Time-Sensitive Mobile User Association and SFC Placement in MEC-Enabled 5G Networks

Rasoul Behravesh, Davit Harutyunyan, Estefanía Coronado, *Member, IEEE*,
and Roberto Riggio, *Senior Member, IEEE*

***Abstract*—The ongoing roll-out of 5G networks paves the way for many fascinating applications such as virtual reality (VR), augmented reality (AR), and autonomous driving. Moreover, 5G enables billions of devices to transfer an unprecedented amount of data at the same time. This transformation calls for novel technologies like multi-access edge computing (MEC) to satisfy the stringent latency and bitrate requirements of the mentioned applications. The main challenge pertaining to MEC is that the edge MEC nodes are usually characterized by scarce computational resources compared to the core or cloud, arising the challenge of efficiently utilizing the edge resources while ensuring that the service requirements are satisfied. When considered with the users' mobility, this poses another challenge, which lies in minimization of the service interruption for the users whose service requests are represented as service function chains (SFCs) composed of virtualized network functions (VNFs) instantiated on the MEC nodes or on the cloud. In this paper, we study the problem of joint user association, SFC placement, and resource allocation, employing mixed-integer linear programming (MILP) techniques. The objective functions of this MILP-based problem formulation are to minimize (i) the service provisioning cost, (ii) the transport network utilization, and (iii) the service interruption. Moreover, a heuristic algorithm is proposed to tackle the scalability issue of the MILP-based algorithms. Finally, comprehensive experiments are performed to draw a comparison between these approaches.**

***Index Terms*—5G, MEC, SFC placement, user association, resource allocation, state exchange.**

I. Introduction

THE 5th generation (5G) of cellular networks promises to transform the mobile communication landscape by offering an extremely high quality of experience (QoE), sub-millisecond latency, higher connection density, multi-Gbps data rates, and so forth to the human and non-human end-users [1]. This opens the door for new revenue streams for mobile network operators (MNO) and the third-party service providers, enabling them to offer many novel applications and services, such as augmented reality, virtual reality, autonomous driving, high-definition sensor sharing, whose stringent QoS have not been able to satisfy with the previous generations of mobile networks [2]. Nonetheless, it also calls for novel technological solutions to meet the requirements of such applications. Multi-access edge computing (MEC) [3] is one of such technologies that is expected to play a pivotal role in 5G networks by shifting the applications, services, and processing capabilities closer to the end-users and, therefore, offloading the transport network and reducing the round-trip delay experienced by the end-users. For instance, owing to the network function virtualization (NFV) technology, MEC enables the 5G core network functions and applications to be deployed at the network edge as a chain of virtualized network functions (VNFs) known as service function chains (SFCs) [4].

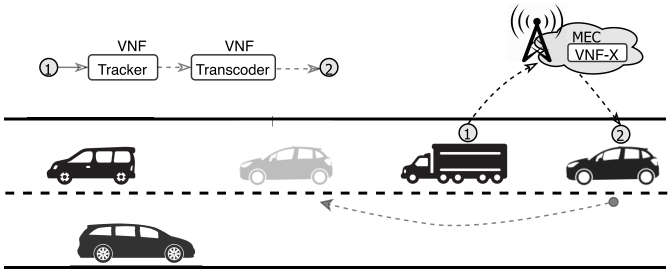


Fig. 1. An application example with a low latency requirement.

Figure 1 demonstrates an example of a use case, called See-through [5], that can take advantage of the MEC and NFV technologies. The figure depicts a car (number 2) being stuck behind a slow-moving truck (number 1) incapable of seeing the front to check whether it is safe to overtake. The truck transmits the live video frames captured by the forward-facing cameras to an application (composed of two VNFs, tracker, and transcoder) hosted on a MEC server collocated with the next generation NodeB (gNB – base station in 5G networks) in proximity. Once the MEC server has processed the video frames, the gNB transmits them to the car behind the truck, which exploits that information to decide if and when to perform the maneuver.

