|W_L| = 10$, despite the huge difference in terms of MSE. This also implies that the average error has greater impact on our algorithm than instantaneous errors.

C. Robustness to the prediction errors

Fig. 3 reports the average total utility over all the users for $|M| = 2$, for different prediction horizons and for three different prediction approaches, i.e. no prediction, prediction without filter (i.e. 'no filter') and with filter. In the scenario without prediction, the reactive model presented in [9] is implemented. We observe that, regardless of the length of the prediction horizon, the application of the proposed filter is proven to improve the performance (in terms of average total utility) with respect to both 'no prediction' and 'no filter'. Moreover, increasing the prediction horizon (and the RI) decreases the prediction accuracy for the 'no filter' scenario, which results in lower average achieved utility. Furthermore, we can also observe from Fig. 3, that for both the 'no prediction' and 'with filter' scenarios, an increase in the total average utility is achieved while increasing the length of RI. This increase indicates that the proposed filter mechanism can filter out the negative effects of low prediction accuracy and exploits the accurate anticipatory information. Moreover, from Fig. 3 we can note that the difference among the achieved utilities for the three scenarios is small for shorter prediction horizons.

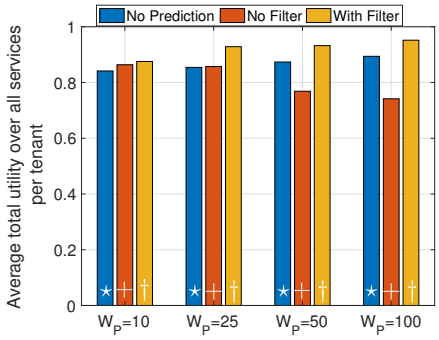


Fig. 3. Comparison of the average achieved utility for different W_P lengths and different scenarios, i.e. no prediction (blue bar marked with " $\star$ "), no filter (orange bar marked with " $+$ ") and application of filter (yellow bar marked with " $\dagger$ ")

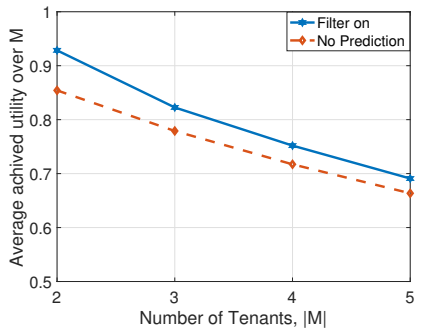


Fig. 4. Average achieved utility over $|M|$ when varying the number of tenants $|M|$ for a constant network capacity

This is due to the fact that with smaller $|RI|$ the advantages of the prediction are less evident, since the negotiation platform behaves similarly to a real time negotiation algorithm ('no prediction'). As a matter of fact, in case $RI = 1$ TTI the two algorithms (i.e. with and without prediction) are identical, since the resource negotiations are done every time slot.

Fig. 4 shows the effects of increasing $|M|$ (and proportionally $|K|$) on the average achieved utility for the 'no prediction' and 'with filter' cases. In this case, $RI = W_P = 25$ TTI and $W_L = 75$ TTI. In this scenario, each tenant serves $|K_m| = 4$ users, thus, the increase in $|M|$ corresponds also to an increase in the network congestion. The results show that the advantages of prediction fade as the network becomes more congested, due to the increase in the non-elastic users. Therefore, to exploit the full potential of prediction, the network capacity should be expanded accordingly to the traffic increase.

D. Business implications of anticipation

To analyze the economical impacts of anticipatory network slicing on the envisioned market model, we propose a comparison in terms of the tenants' willingness to accept a given service quality U_k for a given price p, using the following

TABLE III
EVALUATION OF EQ. (13) WHEN INCREASING THE NUMBER OF TENANTS WHEN CAPACITY IS FIXED WITHOUT PREDICTION

$\lvert M_1\rvert \rightarrow \lvert M_2\rvert$	$(\frac{U_{k,M_1}}{U_{k,M_2}})^{\mu}$	$(\frac{p_{M_1}}{p_{M_2}})^{\epsilon}$	Status
$2 \rightarrow 3$	1.1598	3.1820	YES
$3 \rightarrow 4$	1.1736	1.6810	YES
$4 \rightarrow 5$	1.1901	1.1802	NO

