Anticipatory Allocation of Communication and Computational Resources at the Edge using Spatio-Temporal Dynamics of Mobile Users

Anticipatory Allocation of Communication and Computational Resources at the Edge using Spatio-Temporal Dynamics of Mobile Users

Arcangela Rago, *Graduate Student Member, IEEE*, Giuseppe Piro, *Member, IEEE*, Gennaro Boggia, *Senior Member, IEEE*, and Paolo Dini

***Abstract*—Multi-access Edge Computing represents a key enabling technology for emerging mobile networks. It offers intensive computational resources very close to the end-users, useful for task offloading purposes. Many scientific contributions already proposed approaches for optimally allocating these resources over time. However, most of them fail to take advantage of the prediction of both users' mobility and service demands over a look-ahead temporal horizon. To bridge this gap, this paper formulates a novel methodology for anticipatorily allocating communication and computational resources at the network edge, based on the prediction of spatio-temporal dynamics of mobile users. The conceived architecture exploits a Software-Defined Networking approach to monitor users' mobility, a Convolutional Long Short-Term Memory to predict over different look-ahead horizons the number of users within a given number of cells and their related service demands, and Dynamic Programming to optimally allocate users' requests among available Multi-access Edge Computing servers. Computer simulations investigate the effectiveness of the proposed approach in a realistic autonomous driving use case and compare its behavior against a baseline solution. Obtained results demonstrate its unique ability to dynamically and fairly distribute users' requests among the resources available at the network edge, while ensuring the targeted quality of service level.**

***Index Terms*—ETSI-MEC, Network Optimization, User Mobility, Deep Learning, Dynamic Programming.**

I. Introduction

In both fifth generation (5G) and Beyond 5G (B5G) networks, Multi-access Edge Computing (MEC) is emerging as a fundamental enabling technology for the rapid diffusion of advanced services, such as autonomous driving, virtual/augmented reality, e-Health, robotics, and tactile Internet [1]–[3]. According to European Telecommunication Standard Institute (ETSI)-MEC specifications [4], MEC servers are deployed at the network edge to offer intensive computing and memory capabilities in the proximity of end-users, while guaranteeing low communication latencies to new heavy demanding and real-time services [1]. They are also able to limit network congestions by processing data directly at the edge, instead of forwarding a big amount of data to the cloud. This particularly applies to MEC servers co-located with gNBs (base station of 5G networks), that can provide computational capabilities as close as possible to end-users and capture information for further purposes (data analytics and big data processing) [1].

As expected, communication and computational resources available at the network edge should be properly managed to fulfill the spatio-temporal dynamics and the even growing amount of users' requests [5], [6]. Most of the scientific contributions in this context address network resource management, computational resource allocation, and task offloading through optimization algorithms [7]–[16] or iterative procedures based on artificial intelligence [17]–[22]. Unfortunately, these contributions generally consider the actual static picture of the overall systems and ignore the impact that future spatio-temporal dynamics of mobile users may have on the system behavior. Differently, the knowledge (i.e., prediction) of both users' mobility and communication and computational resources they request over time within a given geographical area could significantly improve network optimization mechanisms [23]–[25]. The current state of the art proposes various instruments to forecast the movements of users [26]–[37], their requests [38]–[44], or both [45] (see Section II for more details). Solutions based on machine and deep learning also promise to better anticipate network behaviors and dynamics in heterogeneous and large scale scenarios [46], [47]. Nevertheless, resulting network optimization problems (including those presented in [26]–[28], [30]–[33], [35], [40]–[43], [45]) fail to take advantage of the joint prediction of both users' mobility and service demands over a look-ahead temporal horizon and within a standard compliant ETSI-MEC context.

To bridge this gap, this work formulates an innovative methodology for the anticipatory allocation of communication and computational resources at the network edge (i.e., task offloading), based on the knowledge of spatio-temporal dynamics of mobile users. The conceived approach significantly extends the very preliminary contributions presented, by the

Arcangela Rago, Giuseppe Piro, and Gennaro Boggia are with the Department of Electrical and Information Engineering (DEI), Politecnico di Bari, 70125 Bari, Italy, and also with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 43124 Parma, Italy (e-mail: arcangela.rago@poliba.it; giuseppe.piro@poliba.it; gennaro.boggia@poliba.it).

Paolo Dini is with the Centre Tecnologic de Telecomunicacions de Catalunya (CTTC/CERCA), 08860 Barcelona, Spain (e-mail: paolo.dini@cttc.es).

This work was supported by the Italian MIUR PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges", by the Apulia Region (Italy) Research project INTENTO (36A49H6), by the Spanish Government under project TEC2017-88373-R (5G-REFINE), by Generalitat de Catalunya under grant 2017 SGR 1195, and by the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 953775 (GREENEDGE). It was also partially supported by the Italian MIUR PON projects Pico&Pro (ARS01_01061), AGREED (ARS01_00254), FURTHER (ARS01_01283), and RAFAEL (ARS01_00305).

same authors of this work, in [48] and [49]. The considered architecture adopts a Software-Defined Networking (SDN) approach to monitor users' mobility over time. Then, starting from the outcomes of the preliminary contributions presented by the same authors in [48] and [49], it exploits a deep learning architecture based on *Convolutional Long Short-Term Memory (ConvLSTM)* [50] for predicting the distribution of users among cells and their related service demands over a look-ahead temporal horizon. A centralized Multi-access Edge Orchestrator uses this information to anticipatorily distribute users' demands among available MEC servers, while satisfying communication and computational constraints at the network edge and the upper bound for latency expected by mobile users. Specifically, the optimal allocation problem is stated as a sequential decision-making process, which considers future steps in the optimization horizon and it is solved by *Dynamic Programming* [51].

The behavior of the proposed approach is investigated in an autonomous driving use case (with real mobility traces [52] and conceivable network and service settings [13], [53]–[59]) by using computer simulations. First of all, the presented study remarks that the usage of both *ConvLSTM* and *Dynamic Programming* ensures results comparable with those obtained by the same optimization algorithm running on a perfect knowledge (i.e., ground truth) of spatio-temporal dynamics of mobile users. This demonstrates the high performance of the prediction process. At the same time, the comparison against a baseline approach, which leverages the distribution of users at the current time instant and allocates users' demands to the closest MEC server, reveals that only the conceived anticipatory approach can fairly distribute users' requests among the resources available at the network edge, while ensuring the targeted quality of service level. Finally, a complexity analysis confirms the effective and easy implementation of the proposed methodology in real deployments.

The remainder of the paper is as follows. Section II reviews the related work on this area and identifies the gaps bridged in this paper. Section III introduces the considered architecture and the targeted scenario. Section IV describes the proposed optimization approach, including the system model, the problem formulation, and the mobility prediction model. Section V presents numerical results coming from computer simulations and formulates a complexity analysis. Finally, Section VI concludes the paper and draws future research directions.

II. State of the Art

Network resource management, computational resource allocation, task offloading, and Virtual Network Function (VNF) placement represent typical technical problems of interest for industry and academia working on mobile communication systems [10], [23], [24]. Very frequently, they are addressed with optimization algorithms willing to minimize energy consumption [7]–[11], delay [12]–[14], or both [15], [16]. Sometimes, a constraint on the maximum allowed delay is taken into account as well [7]–[10], [13].

Emerging methodologies exploit artificial intelligence technologies, like machine learning, deep learning, and deep reinforcement learning, for network optimization [17]. While most of the contributions in this context focus on the optimal management of computational resources only [18]–[20], some other works consider at the same time the goal of managing and allocating communication and computational resources [21], [22]. Available approaches intend to maximize the overall resource capacity [20], to minimize energy consumption [19] and delay [18], [21], [22], as well as to fulfill the expected upper bound for the overall delay [18], [22].

The contributions presented in [23]–[25] highlight that the knowledge (i.e., prediction) of users' mobility and/or the set of requests that they may formulate in a given geographical area over time introduce further key information for network optimization tasks.

The prediction of users' trajectory and location can be achieved with mathematical models [26]–[28]. The mobility forecasting obtained in [26] is used to offload computing tasks (requested by mobile users) to a single remote MEC server. To this end, an optimization problem that jointly minimizes energy consumption and latency, satisfying the expected maximum delay, is formulated in [26]. The knowledge of trajectories during the next look-ahead window is considered in [27] for planning the migration of virtual machines at the network edge. This goal is reached by employing an optimization problem that minimizes communication latencies, ensuring at the same time expected upper bounds. Finally, the work in [28] leverages a Markov Decision Processes to predict user mobility and formulates an iterative approach for jointly allocating communication resources among available users and placing virtual machines at the network edge. Similarly to [26], the presented solution minimizes energy consumption and delay.

Differently from the above-discussed methodologies, solutions based on machine learning promises to better anticipate network behaviors and dynamics, also in heterogeneous and large scale scenarios [46], [47]. For example, the prediction of trajectory and location is performed through deep learning architectures, as Long Short-Term Memorys (LSTMs) [29], [30], [32], [33], LSTMs with attention mechanism [34], Convolutional Neural Networks (CNNs) [31], and a combination of recurrent and CNNs with Markov Chains [35]. Furthermore, the number of users in a given geographical area is predicted through machine learning-based Regressors in [36] and a combination of deep learning and Bayesian networks in [37]. Mobility forecasting in [30] supports an optimization problem willing to distribute computing caching capabilities among mobile users, maximizing the overall resource capacity and satisfying the expected maximum delay. The knowledge of locations, until one [31] or more steps ahead [32], [35], is also adopted to drive the migration of virtual machines at the network edge. In more detail, the contribution in [31] describes an iterative procedure for minimizing the communication latencies and satisfying the expected maximum delays. Optimization problems willing to minimize delay [32] and energy consumption [35] are formulated in [32], [35]. Finally, the work discussed in [33] adopts deep reinforcement learning to manage computation offloading tasks among different remote MEC servers in order to minimize the delay.