MEC servers may reside along with the gNBs as well as with the core network. While these MEC servers can be used to

Manuscript received October 29, 2020; revised February 26, 2021; accepted April 29, 2021. Date of publication May 10, 2021; date of current version September 9, 2021. This work has been performed in the framework of the European Union's Horizon 2020 projects 5GZORRO, 5G-CARMEN, 5GMED, and AI@EDGE co-funded by the EU under grant agreements No. 871533, No. 825012, No. 951947, and No. 101015922, respectively. The associate editor coordinating the review of this article and approving it for publication was B. Cheng. *(Corresponding author: Rasoul Behravesh.)*

Rasoul Behravesh is with the Smart Networks and Services (SENSE) Unit, Fondazione Bruno Kessler, 38100 Trento, Italy, and also with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, 40126 Bologna, Italy (e-mail: rbehravesh@fbk.eu).

Davit Harutyunyan is with Corporate Research, Robert Bosch GmbH, 70839 Gerlingen, Germany (e-mail: davit.harutyunyan@de.bosch.com).

Estefanía Coronado is with the Department of Software Networks, i2CAT Foundation, 08034 Barcelona, Spain (e-mail: estefania.coronado@i2cat.net).

Roberto Riggio is with the Connected Intelligence Group, RISE Research Institutes of Sweden AB, 111 21 Stockholm, Sweden (e-mail: roberto.riggio@ri.se).

Digital Object Identifier 10.1109/TNSM.2021.3078814

host low-latency services, the cloud data centers can be used to accommodate the latency-tolerant ones. In general, the closer the MEC server is located to the user, the less is its computational capacity, which means that the more costly is VNF instantiation on that MEC server [6]. Given the above considerations and a number of users requesting various applications with diverse QoS requirements, the natural question that arises is which gNBs to associate the users with and where to deploy their requested applications, such as to make sure that their service requirements are satisfied while the network resources are used in the most efficient manner?

This paper significantly extends our previous study [7] in multiple aspects. Specifically, we employ mixed-integer linear programming (MILP) techniques to provide a novel formulation of the problem whose objectives are to minimize (i) the service provisioning cost, which considers the CPU cost, the link bandwidth consumption cost, user state exchange cost, as well as the physical resource block (PRB)[1] utilization cost, (ii) the transport network utilization, and (iii) the service interruption. Mobile users are considered making service requests, which are represented as SFCs composed of different numbers of VNFs having diverse latency and data rate requirements. We also propose a heuristic algorithm that reaches a near-optimal solution to minimize service interruption caused for the users in a much shorter time scale compared to the proposed MILP-based algorithm.

The rest of the paper is structured as follows. The related work is discussed in Section II. The problem statement, along with the mobile network model and service request model are introduced in Section III. The MILP problem formulation is presented in Section IV, followed by the numerical results reported in Section V. Finally, Section VI draws the conclusions.

II. Related Work

A. User Association

The user association problem in 5G networks is one of the sub-problems studied in our work. An optimal user association mechanism results in an efficient PRB utilization at gNBs, while ensuring the required QoE for the users [8]. A sizable body of papers have been published on the user association problem in 5G networks [9]–[17].

The study in [9] formulates the problem of user association in HetNets as a Nash bargaining problem. The objective is to maximize data rate utility while guaranteeing the minimal data requested by users and equally distributing the load among the base stations. The authors of [10] design a delay-aware user association strategy for 5G HetNets with the goal of minimizing the overall power consumption in the network while applying strict delay constraints. In [11], the problem of user association in 5G ultra-dense multi-RAT HetNets is formulated as a multi-objective optimization problem, which is solved leveraging the weighted sum technique. The work in [12] presents a constrained optimization method for mobility-aware user association in mmWave networks. The method is capable of tracking the frequent variations in the network topology and channel condition. Similarly, the work in [13] addresses the UE association problem in 5G HetNets to meet the UE's QoE requirements using a one-to-many matching game based on matching theory. Authors in [14] introduce an optimal user association method in 5G mmWave networks, which can recalculate the cost of possible handovers and also the erratic nature of mmWave channels. Authors of [15] study the user association problem in a cache-enabled mobile network, capturing the trade-off between the radio access network and the transport network utilization in 5G networks. A joint user association and user scheduling solution is presented in [16], where the authors aim to minimize the users' achievable throughput. The work proposed in [17] employs a data-driven technique to predict future traffic patterns then associate users with base stations based on pre-calculated association maps of the given time. However, none of those mentioned above studies jointly consider user association, VNF placement, and resource allocation.