TABLE IV
EVALUATION OF EQ. (13) WHEN INCREASING THE NUMBER OF TENANTS WHEN CAPACITY IS FIXEDWITH FILTER

$\lvert M_1\rvert \rightarrow \lvert M_2\rvert$	$(\frac{U_{k,M_1}}{U_{k,M_2}})^{\mu}$	$(\frac{p_{M_1}}{p_{M_2}})^{\epsilon}$	Status
$2 \rightarrow 3$	1.2555	2.7378	YES
$3 \rightarrow 4$	1.2247	1.6242	YES
$4 \rightarrow 5$	1.1815	1.2001	YES

service acceptance inequality, presented in [9], i.e.

$$\left(\frac{U_{k,M_1}}{U_{k,M_2}}\right)^{\mu} \leq \left(\frac{p_{M_1}}{p_{M_2}}\right)^{\sigma}. \tag{13}$$

Tenants are assumed to accept a given service for a given price if (13) holds. Otherwise, it is assumed that the tenants are not willing to accept the price for the given service, and consequently leave the proposed sharing framework. Table III and Table IV show a comparison between the cases with and without prediction, when $W_P = 25$ TTI. We use 'YES' to indicate the case where (13) holds, and 'NO' for the cases where it does not. By comparing the two tables, one can observe that the application of anticipatory techniques increases the resource efficiency, allowing tenants to achieve higher average utilities with relatively lower costs. As a matter of fact, we can observe that, when exploiting a prediction, the tenants accept the price and enter the market in all cases, whereas without a prediction it decreases to two cases out of three. We can conclude that introducing anticipatory information not only improves performance and efficiency in the resource usage, but indirectly increases the market size, i.e. the number of stakeholders involved, as the infrastructure provider is able to serve a larger number of tenants, while using the same infrastructure.

V. CONCLUSION

In this paper, we explored how we can enhance the efficiency of dynamic network slicing by integrating anticipated users' channel conditions into recurring and frequent negotiations between the tenants and the infrastructure providers. To minimize the impact of inaccurate predictions, we proposed a two-step approach and a novel filtering scheme and showed their effectiveness with simulations. Moreover, numerical results have pointed out the importance of capacity expansion in order to exploit the full potential of anticipatory network slicing. Finally, our analysis has shown that the increased resource efficiency achieved by exploiting prediction allows the infrastructure provider to serve more tenants, while using the same infrastructure.

ACKNOWLEDGEMENT

This work is funded by the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 643002.

The authors would like to thank Ms. Noemi Wagner for copy editing the publication material.

REFERENCES

[1] M. Patzold, "5G readiness on the horizon [mobile radio]," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 6–13, March 2018.

[2] D. Sahinel, C. Akpolat, M. A. Khan, F. Sivrikaya, and S. Albayrak, "Beyond 5G vision for iolite community," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 41–47, January 2017.

[3] P. L. Vo, M. N. H. Nguyen, T. A. Le, and N. H. Tran, "Slicing the edge: Resource allocation for RAN network slicing," *IEEE Wireless Communications Letters*, 2018.

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

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

[6] D. Zhang, Z. Chang, T. Hmlinen, and W. Gao, "A contract-based resource allocation mechanism in wireless virtualized network," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2018, pp. 474–479.

[7] K. Zhu, Z. Cheng, B. Chen, and R. Wang, "Wireless virtualization as a hierarchical combinatorial auction: An illustrative example," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, March 2017, pp. 1–6.

[8] D. Zhang, Z. Chang, T. Hmlinen, and F. R. Yu, "Double auction based multi-flow transmission in software-defined and virtualized wireless networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8390–8404, Dec 2017.

[9] O. U. Akgül, I. Malanchini, and A. Capone, "Dynamic resource trading in sliced mobile networks," *IEEE Transactions on Network and Service Management*, 2019.

[10] N. Bui, M. Cesana, S. A. Hosseini, Q. Liao, I. Malanchini, and J. Widmer, "A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1790–1821, 2017.

[11] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, May 2017, pp. 1–9.

[12] S. Anbazhagan and N. Kumarappan, "Day-ahead deregulated electricity market price forecasting using recurrent neural network," *IEEE Systems Journal*, vol. 7, no. 4, pp. 866–872, Dec 2013.

[13] I. Malanchini and V. Suryaprakash, "Minimizing the impact of prediction errors during anticipatory resource allocation," in *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, June 2018, pp. 1–6.

[14] Gurobi Optimization Inc., "Gurobi optimizer reference manual," 2015. [Online]. Available: http://www.gurobi.com