This article has been accepted for publication in IEEE Transactions on Network and Service Management. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3099472

TABLE I
COMPARISON AMONG THIS WORK AND THE OTHER CONTRIBUTIONS PERFORMING MOBILITY/REQUESTS PREDICTION AND NETWORK OPTIMIZATION

Work	**Prediction**							**Network optimization**							
	Mobility			**Requests**	**Characterization**		**Look-ahead horizon**	**Communication**	**Computation**			**RRH-BBU map**	**Delay constraint**	**Solution**	
	Trajectory and location		**Number of users per cell**		Spatial	Temporal			**Offloading/ Execution**		**Migration**			Optimization algorithm	Iterative procedure
	Mathematical model	Deep learning	Deep learning	Deep learning					Edge servers	Users					
[26]	✓					✓			1	✓			✓	✓	
[27]	✓					✓	✓				✓		✓	✓	
[28]	✓					✓		✓			✓				✓
[30]		✓				✓				✓			✓	✓	
[31]		✓			✓						✓		✓		✓
[32], [35]		✓				✓	✓				✓			✓	
[33]		✓				✓			>1						✓
[40], [41]				✓		✓	✓		1					✓	
[42]				✓		✓		✓	>1				✓	✓	
[43], [45]				✓	✓	✓						✓		✓	
This			✓	✓	✓	✓	✓	✓	>1				✓	✓	

Instead, traffic volume/load can be accurately predicted through deep learning methods [38], [39], such as Multi-Layer Perceptrons [42], CNNs [44], LSTMs [40], [41], and Multivariate LSTMs [43]. Traffic forecasting during the next look-ahead horizon assists network optimization in terms of computation offloading and resource allocation with one MEC server in [40], [41], minimizing energy consumption. Traffic prediction also aids the joint communication and computational resource allocation for user association and Service Function Chain placement among MEC servers in [42]. Here, an optimization algorithm is adopted for minimizing delay, while respecting service latency as upper bound. Moreover, the knowledge of traffic requests in Cloud-Radio Access Network context supports the Remote Radio Head (RRH)-Base Band Unit (BBU) mapping in [43], where an optimization problem minimizes deployment cost and energy consumption. The traffic volume of RRHs with the number of users moving between a pair of two RRHs is predicted in [45] through Multivariate LSTM. This information is exploited to optimally perform RRH-BBU mapping, minimizing energy consumption and delay.

To conclude, Table I summarizes the goals and methodologies followed by the reviewed scientific contributions performing mobility/requests prediction and network optimization, highlighting the main differences with respect to the approach proposed in this paper. It emerges that to the best of authors' knowledge no contributions in the current state of the art jointly predict, through deep learning, the geographical distribution of users over time (i.e., the number of users available within each cell in a given moment) and the related requests for a look-ahead horizon, as proposed in this work in order to better manage task offloading in a 5G slicing paradigm. Thus, they do not take advantage of mobility and requests prediction to dynamically and anticipatorily optimize communication and computational resource management among available MEC servers, satisfying the upper bound of communication latencies.

III. REFERENCE SCENARIO

This work mainly refers to the task offloading problem, according to which it is necessary to deploy (and properly use) available communication and intensive computational capabilities at the network edge for offering new heavy demanding and latency-critical services with challenging user expectations [1], [4], [49], [54].

The conceived approach can be implemented within the 5G slicing paradigm. In fact, according to 3rd Generation Partnership Project (3GPP) specifications [60], a slice instance represents a set of network functions and related resources which are arranged and configured in a logical network to meet certain network characteristics. To this end, a service provider declares communication service requirements (e.g., coverage area, number and distribution of users, traffic demand, mobility, latency, etc.) to the infrastructure provider. In turn, the infrastructure provider configures the corresponding network slice instance, whose preparation phase includes the on-boarding and verification of network function products and the necessary network environment. From this moment on, the service provider can dynamically allocate the resources belonging to the aforementioned slice to the served mobile users (i.e., the task offloading within a specific slice). Note that in complex deployments, where heterogeneous services are offered through different slices, the proposed approach can be replicated for each slice.

In line with 5G specifications, emerging guidelines for the upcoming B5G systems, and the ETSI-MEC standard [61], the mobile network considered in this work embraces mobile users, gNBs, MEC servers, SDN controllers, and a Multi-access Edge Orchestrator (see Fig. 1). Here, gNBs are part of the 3GPP network integrated within the ETSI-MEC architecture. They provide wireless connectivity to mobile users through heterogeneous technical components at the radio interface [48], [61]. It is important to remark that gNBs can be connected to each other in different ways. Ring, tree, or mesh

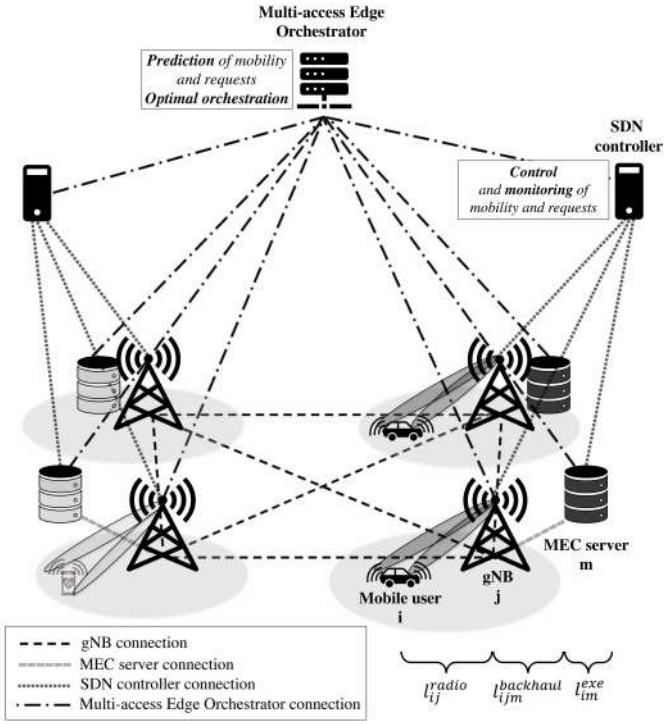

Fig. 1. Reference mobile network with different contributions of latency l_{ijm} in the system model.

topologies can be implemented by the infrastructure provider [62]. Without loss of generality, a mesh topology is depicted in Fig. 1 as an example of the backhaul network topology, even if the system model described in Section IV-A will be general enough for capturing the behavior of any topology.

A number of MEC servers (or MEC hosts) expose resources to mobile users, depending on one or more services they use [61]. In this sense, the example reported in Fig. 1 shows that the black and gray blocks of MEC servers are dedicated to autonomous driving and e-Health services, respectively. According to ETSI-MEC specifications, MEC servers can be deployed at the gNBs, at aggregation points, or at the edge of the core network [4]. Independently from their position, however, MEC resources (i.e., memory and computing) can be used by users attached to different cells. This important flexibility, however, requires a careful distribution of users' demands, that should take care of the stringent communication requirements, instead of just considering the computational capabilities of MEC servers.

Network resources are monitored, configured, and orchestrated [61]. To this end, SDN controllers continuously interact with gNBs and MEC servers for monitoring the number of users served by each cell, the computational resources they request, and the amount of resources exposed and/or available in each MEC server. Note that SDN controllers can retrieve useful information from network elements through standardized protocols (i.e., OpenFlow, RESTCONF, etc.) [63]. Specifically, since gNBs know how many users are attached to them, they can retrieve information about the number of users served by each gBN by simply asking for such information to the gNBs.

This information is delivered to the Multi-access Edge Orchestrator for network optimization purposes. It represents a fundamental entity of the ETSI-MEC reference architecture, included in the MEC system level management [61]. The envisaged solution uses Multi-access Edge Orchestrator capabilities for managing a certain number of gNBs and MEC servers in a given geographical area (i.e., the radio access network is divided into clusters, controlled by one orchestrator) in order to optimally allocate computing and communication resources for task offloading, based on the prediction of spatio-temporal users' dynamics. This is done by satisfying heterogeneous traffic demands. The proposed optimization algorithm, which can be aided by exploiting mobility and service requests prediction, is executed by each orchestrator instance in order to minimize the latency (which is one of the most leading performance measures of 5G and B5G [5], [6]) of each service, while jointly considering network communication and computational requirements and satisfying the upper bound of service latency and related network constraints.

Moreover, an intrinsic characteristic of many 5G services (e.g., autonomous driving, virtual/augmented reality assisting museum tours) is mobility. Therefore, the communication and computational resources must be managed by using a mobility-aware approach, which is considered one of the most critical and challenging issues for network orchestration [6], [33].

IV. Problem Statement

In this section, the system model is described and the optimization problem for the reference scenario and the adopted mobility prediction model are formulated. To facilitate the understanding of the notations adopted in what follows, a summary of symbols is reported in Table II.

A. System model

Let $\mathcal{I}$ and $|\mathcal{I}|$ be the set and the number of users moving in the considered geographical area, respectively. According to the target application, the request formulated by the $i-th$ user is characterized by the following communication and computational requirements: the communication bandwidth set to b_i, the upper bound of latency equal to τ_i, the input data size s_i, the memory requirement set to m_i, and the demanded computational capability (expressed in terms of number of CPU cycles) equal to c_i. Let $\mathcal{J}$ be the set of available gNBs. The number of gNBs is given by $|\mathcal{J}|$, that is the cardinality of $\mathcal{J}$, and B_j represents the amount of bandwidth available within the cell served by the $j-th$ gNB. Let $\mathcal{M}$ and $|\mathcal{M}|$ be the set and the number of the available MEC servers, respectively. M_m and F_m indicate the memory capability and computing ability (expressed in terms of CPU cycles per second) of the $m-th$ MEC server. I_j^{gNB} is the set of users in the $j-th$ cell, attached to the $j-th$ gNB, and I_m^{MEC} is the set of users served by the $m-th$ MEC server.

The system evolves in discrete-time intervals based on the user mobility: every t_k, a different number of users I_j^{gNB} and I_m^{MEC} is served by the $j-th$ gNB and the $m-th$ MEC