B. SFC Placement

As mentioned earlier, an SFC is a composition of different types and numbers of VNFs interconnected in a particular order to provide a certain service. Therefore, the SFC placement problem is yet another sub-problem studied in our work. There is a sizable body of works studying the SFC placement problem [18]–[30]. Moreover, there are also vast surveys that fully explore this problem from different perspectives such as nature, type of required placement (i.e., dynamic or static), objectives, and metrics of the VNFs [31]–[33].

The study in [18] addresses the problem of SFC placement to efficiently utilize the network resources while respecting the E2E latency requirement of the users. The work in [19] proposes a VNF placement method, which takes advantage of the edge, core, and cloud servers in service-customized 5G networks. An interference-aware method is proposed to tackle the negative effect of the VNF consolidation (i.e., VNF interference) with the goal of maximizing the overall throughput of the accepted requests. Authors in [20] provide two models to calculate, respectively, the transmission delay of flows traversing a chain of VNFs and the availability of SFC for VNF resiliency. Furthermore, they propose an integer non-linear programming (INLP) model and a heuristic algorithm to jointly solve the problems of delay-sensitive VNF placement and VNF resiliency. Similar to this, the approach in [21] solves an SFC-based resource allocation problem using ILP by jointly tackling the VNF placement and routing problem with the objective of reducing energy consumption. The same problem is investigated in [22] by employing MILP techniques through a three-phase study, namely VNF chain composition, VNF forwarding graph embedding, and VNF scheduling. The study in [23] jointly solves the problems of VNF placement and CPU allocation in 5G networks. The authors consider the latency as the main key performance indicator (KPI) and try to minimize the ratio between the actual and maximum allowed latency. The work in [24] utilizes the theory of open Jackson network to evaluate the data traffic in data centers and proposes two heuristic algorithms

[1] PRB, is a chunk of the time-frequency matrix in the radio access network, which is allocated to the users by the gNB scheduler.

to jointly optimize the SFC placement and request scheduling while minimizing the latency and resource utilization in the network. Similarly, the study in [25] proposes a MILP model for VNF placement in hierarchical 5G networks, where VNFs can be deployed on edge, core, and cloud nodes. The main goal is to minimize the overall latency, which is composed of queuing, processing, transmission, propagation, and optical-electronic-optical conversion delay. The parallel VNF deployment approach is adopted in [26] to achieve latency reduction in service delivery. The bottleneck issue caused by the imbalanced deployment of parallel VNFs is mitigated by mapping multiple instances of the VNFs. Authors in [27] introduce an ILP model to map VNFs on the servers to minimize the number of utilized servers. The work, however, does not consider the underlying network characteristics but only services and VM requests. The study in [28] investigates a VNF orchestration problem (VNF-OP) and proposes an ILP and a heuristic solution to determine the number of required VNFs and their locations without violating service level agreements (SLAs). The main objective of the work is to minimize OPEX and resource fragmentation. The authors of [29] jointly study the problem of VNF placement and routing, having an objective of maximizing network throughput. Finally, the authors of [30] jointly tackle VNF placement and resource allocation problems as a mixed-integer program (MIP) based on an SDN/NFV-enabled MEC infrastructure. However, the fitness function does not consider E2E service latency requirements. Our study stands out from these works by taking into account also the impact of user-gNB associations, user equipment (UE) mobility, and state exchange during the SFC placement process.

C. VNF Migration

One of the important aspects to consider in the joint user association, SFC placement, and resource allocation problem is the VNF migration, which mainly occurs due to the UE mobility and increased number of users who share the same VNF. There is a vast array of works studying the VNF migration problem [34]–[38].

The study in [34] defines the VNF migration cost as the overall traffic served by the VNF, which is minimized by an ILP model. Furthermore, trying to tackle the scalability issue of the ILP model, a heuristic model is proposed to minimize the migration cost and satisfy the computing and transport resource utilization constraints. Another study [35] models the problem of VNF migration for latency stringent applications in a highly dynamic environment. The work proposes a heuristic algorithm that triggers the VNF migration based on the applications' latency requirement violation. Authors in [36] introduce a linear programming model to combat the problems of QoS degradation caused by service interruptions and improper load distribution among servers. They study the trade-off between VNF replication and migration of already deployed VNFs to balance the load on servers and reduce the number of migrations. The study in [37] proposes a MILP model to smartly decide whether to migrate or instantiate the VNF of the same service, in case of failure or resource scaling, having the objective of minimizing service downtime and service latency. The work presented in [38] considers flexible placement and migration of VNFs in a MEC-enabled 5G architecture. The authors take into account both computational and network needs of the UEs, and present a proof of concept where the NFV orchestrator handles network resources and services in real-time. As opposed to the studies mentioned above, our work also considers the SFC placement problem apart from the VNF migration. Specifically, one of our objective functions is to minimize the number of UEs that change their serving node. One way to achieve this goal is to minimize the number of VNF migrations and, upon an urgent need for a VNF migration, deciding which VNF to migrate in order to ensure a minimal effect on the UEs served from that VNF.

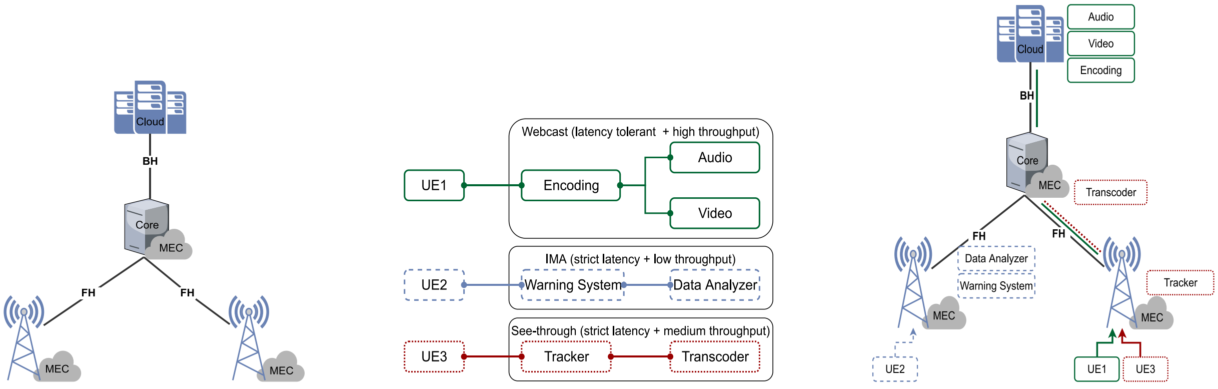
D. Joint User Association and VNF Placement

The closest studies to ours are [39] and [40]. The work in [39] formulates the problem of VNF placement at the network edge to minimize the network latency from the users to their respective VNF hosted on edge servers. A method is presented to dynamically re-schedule VNFs to attain optimal allocation and avoid SLA violations. The study by [40] presents an ILP model to jointly solve the problems of user association, SFC placement, and resource allocation, in which users are assumed to have different E2E latency and data rate requirements. However, both of these studies lack a realistic model to compute the air interface delay. Moreover, they do not consider the state exchange cost for the UEs when they change their serving node. Finally, as opposed to our study, they do not consider the case in which the UEs may be associated with one gNB while still receive service from the VNFs that are instantiated on a MEC server collocated with neighboring gNBs.