TABLE II
LIST OF ADOPTED MATHEMATICAL NOTATION

Symbol	Description
i	Index of the $i-th$ user
j	Index of the $j-th$ gNB
m	Index of the $m-th$ MEC server
t_k	Discrete-time interval
$\mathcal{I}(k)$	Set of users
$\mathcal{J}$	Set of available gNBs and related attached cells
$\mathcal{M}$	Set of available MEC servers
$I_j^{gNB}(t_k)$	Set of users attached to the $j-th$ gNB
$I_m^{MEC}(t_k)$	Set of users served by the $m-th$ MEC server
$\|set\|$	Set cardinality
$\|\hat{I_j}^{gNB}(t_k)\|$	Predicted number of users attached to the $j-th$ gNB
b_i	Communication bandwidth of the $i-th$ user
τ_i	Upper bound of latency for the service requested by the $i-th$ user
s_i	Input data size of the $i-th$ user
m_i	Memory requirement of the $i-th$ user
c_i	Computational capability requirement (in CPU cycles) of the $i-th$ user
B_j	Available bandwidth within the cell attached to the $j-th$ gNB
$e_{ij}(t_k)$	Spectral efficiency between the $i-th$ user and the $j-th$ gNB
$M_m(t_k)$	Memory capability of the $m-th$ MEC server
$M_m^{opt}(t_k)$	Memory capability of the m–th MEC server in the optimization problem
$M_m^{cons}(t_k)$	Consumed memory by the $m-th$ MEC server
$F_m(t_k)$	Computing ability (in CPU cycles/second) of the $m-th$ MEC server
$F_m^{opt}(k)$	Computing ability (in CPU cycles/second) of the $m-th$ MEC server in the optimization problem
$l_{ijm}(t_k)$	Total latency experienced by the $i-th$ user attached to the j–th gNB and served by the $m-th$ MEC server
$\bar{l}_{ijm}(t_k)$	Average latency per user
$l_{ij}^{radio}(t_k)$	Communication latency experienced between the $i-th$ user and the $j-th$ gNB over the radio interface
$l_{ijm}^{backhaul}(t_k)$	Backhaul latency between the $j-th$ gNB and the $m-th$ MEC server for the $i-th$ user
$l_{im}^{exe}(t_k)$	Execution latency experienced at the $m-th$ MEC server for serving the $i-th$ user
$I_{jm}(t_k)$	Portion of users attached to the $j-th$ gNB and served by the $m-th$ MEC server
$r_{jm}(t_k)$	Capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server
$f_{im}(t_k)$	Number of CPU cycles/second allocated by the $m-th$ MEC server to the $i-th$ user
$\alpha_{im}(t_k)$	Binary decision variable denoting if the $i-th$ user is served by the $m-th$ MEC server
T	Observation window
N	Look-ahead temporal horizon
$t_{k,n}$	Decision cycle
k	Index of the decision epoch t_k
γ	Discount factor
n	Index of the considered time steps in each decision epoch t_k

server, respectively. For every time interval t_k, it holds that $|\mathcal{I}(t_k)| = \sum_{j\in\mathcal{J}} |I_j^{gNB}(t_k)| = \sum_{m\in\mathcal{M}} |I_m^{MEC}(t_k)|$.

The total latency experienced by the $i-th$ user attached to the $j-th$ gNB and served by the $m-th$ MEC server in the $k-th$ time interval is given by:

$$l_{ijm}(t_k) = l_{ij}^{radio}(t_k) + l_{ijm}^{backhaul}(t_k) + l_{im}^{exe}(t_k), \quad (1)$$

where $l_{ij}^{radio}(t_k)$ is the communication latency experienced between the $i-th$ user and the $j-th$ gNB over the radio interface, $l_{jm}^{backhaul}(t_k)$ is the backhaul latency experienced between the $j-th$ gNB and the $m-th$ MEC server, and $l_{im}^{exe}(t_k)$ is the execution latency experienced at the $m-th$ MEC server [6], [9], [54]. These different latency contributions are shown in Fig. 1.

In compliance with ITU specifications, the communication latency over the radio interface, $l_{ij}^{radio}(t_k)$, is expected to be less than 5 ms [53], [54].

The backhaul latency $l_{ijm}^{backhaul}(t_k)$ is obtained by dividing the aggregate traffic load generated by the users attached to the $j-th$ gNB and served by the $m-th$ MEC server, that is $\sum_{i\in\mathcal{I}_{jm}(t_k)} s_i$, and the capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server, $r_{jm}(t_k)$ [42]:

$$l_{ijm}^{backhaul}(t_k) = \frac{\sum_{i\in I_{jm}(t_k)} s_i}{r_{jm}(t_k)}, \quad (2)$$

where $I_{jm}(t_k)$ is the portion of users attached to the $j-th$ gNB and served by the m–th MEC server, that share the same backhaul link. The system model described herein is general enough for capturing the behavior of any backhaul topology. Without loss of generality, a mesh topology, with the same capacity for each backhaul link, is considered (see Fig. 1). MEC servers can be deployed at the gNBs, at aggregation points, or at the edge of the core network. Therefore, the backhaul latency varies depending on the scenario. Specifically, assuming that MEC servers are co-located with gNBs without loss of generality, there are two possibilities when calculating the backhaul latency. No additional delay (i.e., $l_{jm}^{backhaul}(t_k) = 0$) is introduced in the backhaul if the $m-th$ MEC server co-located with the $j-th$ gNB (i.e, $m = j$), to which it is attached the user, is the one serving the user. Conversely, the backhaul latency is considered and calculated for the backhaul path connecting the gNB, to which it is attached the user, with a neighboring MEC host, which is serving the user.

During the time interval t_k, the computing capabilities exposed by each MEC host are assumed to be uniformly allocated among served users. Therefore, the execution latency $l_{im}^{exe}(t_k)$ is equal to [9], [14]:

$$l_{im}^{exe}(t_k) = \frac{c_i}{f_{im}(t_k)} = \frac{c_i}{F_m(t_k)/|I_m^{MEC}(t_k)|}, \quad (3)$$

where $f_{im}(t_k)$ is the number of CPU cycles per second allocated by the $m-th$ MEC server to the $i-th$ user. Such an equation is generic enough to be used in any realistic scenario with homogenous and heterogeneous service requirements: the execution latency refers to the computational capability requirements of users, that can execute a single application task, as well as more heterogeneous application tasks.

B. Optimization problem

The goal of this paper is to distribute users' requests among the available MEC servers, so that the latency of each

$$P1: \min_{\{\alpha_{im}(t_k)\}} \left\{ \sum_{n=0}^{N} \gamma^n \left[\sum_{j \in \mathcal{J}} \sum_{i \in I_j^{gNB}(t_{k,n})} \left(l_{ij}^{radio}(t_{k,n}) + \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) \cdot \left[l_{ijm}^{backhaul}(t_{k,n}) + l_{im}^{exe}(t_{k,n}) \right] \right) \right] \right\}, \forall t_k \tag{4}$$

subject to:

$$\sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) \cdot m_i \leq M_m^{opt}(t_{k,n}), \forall m \in \mathcal{M}, \forall n, \sum_{i \in \mathcal{I}(t_{k,n})} m_i \leq \sum_{m \in \mathcal{M}} M_m^{opt}(t_{k,n}), \forall n \tag{4a}$$

$$\sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) \cdot f_{im}(t_{k,n}) \leq F_m^{opt}(t_{k,n}), \forall m \in \mathcal{M}, \forall n \tag{4b}$$

$$l_{ij}^{radio}(t_{k,n}) + \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) \cdot \left[l_{ijm}^{backhaul}(t_{k,n}) + l_{im}^{exe}(t_{k,n}) \right] \leq \tau_i, \forall i \in \mathcal{I}(t_{k,n}), \forall n \tag{4c}$$

$$B_j \cdot e_{ij}(t_{k,n}) \geq b_i, \forall i \in I_j^{gNB}(t_{k,n}), \forall n \tag{4d}$$

$$\alpha_{im}(t_{k,n}) \in \{0,1\}, \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) = 1, \forall i \in \mathcal{I}(t_{k,n}), \sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) = |I_m^{MEC}(t_{k,n})|, \forall m \in \mathcal{M}, \forall n \tag{4e}$$

$$|I_j^{gNB}(t_{k,n})| = \sum_{m \in \mathcal{M}} |I_{jm}(t_{k,n})|, \forall j \in \mathcal{J}, |I_m^{MEC}(t_{k,n})| = \sum_{j \in \mathcal{J}} |I_{jm}(t_{k,n})|, \forall m \in \mathcal{M}, \forall n \tag{4f}$$

considered service is minimized and network outage in terms of memory, computing, and bandwidth resources is avoided. Such a problem is stated as a sequential decision-making process: at every decision epoch t_k, control actions aiming at assigning users' demands to the best suitable MEC servers are executed, according to their available memory capabilities and computing abilities, in order to minimize latencies experienced by users and to satisfy service latency constraint. At every decision epoch t_k, the requests and, hence, the resources needed to run the user services for the N steps ahead are leveraged and the control is executed based on the optimization problem $P1$ stated in (4). The solution of the problem is found by executing the dynamic programming approach [51] at every decision epoch t_k (i.e., each point of the sequential decision-making process where decisions are made), transforming a complex problem into a sequence of simpler problems. In line with the dynamic programming approach [51], the discount factor γ, $(0 < \gamma \leq 1)$, is introduced to incorporate the concept of discounting for the look-ahead temporal horizon N. Specifically, the decision cycle $t_{k,n}$, with $n \in \{0, 1, ..., N\}$, represents the sequence of the considered time steps (with $t_{k,n} = t_{k+n}$) to reach and implement decisions in each epoch t_k, whose impact is exponentially weighted through γ^n. Thus, it is possible to understand that, when $n = 0$, $t_{k,0}$ is weighted through $\gamma^0 = 1$, while the future time steps in the sequence have a gradually decreasing weight (i.e., from γ^1 for $t_{k,1}$ to γ^N for $t_{k,N}$) in the decision cycle $t_{k,n}$.

The implemented control is expressed by a binary decision variable $\alpha_{im}(t_k)$, that is:

$$\alpha_{im}(t_k) = \begin{cases} 1 & \text{if the } i-th \text{ user is served by the } m-th \\ & \text{MEC server, i.e., } \forall i \in I_m^{MEC}(t_k); \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Note that $\alpha_{im}(t_k)$ only involves the backhaul and the execution latency because they depend on the concerned MEC server, while the radio component is independent thereof. The constraints in (4a) consider the memory capabilities and requirements: the memory capability of the $m-th$ MEC server $M_m^{opt}(t_{k,n})$ cannot be exceeded by served users in each decision cycle $t_{k,n}$ and the overall memory capabilities need to be sufficient for satisfying memory requirements. The constraint in (4b) regards the CPU ability of the $m-th$ MEC server $F_m^{opt}(t_{k,n})$ in each decision cycle $t_{k,n}$. Because of the definition of the execution latency component in (3), computing capabilities are included in the service latency constraint (4c), that is valid for each $i-th$ user in the network , where the maximum tolerable latency τ_i is the upper bound of user latency experienced during each decision cycle $t_{k,n}$. If the computing abilities are not enough, (4c) is not verified. Bandwidth requirements are considered in (4d), where $e_{ij}(t_{k,n})$ is the spectral efficiency between the $i-th$ user and the $j-th$ gNB. Moreover, in every decision cycle $t_{k,n}$ each user can be served by one and only one MEC server, as reported in (4e) and (4f), that means the number of users attached to different gNBs should be equal to the number of users served by different MEC hosts.

The solution of the optimization problem $P1$ stated in (4) may be anticipatorily found, forecasting the number of users in the coverage area of each gNB. In what follows, the anticipatory optimization approach presented in this work is referred to as *Prediction-based Control (P-C)*. Since the solution of the network optimization problem $P1$ stated in (4) may be also found supposing to know the mobility of users in advance, in Section V also the anticipatory network optimization approach based on ground truth, i.e., *Ground Truth-based Control (GT-C)*, is evaluated.