III. Network Model

A. Problem Statement

Figure 2(a) depicts the reference network architecture in which the gNBs are collocated with MEC servers, referred to as edge nodes, and are in charge of providing coverage to the users and performing their baseband signal processing. The edge nodes have a limited amount of computational capacity, which makes their usage quite costly. It is important to mention that we also consider the case in which it is possible for a user to be associated with one gNB while be served by a MEC server collocated with another gNB. While all the nodes possess computing capabilities, only the gNBs and the core are equipped with MEC servers. As opposed to the gNBs, the MEC server collocated with the core node has much more computational capacity, making the VNF instantiation upon much cheaper. Nevertheless, VNF instantiation on the core node requires the use of the Fronthaul (FH) transport resources, which contributes to the total cost computation for the VNF instantiation. As for the cloud data center, it has abundant computational resources, which makes it the cheapest solution to be used for instantiating VNFs compared to the edge nodes and the core, regardless of the additionally required transport network resources (i.e., both Fronthaul and



(a) Mobile network.

(b) Service requests.

(c) VNF Mapping.

Fig. 2. Sample mobile network, service requests, and VNF mappings.

Backhaul (BH) resources). Thus, the closer is the computing node to the end-user, the less is its computation capacity.

It is assumed that each UE requests a service with a specific data rate and delay tolerance. Upon receiving the service request from the UE, the MNO shall decide on how to associate the UE to the network and embed its request, such as to make sure that the UE service requirements are satisfied while the network resources are used in the most efficient manner. Figure 2(b) depicts sample service requests composed of UEs and the requested service, having either strict or loose latency requirements as well as low, medium, and high throughput requirements, which are numerically defined in Section V-A. The first is a Webcasting service that provides on-demand high-quality videos to users. The second one is an intersection movement assist (IMA) service that provides warnings to the cars [41]. Finally, the last one is a See-through service that enables drivers to see the blocked areas of the road through other cars (described in Section I). As shown, each service is composed of multiple VNFs that are chained together to deliver the service. Figure 2(c) illustrates a sample service mapping whose objective is to minimize the service provisioning cost. The service requested by UE1 is placed in the cloud, while the services of UE2 and UE3 are mapped on the MEC servers at the edge or core due to their stringent latency requirement. Note that since the VNFs composing a service have different requirements, some VNF instances are placed in the core while the other VNFs of the same service are placed at the edge due to the resource limitations at the edge or cheaper resource cost at the core.

Depending on the requirements of the services and the availability of the substrate network resources, there may be several mapping possibilities, each of which optimizing certain aspects of the network. The problem of joint user association, SFC placement, and resource allocation can be formally stated as follows:

Given: A 5G network composed of gNBs and a core node that have collocated MEC servers and are interconnected via Fronthaul links. Additionally, given a cloud data center node that is interconnected with the core node via a Backhaul link. Moreover, given a set of mobile UEs randomly scattered in a geographical area, requesting a service with a respective data rate and latency requirement.

Find: Joint user association, SFC placement, and resource allocation in the network.

Objective: Minimize (i) the service provisioning cost, (ii) the transport bandwidth consumption, and (iii) service interruption for the UEs.

B. Mobile Network Model

Let $G = (N, E)$ be an undirected graph modeling the mobile network, where N represents the computing nodes, which are the union of the set of gNBs N_{gnb}, the core N_{core}, and the cloud N_{cloud}, $N = N_{gnb} \cup N_{core} \cup N_{cloud}$. E represents the set of FH and BH links interconnecting, respectively, the gNBs with the core, and the core with the cloud. Each computing node $n \in N$ in the network is equipped with a certain amount of processing capacity represented by $\mathcal{C}_{cpu}(n)$. There is a link $e^{m,n} \in E$ between the nodes $m, n \in N$ if they are directly connected. Although the network considered here is composed of a three-tier architecture, it can be adapted easily adapted to different network architecture composed of different computing layers [42], [43] with different costs assigned to resources.