C. Mobility prediction model

The users moving in the considered geographical area may pass from one cell to an adjacent cell. Accordingly, the number of users attached to each gNB changes over time. The goal of the mobility prediction model described in this subsection is to anticipatorily discover the distribution of mobile users

This article has been accepted for publication in IEEE Transactions on Network and Service Management. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3099472

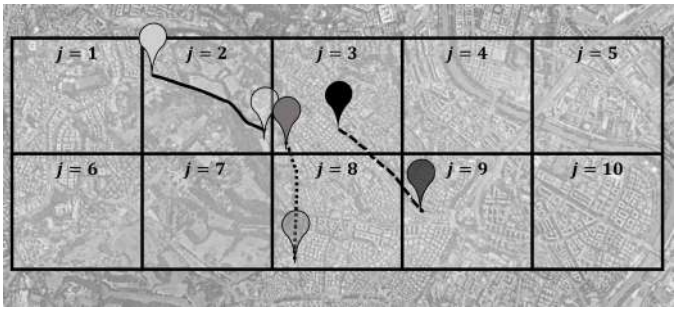

Fig. 2. The considered geographical area with example temporal movements of three taxi cabs.

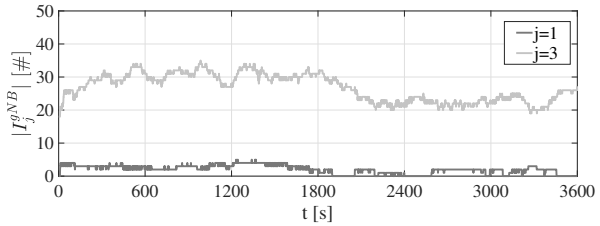

Fig. 3. Distribution of the number of users over time for two cells.

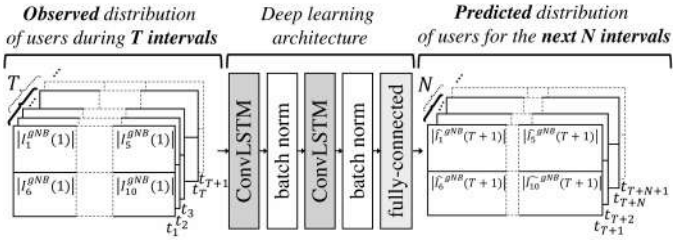

Fig. 4. The conceived mobility prediction model.

among available cells, based on the knowledge of the number of users attached to related gNBs in the past. Service demands can be later estimated based on the services requested by the users.

To this aim, this paper leverages real mobility data from the dataset presented in [52], which reports the movements of around 100 taxi cabs in Rome (Italy), from 1 February 2014 to 2 March 2014, with a granularity of about 15 s. The traces of the published version of the dataset gives information on when the taxi position has been collected, with a precision of microseconds, and Global Positioning System (GPS) coordinates, in the decimal format. The considered geographical area of the center of Rome has been divided using square cells, covering an area of 1 km $\times$ 1 km each (an example is reported in Fig. 2). Note that square cells have been considered, but the considerations are also valid for arbitrarily shaped cells.

These real traces are used to generate a list of matrices describing the geographical distribution of users over time. For example, Fig. 3 shows the distribution of the number of users for two cells (i.e., $j = 1$ and $j = 3$) with a low and high number of users, respectively.

The conceived mobility prediction model exploits the ConvLSTM architecture to predict the distribution of mobile users among the available cells and for the upcoming N consecutive time intervals, based on the knowledge of the distribution of users (i.e., retrieved by SDN controllers from gNBs, that know how many users are attached to them) observed during the latest T observation time intervals. As depicted in Fig. 4, the considered ConvLSTM architecture is based on LSTM [64], with the convolution operator as input, forget, and output gates instead of the element-wise or Hadamard product [50]. Therefore, it is able to extract temporal and spatial correlations of data through LSTM memory cells and the convolutional operation, respectively [38], [65]. More specifically, this work conceives a learning architecture embracing two 2-dimensional ConvLSTM layers, after each one a batch normalization layer is used to accelerate deep network training [66]. The number of epochs and the number of filters are set equal to 30 (see the convergence analysis proposed in Section V-A) and 200, respectively. At the end, there is a fully-connected layer with the Rectified Linear Unit (ReLU) activation function [38] to predict the expected distribution of users, after the observation window T, for a specific look-ahead temporal horizon N. The predictor is configured in order to minimize the Mean Square Error (MSE) loss function, which minimizes the difference between the ground truth and the predicted distribution of users [48], [65]. The Adam optimization [67], with a learning rate equal to 0.001, is used to iteratively update the network weights.

V. Performance Evaluation

Herein, the performance of the conceived anticipatory network optimization scheme is evaluated by using computer simulations. Without loss of generality, the study considers an autonomous driving use case (with real mobility traces [52] described in Section IV-C and conceivable network and service settings [13], [53]–[59]). Of course, the whole approach can be applied to each use case and heterogeneous scenarios by properly adapting the related parameter settings.

A real geographical area of 10 km^2 in Rome (Italy) is considered, divided into 10 square cells (i.e., $|\mathcal{J}| = 10$). According to the autonomous driving use case, for the $i-th$ user the communication bandwidth and the upper bound of service latency are set to $b_i = 700$ Mbps and $\tau_i = 100$ ms, respectively [54], the input data size is set to $s_i = 5$ Mbit [13], [56], and the memory and computational capability requirements are set to $m_i = 16$ GB [57] and $c_i = 300$ Megacycles [56], respectively. The available bandwidth within the cell attached to the $j-th$ gNB and the capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server are set to $B_j = 40$ MHz [58] and $r_{jm} = 10$ Gbps, respectively. Since it is assumed that MEC servers are co-located with gNBs without loss of generality, $|\mathcal{J}| = |\mathcal{M}| = 10$ in the tests. The parameters of MEC servers, whose sizing is a key issue in such systems, are adequately sized with respect to overall requests in each $t_{k,n}$ [59]: the memory capability and the computing ability of the $m-th$ MEC server are set to $M_m^{opt} = 176$ GB and $F_m^{opt} = 36$ Gigacycles/s, respectively. In simulations, it is considered the upper bound of the communication latency experienced over the radio

TABLE III
MAIN SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
Area	10 km^2	$\lvert\mathcal{J}\rvert = \lvert\mathcal{M}\rvert$	10
T	40 s	N	5 s, 10 s, 20 s, 40 s
Learning rate	0.001	γ	0.9
b_i	700 Mbps [54]	τ_i	100 ms [54]
s_i	5 Mbit [13], [56]	m_i	16 GB [57]
c_i	300 Megacycles [56]	B_j	40 MHz [58]
r_{jm}	10 Gbps	M_m^{opt}	176 GB
F_m^{opt}	36 Gigacycles/s	l_{ij}^{radio}	5 ms [53], [54]
e_{ij}	30 bit/s/Hz [55]	Period of time	3600 s

interface $l_{ij}^{radio}(t_{k,n})$, that means to use constant values for l_{ij}^{radio} (i.e., 5 ms [53], [54]) and the spectral efficiency e_{ij} (i.e., 30 bit/s/Hz [55]) in each $t_{k,n}$. Table III summarizes the main adopted parameters.

In line with the dynamic programming approach [51], the optimal solution has been found at each decision epoch t_k through the value iteration algorithm implemented by using Matlab. As anticipated in Section IV-B, the optimization problem $P1$ is solved by considering the predicted number of users per cell over a specific temporal horizon, i.e., $|\hat{I}_j^{gNB}(t_{k,n})|$ by using the actual distribution of users for $n = 0$ and the prediction of the number of users for the future time steps in the decision cycle $t_{k,n}$, and the perfect knowledge (ground truth) of users' distributions, i.e., $|I_j^{gNB}(t_{k,n})|$. These anticipatory mechanisms based on prediction and ground truth are denoted with *P-C* and *GT-C*, respectively. Note that the comparison between P-C and GT-C intends to highlight the effectiveness of the prediction procedure and its impact on the overall system performance. The behavior of both P-C and GT-C is studied for different temporal horizons, that are $N = 5$ s, 10 s, 20 s, 40 s, to evaluate their effect on key performance indicators. Moreover, to provide further insight, the anticipatory methods P-C and GT-C are compared with a *Baseline* approach. It just leverages the distribution of users at the current time instant t_k and allocates their requests to the closest MEC server (i.e., co-located with the gNB in the related cell), without envisaging optimization problems and constraints. Therefore, since the related literature is missing works that perform network resource optimization based on the prediction of the number of users problem (please see Section II for further details), the proposed anticipatory optimization approach based on mobility prediction is compared with the anticipatory network optimization approach based on ground truth and with the Baseline approach defined above.

Since the system configuration decision and the user latency are updated with a time granularity of 1 s, the latency constraint (4c) is continuously taken into account by P-C and GT-C for considering the impact of users' mobility, handovers, and possible service migrations among MEC servers. As a consequence, the proposed system model and the conceived optimization problem allow to successfully meet the whole service latency constraint. Note that the developed approach has been conceived for 5G and B5G networks. Indeed, it is possible to assume a 0 ms handover latency (namely mobility interruption time in 3GPP specifications) [68]. At the same time, this assumption does not influence the behavior of the proposed approach because resources are optimally allocated on a much higher time granularity than the mobility interruption time allowed in 5G (and beyond) deployments. For the same reason, the presented approach does not explicitly consider the virtual machine/container migration process. With a time granularity of 1 s, the Multi-access Edge Orchestrator, that optimally orchestrates requested services and available communication and computational resources, communicates with all the network entities. Therefore, any configuration changes (i.e., on the number of users and related resources to be allocated) are known by MEC servers through the interaction with the orchestrator. Moreover, the delay of task migration between MEC servers can be considered negligible in vehicular context [69], as the analyzed case, or, through prediction information for the look-ahead temporal horizon N, the migration process can eventually occur before it actually happens so that the users do not experience any additional delay due to migration [70].

The measured key performance indicators entail a complete analysis on mobility prediction performance and latency per user, as well as the number of changes among MEC servers, the distribution of users among MEC servers, consumed memory, and CPU usage. The number of changes among MEC servers is included as key performance metric because, in mobile scenarios, it is important to guarantee service continuity. The changes among MEC servers (and so also the number of potential migrations) imply the establishment of new backhaul connections, having negative effects on experienced latency. Also the backhaul connection quality affects the computation execution [3]. Therefore, it is better to avoid changes among MEC servers and keep connectivity between the user and the serving MEC host [3], [71]. Finally, also the complexity of the proposed anticipatory network optimization scheme is evaluated.

All the results reported below have been evaluated in a period of time (i.e., decision epochs) equal to 3600 s, with a time granularity of 1 s, and have been obtained by averaging the outcomes on the 3600 realizations. Together with average values, the $95\%-$confidence intervals, computed through the Gaussian statistical distribution, are reported as well for the spatial characterization. For the characterization during $3600s$, the Cumulative Distribution Functions (CDFs) illustrate only P-C and Baseline, because GT-C trends overlap with P-C and they are omitted for the sake of clarity.

A. Mobility prediction performance

Regarding the prediction procedure (integrated within the P-C scheme), the conceived mobility prediction model exploits the ConvLSTM architecture, as described in Section IV-C. To provide further insight, the comparison with a state-of-the-art mobility prediction approach, that uses the LSTM architecture aided by the attention mechanism for capturing long-range dependency [34], is presented as well. In particular, the reference learning architectures selected for the cross-comparison

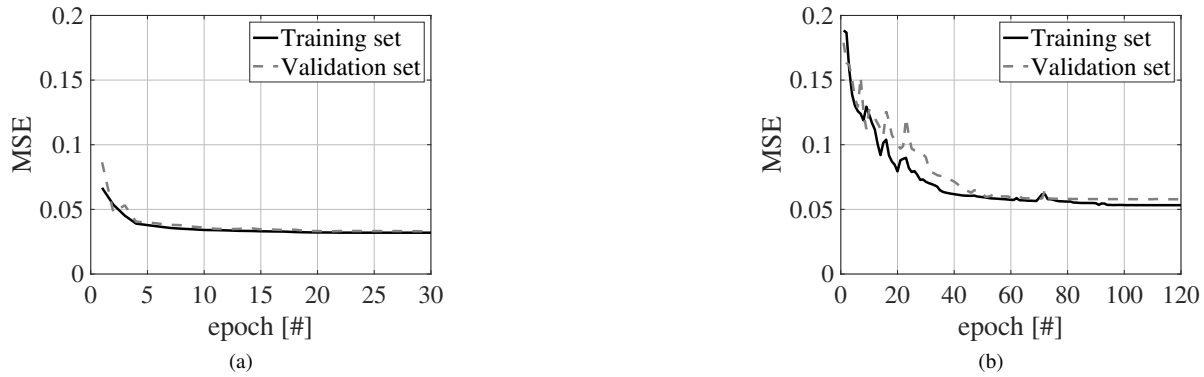

Fig. 5. Prediction loss (i.e., MSE) vs number of epochs for a) ConvLSTM architecture and b) LSTM architecture with attention.

TABLE IV
COMPLEXITY ANALYSIS OF LEARNING ARCHITECTURES

Architecture	# Parameters			
	N=5s	N=10s	N=20s	N=40s
ConvLSTM	801205	802210	804220	808240
LSTM with attention	799940	801850	804030	808140

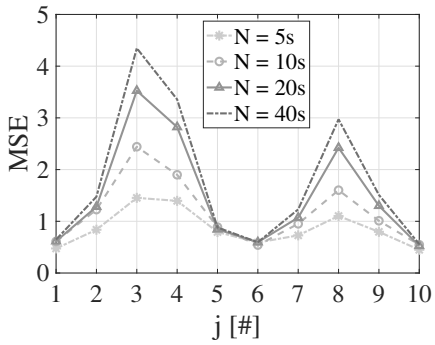

Fig. 6. MSE for each $j-th$ cell, registered with different N.

embrace four LSTM layers (i.e., two with 200 and 2 with 100 hidden units, respectively, after each one a batch normalization layer is used) in order to have a comparable complexity of the corresponding ConvLSTM architecture. Note that the mobility prediction architecture needs a different number of training parameters for each considered temporal horizon N, as summarized in Table IV.

Fig. 5 shows the prediction loss (i.e., MSE) of the ConvLSTM architecture and the LSTM architecture with attention as a function of the number of epochs for the training set and the validation set, representing 80% and 20% of the adopted dataset, respectively. The reported curves confirm that the developed ConvLSTM architecture reaches lower values of MSE with respect to the LSTM architecture with attention. Moreover, differently from the LSTM architecture with attention, the ConvLSTM architecture fastly converges to stable values and does not need a long training period. Accordingly, the ConvLSTM architecture trained during 30 learning epochs, which achieves better results in terms of prediction loss and convergence time /complexity, is considered hereafter.

To deeply evaluate the mobility prediction performance in the investigated scenario, Fig. 6 reports the MSE values for each $j-th$ cell, registered with the different temporal horizons N. Cells $j = 3$ and $j = 8$, with the highest number of users (see the trend of $|I_3^{9NB}|$ in Fig. 3), reach the greatest values of prediction loss. Moreover, the MSE value tends to increase with N, as expected. In fact, the highest values of MSE are registered for $N = 40$ s, even if in this case MSE is generally lower than 4.

B. Latency per user

Fig. 7 depicts the average latency per user served by each MEC server. The average latency per user registered by both P-C and GT-C schemes is always lower than the maximum tolerable latency τ_i. Furthermore, the proposed optimization approaches, involving a load balancing among MEC servers, keep a nearly stable and uniform latency throughout the network. Thus, they can improve the computation efficiency of MEC servers, avoiding overloaded MEC servers, as well as user experience, balancing MEC servers loads and always satisfying service latency requirements [6], [59].

On the contrary, without the previous implications, the Baseline scheme registers an average latency per user that exceeds the maximum tolerable latency τ_i in high-loaded cells.

The CDFs of all the latencies per user reported in Fig. 8 thoroughly confirm that only P-C always ensures service latency requirements for all the users in the network, differently from the Baseline approach. To deeply analyze the impact of the horizon N, the values of the average latency per user among MEC servers for P-C with each analyzed horizon are reported. They are equal to 83.2 ms for $N = 5$s, 81.1 ms for $N = 10$ s, 82.7 ms for $N = 20$ s, and 83.6 ms for $N = 40$ s. Obtained results clearly reveal that from $N = 5$ s to $N = 10$ s the average latency per user among different MEC servers decreases, while it tends to increase with values higher than $N = 10$ s, registering the highest value of latency for $N = 40$ s.

To conclude, $N = 10s$ is a suitable optimization horizon because of slightly lower values of latency.

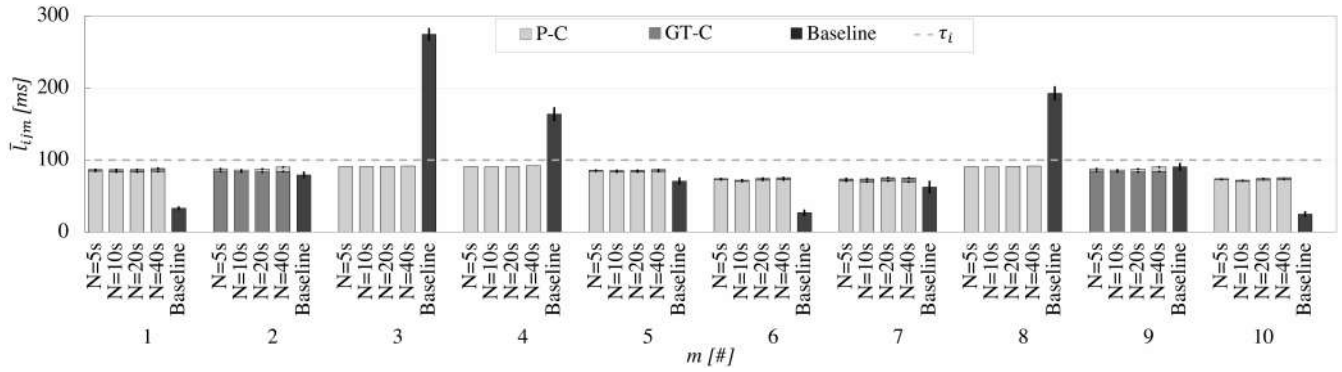

Fig. 7. Average latency per user (with the $95\%-$confidence intervals) served by each MEC server for P-C, GT-C, and Baseline.

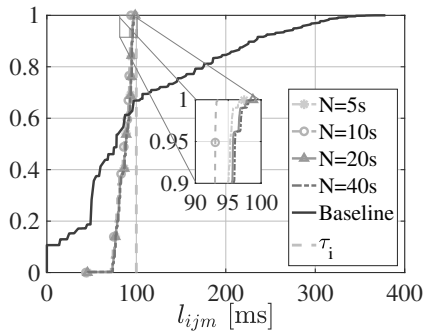

Fig. 8. CDF of the latency per user for P-C and Baseline.

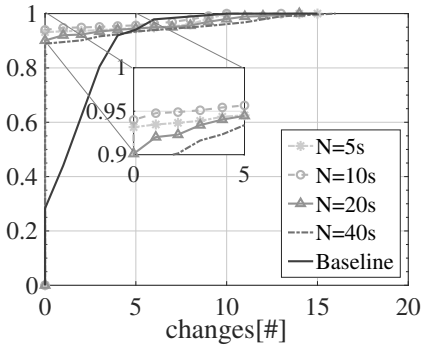

Fig. 9. CDF of the number of changes among MEC servers for P-C and Baseline.

C. Changes among MEC servers

Fig. 9 shows the CDFs of the number of changes among MEC servers. Reported curves demonstrate that the presented proposal generally has the highest performance levels: a number of changes equal to 0 is registered by only 28.31% of realizations for the Baseline scheme, whereas the proposed approach presents around 90% of samples with 0 changes. Focusing on the horizon N, from $N = 5$ s to $N = 10$ s the performance improves, while it tends to decrease with values higher than $N = 10$ s. In fact, when $N = 10$ s, the number of changes is always lower and the value of the $95-th$ percentile is 4 changes with respect to 7, 5, and 9 changes registered for $N = 5$ s, $N = 20$ s, and $N = 40$ s, respectively. Thus, increasing the considered future steps (i.e., from $N = 5$ s to $N = 10$ s) in the optimization problem $P1$ reduces the number of changes among MEC servers. However, because of higher variability, for longer temporal horizons (i.e, $N = 20$ s and $N = 40$ s) the anticipatory network optimization approach starts to imply a higher number of changes among MEC servers and the highest number is reached with $N = 40$ s. The $95-th$ percentile of the presented approach with $N = 10$ s outperforms also Baseline (i.e., 5 changes).

In summary, this analysis further confirms that $N = 10$ s is a suitable optimization horizon since it minimizes the average user latency and the number of changes among MEC servers.

D. Distribution of users among MEC servers

Fig. 10 shows the average number of users served by each MEC server. Both anticipatory network optimization methods (P-C and GT-C) are able to fairly distributed users' demands among the different MEC servers, regardless of the gNB to which they are co-located. Moreover, since the ConvLSTM architecture has very high prediction performance, P-C behaves very similarly to GT-C. They have exactly the same behaviors for MEC servers co-located with gNBs having a high number of users (e.g., $j = 3$), that are fully used under memory and computing constraints. Also by varying the temporal horizon N, P-C and GT-C achieve a very similar average number of users served by each MEC host.