Let ω^i_{cpu} represent the number of CPU cores assigned to the instance $i \in N^v_{inst}$ of VNF of $v \in N^s_{vnf}$ of service $s \in N_{ser}$. It is assumed that at least a single CPU core is required to spawn/instantiate a VNF, while it is also possible to allocate three CPUs to a VNF instance depending on the data processing demand. $\mathcal{C}^i_{proc}(n)$ is the processing capacity of the instance $i \in N^v_{inst}$ of VNF $v \in N^s_{vnf}$ of service $s \in N_{ser}$ on node $n \in N$. There is an upper bound on the number of UEs that can use one CPU core. Thus, the capacity $C^i_{ue}(n)$ of a VNF instance in a service can be expressed in terms of the maximum number of UEs that can use that service, which depends on the service type and the number of CPU cores allocated to that VNF. The more critical a service is, the less is the number of UEs that can share the VNFs of that service due to security reasons. For example, for a video service, the number of users simultaneously using the service is less important, while for the see-through use case the number of users that use the service at the same time is of great importance since it might impact the security aspects of the service. It is worth to mention that we also tackle the case in which multiple instances of the same VNF are needed on a node due

TABLE I
MOBILE NETWORK PARAMETERS

Parameters	Description
$G(N, E)$	Graph representing the mobile network.
N_{gnb}	Set of gNBs in G.
N_{core}	Set of core servers in G.
N_{cloud}	Set of cloud servers in G.
N	Set of computing nodes in G.
E	Set of links connecting the nodes in G.
N_{ser}	Set of services.
N^{s}_{vnf}	Set of VNFs composing a service.
N^{v}_{inst}	Set of instances of VNF $v \in N^{s}_{vnf}$ of service $s \in N_{ser}$.
ω^{i}_{cpu}	The number of CPU cores assigned to instance $i \in N^{v}_{inst}$ of VNF $v \in N^{s}_{vnf}$ of service $s \in N_{ser}$.
ω^{g}_{prb}	The amount of PRB available on gNB $g \in N_{gnb}$.
$\omega^{u,n}_{sta}$	The amount of state of UE $u \in N_{ue}$ on instance $i \in N^{v}_{inst}$.
ξ^{n}_{cpu}	The cost of one CPU core on node $n \in N$.
ξ^{e}_{bwt}	The cost of using one Mbps bandwidth of link $e \in E$.
ξ^{g}_{prb}	The cost of using one PRB in gNB $g \in N_{gnb}$.
ξ^{n}_{sta}	The cost of exchanging one Mbps of UE state from node $n \in N$.
$\mathcal{C}^{u}_{g}$	The maximum achievable data rate between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.
$\mathcal{C}^{i}_{ue}(n)$	The maximum number of UEs that can use the instance $i \in N^{v}_{inst}$ of VNF $v \in N^{s}_{vnf}$ of service $s \in N_{ser}$ on node $n \in N$.
$\mathcal{C}_{cpu}(n)$	The CPU cores of node $n \in N$.
$\mathcal{C}^{i}_{proc}(n)$	Processing capacity of instance $i \in N^{v}_{inst}$ of VNF $v \in N^{s}_{vnf}$ of service $s \in N_{ser}$ on node $n \in N$.
$\mathcal{C}_{bwt}(e)$	The bandwidth capacity of the substrate link $e \in E$.
$d_{(g,u)}$	Distance between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$.
P^{g}_{tx}	The transmission power of gNB $g \in N_{gnb}$.
$\tilde{\chi}^{i}_{u,n}$	A parameter which shows the previous assignment of UE $u \in \bar{N}_{ue}$ on instance $i \in N^{v}_{inst}$ of VNF $v \in N^{s}_{vnf}$ of service $s \in N_{ser}$ on node $n \in N$.
$\Lambda^{i}_{ue}(n)$	The number of UEs $u \in \bar{N}_{ue}$ served from the VNF instance $i \in N^{v}_{inst}$ on the node $n \in N$ in the previous run.
μ	A big positive number.

to high traffic demand. Finally, each link $e^{m,n} \in E$ connecting the nodes $m, n \in N$ in the network has a certain bandwidth capacity $\mathcal{C}_{bwt}(e)$ in Gbps. Table I summarizes the parameters of the mobile network.