Instead, the Baseline approach is deeply biased by the distribution of the users among cells and, in particular, its policy is to maintain the users at the MEC server co-located with the gNB in which they are attached.

Fig. 11 illustrates the CDFs of the number of users served by different MEC servers. It further confirms the extremely high similarity between the trends of P-C with different N, that behaves differently from the Baseline scheme having higher variability.

E. Amount of memory consumed by MEC servers

Fig. 12 represents the average values of the memory consumed by each MEC server. Also in this case it is possible to observe that both P-C and GT-C methods, with different horizons N, well balance the load among the MEC servers. This result confirms the fairness property investigated in the previous subsection. In high-loaded cells, the MEC servers

This article has been accepted for publication in IEEE Transactions on Network and Service Management. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3099472

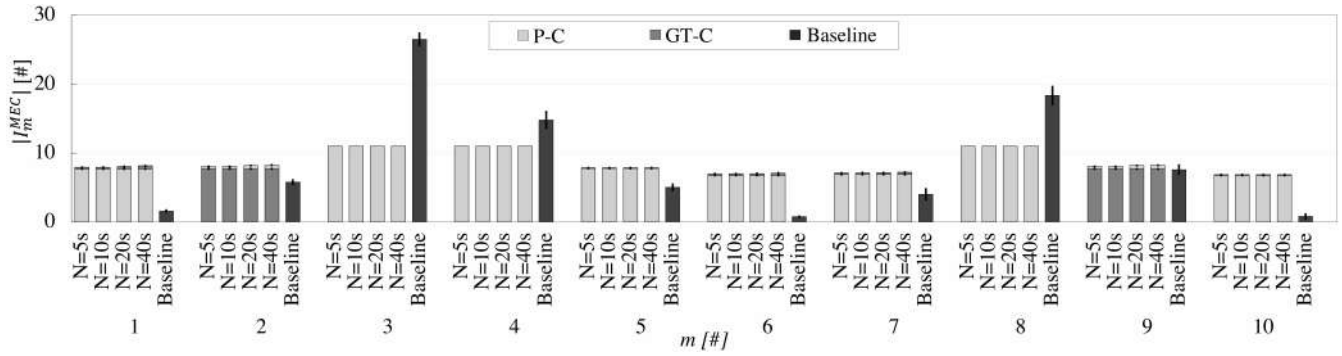

Fig. 10. Average number of users (with the $95\%-$confidence intervals) served by each MEC server for P-C, GT-C, and Baseline.

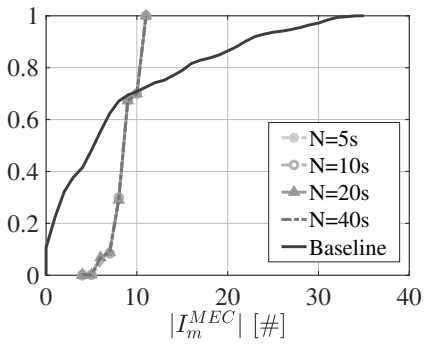

Fig. 11. CDF of the number of users served by MEC servers for P-C and Baseline.

co-located with the gNBs saturate their available memory. As a consequence, the proposed approaches redirect some of the requests generated within these cells towards other MEC servers, thus always satisfying the constraint reported in (4a).

Instead, the consumed memory M_m^{cons} for Baseline, without memory constraints, reaches an average value of around 400 GB for MEC servers corresponding to cells with a high number of users (e.g., $m = j = 3$), as demonstrated by the reported results.

As an additional confirmation, the CDFs reported in Fig. 13 describe how the Baseline approach registers peak usage of memory equal to around 600 GB. On the contrary, the anticipatory optimization scheme developed in this work guarantees quite balancing of the amount of memory consumed in the available MEC servers, which is always below the target upper bound.

F. CPU usage of MEC servers

According to the definition in (3) and the related constraint (4b) of the formulated optimization problem $P1$, the CPU capability of each MEC server is completely consumed in all the implemented approaches. Note that the computing ability of the $m-th$ MEC server in the optimization problem $P1$, i.e., $F_m^{opt} = 36$ Gigacycles/s, is adequately set with respect to overall requests and it is lower compared to typical values of the MEC server ability. In fact, they can be greater than 1000 Gigagycles/s [72] and, with that assumption, the vast majority of the available CPU resources could be dedicated to other services and purposes.

Of course, the CPU ability of MEC server affects the execution latency experienced by each user, which is the most significant component. In fact, because of the computing sizing of MEC servers (i.e., F_m^{opt}), the average total latency per user performed through the optimization approach is generally closer to the maximum tolerable latency τ_i (as detailed in the next subsection) and it validates the current hypothesis in considering the radio component as constant. Without load balancing among MEC servers, the same computing abilities (i.e., $F_m = F_m^{opt}$) are not sufficient to always satisfy the upper bound of service latency τ_i in the Baseline case. In particular, the average number of CPU cycles/second allocated by the $m-th$ MEC server to the $i-th$ user, i.e., $\bar{f}_{im}$, is generally lower for Baseline compared to P-C and GT-C, as demonstrated in Fig. 14. Therefore, the Baseline case has higher execution latency components because of lower values of $\bar{f}_{im}$. Furthermore, since $\bar{f}_{im}$ is inversely related to the number of users served by the $m-th$ MEC server $|I_m^{MEC}|$, the Baseline scheme registers the lowest and the highest values of $\bar{f}_{im}$ for MEC servers co-located with gNBs having a high and a low number of users, respectively.

The related CDFs reported in Fig. 15 clearly illustrate the similar behaviors of P-C with different horizons N, that generally has higher values of $\bar{f}_{im}$ with respect to Baseline. Moreover, this figure confirms that the maximum possible value for the number of CPU cycles per second allocated by the $m-th$ MEC server to the $i-th$ user (i.e., f_{im}) is the CPU ability of MEC server F_m^{opt}.

G. Complexity analysis

Despite the overall better performance reported in the above subsections, introducing the anticipatory methods increases the complexity in the network management system, basically due to finding the solution to the optimization problem $P1$. In Table V, P-C and Baseline are compared in terms of the average running time of each decision epoch t_k and the average total number of objective function evaluations needed for each decision epoch t_k, characterizing only P-C solved through value iteration, which is $\sum_{j\in\mathcal{J}} |I_j^{gNB}(t_{k,n})| \cdot |\mathcal{M}| \cdot N$ [73]. GT-C is omitted because the cost of the optimization process is analogous to P-C. However, here it is highlighted that the

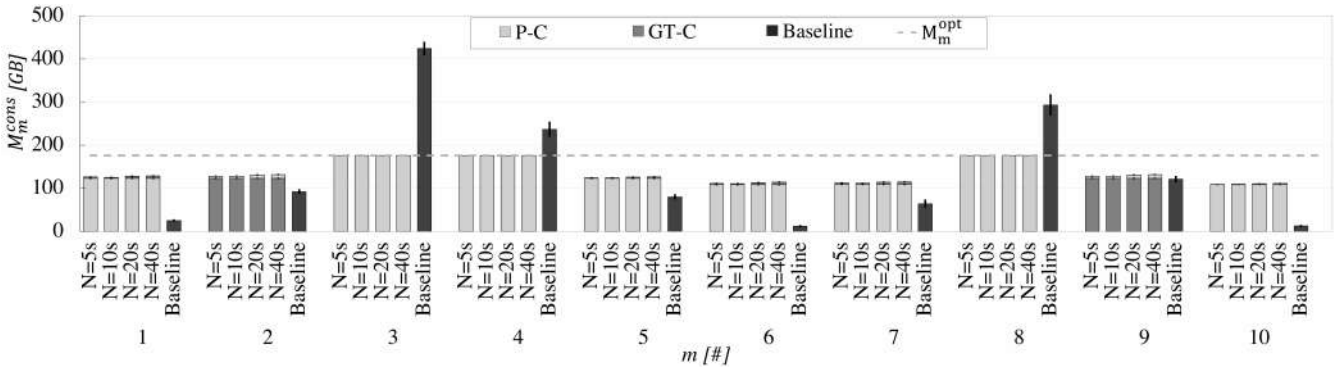

Fig. 12. Average amount of memory (with the 95%−confidence intervals) consumed by each MEC server for P-C, GT-C, and Baseline.

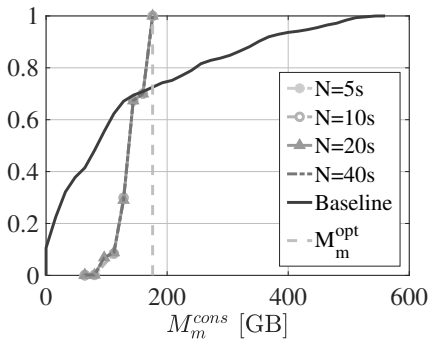

Fig. 13. CDF of the amount of memory consumed by MEC servers for P-C and Baseline.

TABLE V
COMPLEXITY ANALYSIS PER DECISION EPOCH

Complexity parameter	Approach				
	P-C				Baseline
	N=5s	N=10s	N=20s	N=40s	
Running time	4.37s	7.81s	34.78s	78.62s	0.04s
# Objective function evaluations	5321	10125	21369	43791	-

mobility prediction model required by P-C needs an extra-training phase, which early converges anyhow (Section V-A). Without the optimization problem, Baseline has an extremely lower running time because it does not implement any controls and does not anticipatorily evaluate the user distributions. For P-C, it is evident that the complexity increases with N. In fact, the larger the look-ahead horizon N, the deeper in future in the objective function of each $k-th$ optimization problem (i.e., by considering N steps ahead in each decision cycle $t_{k,n}$). Thus, P-C with $N = 5$ s has the lowest average running time and the lowest number of objective function evaluations per decision epoch. Intermediate values are reached when $N = 10$ s and $N = 20$ s and P-C with $N = 40$ s registers the highest complexity. Again, $N = 10$ s is the best trade-off between performance and complexity.

Note that simulations have been executed on an Intel Core i5 CPU quad-core with 8 GB of RAM and the running time will be extremely reduced on a powerful machine with GPU, by improving the efficiency of the proposed approach [74], [75]. In particular, GPU server is at least 4-5 times faster than CPU server (with 16/24 cores) [75]. The significant profit of using a powerful machine makes the running time not only comparable to but much lower than the optimization epoch of the optimization algorithm. The effectiveness can be further enhanced (i.e., much shorter running time) by using a GPU server with more features [75], that are actually used by network operators.

It is remarked here that the encouraging results achieved by the proposed anticipatory network optimization approach open future research directions aiming at decreasing the computational complexity of the proposed solution based on dynamic programming, while maintaining the same performance. To this aim, solutions based on deep reinforcement learning [76] and distributed training [77] seem interesting areas to be further explored.

VI. CONCLUSIONS

This work presented a novel methodology for anticipatorily allocating communication and computational resources at the network edge, and over different look-ahead temporal horizons. Specifically, the Convolutional Long Short-Term Memory has been used to predict the number of users served within a given number of cells and their related service demands, and the Dynamic Programming has been exploited to optimally allocate users' requests among Multi-access Edge Computing servers for better managing task offloading within a network slice created into a 5G system. By focusing the attention on the autonomous driving use case, computer simulations demonstrated how the proposed solution is able to fairly distribute users' requests at the network edge, while satisfying communication and computational constraints, as well as ensuring the upper bound of the communication latency expected for the considered service. Future research activities will further extend the presented study by considering more complex network scenarios with heterogeneous services sharing a variable amount of resources at the network edge, energy consumptions, and realistic Beyond 5G deployments.

This article has been accepted for publication in IEEE Transactions on Network and Service Management. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2021.3099472

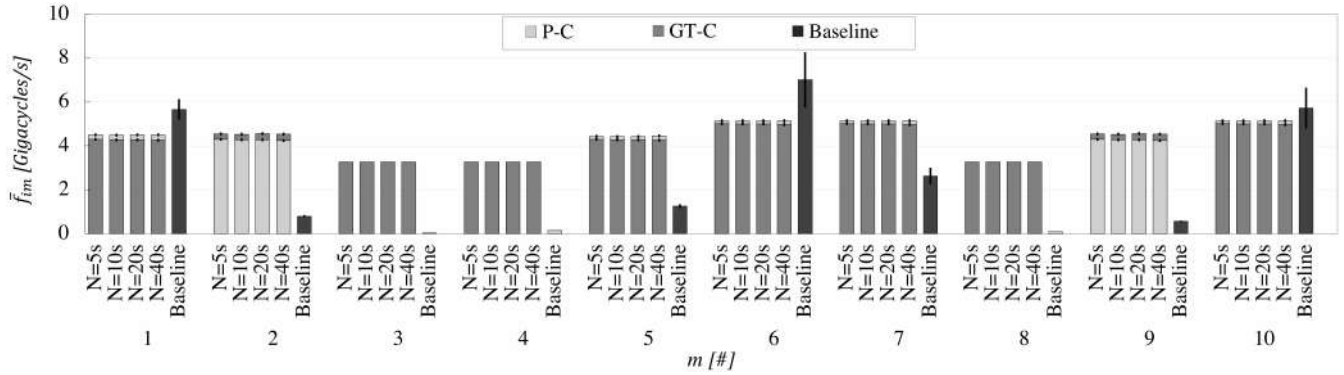

Fig. 14. Average number of CPU cycles/second (with the $95\%-$confidence intervals), allocated by each MEC server to served users, for P-C, GT-C, and Baseline.

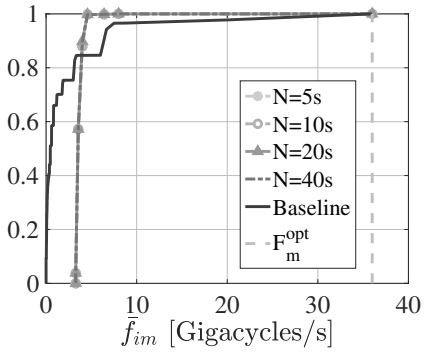

Fig. 15. CDF of the average number of CPU cycles/second per user for P-C and Baseline.

REFERENCES

[1] Q. Pham *et al.*, "A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art," *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020.

[2] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Comput. Netw.*, vol. 167, p. 106984, 2020.

[3] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 2017.

[4] ETSI, "Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements," European Telecommunications Standards Institute, Group Specification (GS) MEC 002, Oct. 2018, V2.1.1.

[5] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine Learning for 5G/B5G Mobile and Wireless Communications: Potential, Limitations, and future Directions," *IEEE Access*, vol. 7, pp. 137 184–137 206, 2019.

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[7] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4188–4200, 2019.

[8] M. Berno, J. J. Alcaraz, and M. Rossi, "On the Allocation of Computing Tasks under QoS Constraints in Hierarchical MEC Architectures," in *Proc. 4th IEEE Int. Conf. Fog Mobile Edge Comput.*, 2019, pp. 37–44.

[9] S. Sardellitti, M. Merluzzi, and S. Barbarossa, "Optimal Association of Mobile Users to Multi-Access Edge Computing Resources," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2018, pp. 1–6.

[10] Y. Yue *et al.*, "Resource Optimization and Delay Guarantee Virtual Network Function Placement for Mapping SFC Requests in Cloud Networks," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 1508–1523, 2021.

[11] F. Zhao, Y. Chen, Y. Zhang, Z. Liu, and X. Chen, "Dynamic Offloading and Resource Scheduling for Mobile Edge Computing With Energy Harvesting Devices," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 2154–2165, 2021.

[12] L. Ferdouse, A. Anpalagan, and S. Erkucuk, "Joint Communication and Computing Resource Allocation in 5G Cloud Radio Access Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 9122–9135, 2019.

[13] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency and Mobility-Aware Service Function Chain Placement in 5G Networks," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2020.

[14] K. Guo, R. Gao, W. Xia, and T. Q. S. Quek, "Online Learning based Computation Offloading in MEC Systems with Communication and Computation Dynamics," *IEEE Trans. Commun.*, vol. 69, no. 2, pp. 1147–1162, 2021.

[15] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. S. Monroy, "Multi-Objective Computation Sharing in Energy and Delay Constrained Mobile Edge Computing Environments," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2020.

[16] H. Peng, Q. Ye, and X. S. Shen, "SDN-Based Resource Management for Autonomous Vehicular Networks: A Multi-Access Edge Computing Approach," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 156–162, 2019.

[17] S. Deng *et al.*, "Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7457–7469, 2020.

[18] P. Roy *et al.*, "User mobility and Quality-of-Experience aware placement of Virtual Network Functions in 5G," *Comput. Commun.*, vol. 150, pp. 367–377, 2020.

[19] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "LORM: Learning to Optimize for Resource Management in Wireless Networks With Few Training Samples," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 665–679, Jan 2020.

[20] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep Reinforcement Learning for Offloading and Resource Allocation in Vehicle Edge Computing and Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11 158–11 168, 2019.

[21] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, pp. 1–1, 2019.

[22] G. Wang, F. Xu, and C. Zhao, "Multi-Access Edge Computing Based Vehicular Network: Joint Task Scheduling and Resource Allocation Strategy," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2020, pp. 1–6.

[23] A. Azari, P. Papapetrou, S. Denic, and G. Peters, "User Traffic Prediction for Proactive Resource Management: Learning-Powered Approaches," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[24] B. Ma, W. Guo, and J. Zhang, "A Survey of Online Data-Driven Proactive 5G Network Optimisation Using Machine Learning," *IEEE Access*, vol. 8, pp. 35 606–35 637, 2020.

[25] X. Wang *et al.*, "Convergence of Edge Computing and Deep Learning: A Comprehensive Survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2020.

[26] W. Zhan *et al.*, "Mobility-Aware Multi-User Offloading Optimization for Mobile Edge Computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3341–3356, 2020.

[27] X. Yu, M. Guan, M. Liao, and X. Fan, "Pre-Migration of Vehicle to Network Services Based on Priority in Mobile Edge Computing," *IEEE Access*, vol. 7, pp. 3722–3730, 2019.

[28] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic Resource Allocation Exploiting Mobility Prediction in Mobile Edge Computing," in *Proc.*

27th IEEE Annu. Int. Symp. Pers. Indoor Mobile Radio Commun., 2016, pp. 1–6.

[29] C. Nguyen, C. Klein, and E. Elmroth, "Multivariate LSTM-based Location-aware Workload Prediction for Edge Data Centers," in *Proc. 19th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, 2019, pp. 341–350.

[30] W.-C. Chien *et al.*, "Multiple Contents Offloading Mechanism in AI-enabled Opportunistic Networks," *Comput. Commun.*, vol. 155, pp. 93–103, 2020.

[31] A. Dalgkitsis, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Data Driven Service Orchestration for Vehicular Networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4100–4109, 2021.

[32] H. Ma, Z. Zhou, and X. Chen, "Leveraging the Power of Prediction: Predictive Service Placement for Latency-Sensitive Mobile Edge Computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6454–6468, 2020.

[33] C.-L. Wu, T.-C. Chiu, C.-Y. Wang, and A.-C. Pang, "Mobility-Aware Deep Reinforcement Learning with Glimpse Mobility Prediction in Edge Computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–7.

[34] Y. Chen, C. Long, G. Cong, and C. Li, "Context-aware deep model for joint mobility and time prediction," in *Proc. 13th ACM Int. Conf. Web Search Data Mining*, 2020, pp. 106–114.

[35] I. Labriji, F. Meneghello, D. Cecchinato, S. Sesia, E. Perraud, E. C. Strinati, and M. Rossi, "Mobility Aware and Dynamic Migration of MEC Services for the Internet of Vehicles," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 1, pp. 570–584, 2021.

[36] M. Polese *et al.*, "Machine learning at the edge: A data-driven architecture with applications to 5G cellular networks," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2020.

[37] O. Narmanlioglu, E. Zeydan, M. Kandemir, and T. Kranda, "Prediction of Active UE Number with Bayesian Neural Networks for Self-Organizing LTE Networks," in *Proc. 8th IEEE Int. Conf. Netw. Future*, 2017, pp. 73–78.

[38] C. Zhang, P. Patras, and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 2019.

[39] N. Bui *et al.*, "A Survey of Anticipatory Mobile Networking: Context-Based Classification, Prediction Methodologies, and Optimization Techniques," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1790–1821, 2017.

[40] T. Dlamini, Á. F. Gambín, D. Munaretto, and M. Rossi, "Online Resource Management in Energy Harvesting BS Sites Through Prediction and Soft-Scaling of Computing Resources," in *Proc. 29th IEEE Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, 2018, pp. 1820–1826.

[41] Á. F. Gambín and M. Rossi, "A Sharing Framework for Energy and Computing Resources in Multi-Operator Mobile Networks," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 2, pp. 1140–1152, 2019.

[42] T. Subramanya, D. Harutyunyan, and R. Riggio, "Machine Learning-driven Service Function Chain Placement and Scaling in MEC-enabled 5G Networks," *Comput. Netw.*, vol. 166, p. 106980, 2020.

[43] L. Chen *et al.*, "Deep Mobile Traffic Forecast and Complementary Base Station Clustering for C-RAN Optimization," *J. Netw. Comput. Appl.*, vol. 121, pp. 59–69, 2018.

[44] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide Cellular Traffic Prediction based on Densely Connected Convolutional Neural Networks," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1656–1659, 2018.

[45] L. Chen *et al.*, "Data-Driven C-RAN Optimization Exploiting Traffic and Mobility Dynamics of Mobile Users," *IEEE Trans. Mobile Comput.*, vol. 20, no. 5, pp. 1773–1788, 2021.

[46] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 23 511–23 528, 2018.

[47] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine Learning Meets Computation and Communication Control in Evolving Edge and Cloud: Challenges and Future Perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 38–67, 2020.

[48] A. Rago, P. Ventrella, G. Piro, G. Boggia, and P. Dini, "Towards an Optimal Management of the 5G Cloud-RAN through a Spatio-Temporal Prediction of Users' Dynamics," in *Proc. 18th IEEE Mediterranean Commun. Comput. Netw. Conf. (MedComNet)*, June 2020, pp. 1–4.

[49] A. Rago, G. Piro, G. Boggia, and P. Dini, "A Softwarized Service Infrastructure for the Dynamic Orchestration of IT Resources in 5G Deployments," in *Proc. 22nd IEEE Int. Conf. Transparent Opt. Netw. (ICTON)*, July 2020, pp. 1–4.

[50] S. Xingjian *et al.*, "Convolutional LSTM network: A Machine Learning Approach for Precipitation Nowcasting," in *Proc. ACM Advances Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[51] D. P. Bertsekas, *Dynamic programming and optimal control*. Belmont, MA, USA: Athena Scientific, 2005, vol. 1.

[52] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi, "CRAWDAD dataset roma/taxi (v. 2014-07-17)," Downloaded from https://crawdad.org/roma/taxi/20140717/taxicabs, Jul. 2014.

[53] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture," *IEEE Commun. Mag.*, vol. 52, no. 11, pp. 65–75, 2014.

[54] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A Survey on Low Latency Towards 5G: RAN, Core Network and Caching Solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3098–3130, 2018.

[55] ITU-R, "Minimum requirements related to technical performance for IMT-2020 radio interface(s)," ITU Radiocommunication Sector, Rep. M.2410, November 2017.

[56] W. Almughalles, R. Chai, J. Lin, and A. Zubair, "Task Execution Latency Minimization-based Joint Computation Offloading and Cell Selection for MEC-Enabled HetNets," in *Proc. 28th IEEE Wireless Opt. Commun. Conf.*, 2019, pp. 1–5.

[57] M. Jung *et al.*, "Driving into the Memory Wall: the Role of Memory for Advanced Driver Assistance Systems and Autonomous Driving," in *Proc. ACM Int. Symp. Memory Syst.*, 2018, pp. 377–386.

[58] J. Xiong, H. Guo, and J. Liu, "Task Offloading in UAV-Aided Edge Computing: Bit Allocation and Trajectory Optimization," *IEEE Commun. Lett.*, vol. 23, no. 3, pp. 538–541, 2019.

[59] M. Bouet and V. Conan, "Mobile Edge Computing Resources Optimization: A Geo-clustering Approach," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 2, pp. 787–796, 2018.

[60] 3GPP, "5G; Management and orchestration; Concepts, use cases and requirements," 3rd Generation Partnership Project, Tech. Specification (TS) 28.530, Oct. 2019, V15.2.0.

[61] ETSI, "Multi-access Edge Computing (MEC); Framework and Reference Architecture," European Telecommunications Standards Institute, Group Specification (GS) MEC 003, Dec. 2020, V2.2.1.

[62] A. H. Mousa, N. T. Mohammed, and E. A. Mohammed, "EFCNT: An evaluation framework for computer's network topologies," *AIP Conf. Proc.*, vol. 2144, no. 1, p. 050010, 2019.

[63] D. Kreutz *et al.*, "Software-Defined Networking: A Comprehensive Survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[64] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[65] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[66] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proc. 32nd ACM Int. Conf. Machine Learn.*, 2015, pp. 448–456.

[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015, pp. 1–15.

[68] 3GPP, "5G; Study on scenarios and requirements for next generation access technologies," 3rd Generation Partnership Project, Tech. Rep. (TR) 38.913, July 2020, V16.0.0.

[69] T. Lian, Y. Zhou, X. Wang, N. Cheng, and N. Lu, "Predictive Task Migration Modeling in Software Defined Vehicular Networks," in *Proc. 4th IEEE Int. Conf. Comput. Comm. Syst.*, 2019, pp. 570–574.

[70] K. Gilly, S. Filiposka, and S. Alcaraz, "Predictive Migration Performance in Vehicular Edge Computing Environments," *Appl. Sci.*, vol. 11, no. 3, p. 944, 2021.

[71] ETSI, "Multi-access Edge Computing (MEC); MEC 5G Integration," European Telecommunications Standards Institute, Group Report (GR) MEC 031, Oct. 2020, V2.1.1.

[72] H. Mazouzi, K. Boussetta, and N. Achir, "Maximizing Mobiles Energy Saving Through Tasks Optimal Offloading Placement in two-tier Cloud: A Theoretical and an Experimental Study," *Comput. Commun.*, vol. 144, pp. 132–148, 2019.

[73] N. Shimkin, "Learning in Complex Systems: Dynamic Programming – Finite Horizon," 2011. [Online]. Available: https://webee.technion.ac.il/shimkin/LCS11/ch2_DP_finite.pdf

[74] C. Cullinan, T. R. Frattesi, and C. Wyant, "Computing Performance Benchmarks among CPU, GPU, and FPGA," MathWorks, 2013. [Online]. Available: https://m.wpi.edu/Pubs/E-project/Available/E-project-030212-123508/unrestricted/Benchmarking_Final.pdf

[75] E. Buber and D. Banu, "Performance Analysis and CPU vs GPU Comparison for Deep Learning," in *Proc. 6th IEEE Int. Conf. Control Eng. Inf. Technol.*, 2018, pp. 1–6.

[76] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[77] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, Methods, and Future Directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.

Arcangela Rago (Graduate Student Member, IEEE) received the M.Sc. degree (with honors) in telecommunication engineering from Politecnico di Bari, Bari, Italy, in 2018, where she is currently pursuing the Ph.D. degree with the Department of Electrical and Information Engineering. Her main research interests include machine learning and data analytics for network optimization. She was a recipient of the Best Poster Award at SMFC 2019, held in conjunction with IEEE SMC 2019. She is involved in the Apulia Region (Italy) Research project INTENTO (36A49H6).

Giuseppe Piro (Member, IEEE) is an Assistant Professor in Telecommunication at Politecnico di Bari since November 2018. In March 2018, he held the habilitation as "Associate Professor" in Telecommunications Engineering, according to the National Scientific Habilitation procedure (ASN 2016-2018). He received the first level degree and the second level degree (both cum laude) in Telecommunications Engineering from "Politecnico di Bari", Italy, in 2006 and 2008, respectively. He received the Ph.D. degree in Electronic Engineering from "Politecnico di Bari", Italy, in March 2012. His main research interests include secure Internet of Things and Industry 4.0, 5G and B5G systems, data-centric and programmable architectures for the Future Internet, nano-networks, Internet models, and network measurements. His research activity is documented in more than 100 peer-reviewed international journals and conference papers, accounting for more than 4200 citations and an H-index of 25 (Scholar Google). At the time of this writing, he is the local investigator of the PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges". Moreover, he is involved in the European EU H2020 GUARD project and in the European Space Agency (ESA) project funded under the contract no. 4000129810/20/NL/CLP. He is also involved in Italian MIUR PON projects (Pico&Pro, FURTHER, AGREED, RAFAEL) and in Apulia Region (Italy) Research project INTENTO. He founded 5G-air-simulator, LTE-Sim, and NANO-SIM projects and is a developer of Network Simulator 3. In the past, he was involved in EU H2020 projects, like FANTASTIC-5G, BONVOYAGE, and symbIoTe, in the "Apulia Israel joint Accelerator (AIJA)" project, and in the Italian MISE project entitled "Pre-commercial trials of 5G technology using spectrum in the 3.6 GHz-3.8 GHz range" - Area Milano (bando MISE), coordinated by Vodafone. He is also regularly involved as member of the TPC of many prestigious international conferences. Currently, he serves as Associate Editor for Sensors journal (MDPI), Internet Technology Letter (Wiley), and Wireless Communications and Mobile Computing journal (Hindawi).

Gennaro Boggia (Senior Member, IEEE) received, with honors, the Dr. Eng. and Ph.D. degrees in electronics engineering, both from the Politecnico di Bari, Bari, Italy, in July 1997 and March 2001, respectively. Since September 2002, he has been with the Department of Electrical and Information Engineering, Politecnico di Bari, where he is currently a Full Professor. From May 1999 to December 1999, he was a Visiting Researcher with the TILab, TelecomItalia Lab, Torino, Italy, where he was involved in the study of the core network for the evolution of Third-Generation (3G) cellular systems. In 2007, he was a Visiting Researcher at FTW, Vienna, Austria, where he was involved in activities on passive and active traffic monitoring in 3G networks. He has authored or coauthored more than 150 papers in international journals or conference proceedings. He is active in the IETF ICNRG working group and in the IEEE WG 6TiSCH. He is also regularly involved as a Member of the Technical Program Committee of many prestigious international conferences. His research interests include the fields of Wireless Networking, Cellular Communication, Internet of Things (IoT), Network Security, Security in IoT, Information-Centric Networking (ICN), Protocol stacks for industrial applications, Internet measurements, and Network Performance Evaluation. Dr. Boggia is currently an Associate Editor for the ETT Wiley Journal and the Springer Wireless Networks journal.

Paolo Dini received M.Sc. and Ph.D. from the Università di Roma La Sapienza, in 2001 and 2005, respectively. He is currently a Senior Researcher with the Centre Tecnologic de Telecomunicacions de Catalunya (CTTC). His current research interests include sustainable networking and computing, distributed optimization and optimal control, multi-agent systems, machine learning and data analytics. His research activity is documented in more than 80 peer-reviewed scientific journals and international conference papers. He received two awards from the Cisco Silicon Valley Foundation for his research on heterogeneous mobile networks, in 2008 and 2011, respectively. He has been involved in more than 20 research and development projects and is currently the Scientific Coordinator of the EU H2020 MSCA Greenedge European Training Network on edge intelligence and sustainable computing. He serves as a TPC in many international conferences and workshops and as a reviewer for several scientific journals of the IEEE, Elsevier, ACM, Springer, Wiley.