C. Service Request Model

We model the service requests as a directed graph $\bar{G} = (\bar{N}, \bar{E})$, where $\bar{N}$ is the union of UEs and their requested services, $\bar{N} = \bar{N}_{ue} \cup \bar{N}_{ser}$, and $\bar{E}$ represents the virtual links between UEs and their requested services. It is assumed that the UEs, each of which can be associated with one gNB, are randomly scattered in the given geographical area and are moving in different directions with different speeds mimicking real-life scenarios.

In our model each UE $u \in \bar{N}_{ue}$ requests only one service $s \in \bar{N}_{ser}$, specifying the maximum delay tolerance by T^{u}_{max} and data rate demand ω^{u}_{bwt}. The allocated VNF instances that compose the service should process the data transmitted by the UE. The total delay of the service is calculated as the summation of the transmission time over the air, which is considered to be equal to one transmission time interval (TTI = 1ms), transmission time over FB and BH links, propagation time over the air and transport network, and the processing time of the VNF instances. Table II summarizes the notations used for the service requests.

TABLE II
SERVICE REQUEST MODEL

Parameters	Description
$\bar{G}(\bar{N}, \bar{E})$	Service request graph.
$\bar{N}$	Set of UEs and requested services $\bar{N} = \bar{N}_{ue} \cup \bar{N}_{ser}$ in $\bar{G}$.
$\bar{N}_{ue}$	Set of UEs in $\bar{G}$.
$\bar{N}_{ser}$	Set of services requested by the UEs in $\bar{G}$.
$\bar{E}$	Set of virtual links connecting UEs to the VNFs of the requested service in $\bar{G}$.
ω^{u}_{bwt}	Data rate requested from UE $u \in \bar{N}_{ue}$.
$\omega^{u,g}_{prb}$	The number of required PRBs to support the data request of UE $u \in \bar{N}_{ue}$ from gNB $g \in N_{gnb}$.
T^{u}_{max}	Maximum delay tolerance of UE $u \in \bar{N}_{ue}$.
$T^{u}_{tx}(g)$	The transmission time between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.
$T^{u}_{prp}(g)$	The propagation time between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.

D. Air Interface Capacity Calculation

The air interface capacity between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$ is denoted by $\mathcal{C}^{u}_{g}$, which is a function of signal-to-interference-plus-noise-ratio (SINR) that can be computed through the following equation:

$$\forall g \in N_{gnb},\ \forall u \in \bar{N}_{ue}:$$

$$SINR_{g,u} = \frac{P^{g}_{tx}\, d^{-\delta}_{(g,u)}}{\mathcal{N}^2 + \sum_{k \neq g} P^{k}_{tx}\, d^{-\delta}_{(k,u)}} \tag{1}$$

where P^{g}_{tx} denotes the transmission power of gNB $g \in N_{gnb}$. It is worth noting that UEs will experience different signal strengths from the gNBs since cells are overlapping in the area of coverage. $d(g, u)$ is the Euclidean distance between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$, while δ represents the path loss coefficient and $\mathcal{N}$ is the noise power. Accordingly, if we define W as the system bandwidth, the maximum achievable air interface capacity $\mathcal{C}^{u}_{g}$ between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$ can be computed as follows:

$$\mathcal{C}^{u}_{g} = W \log\bigl(1 + SINR_{g,u}\bigr) \tag{2}$$

Based on the UE's channel quality indicator (CQI) value, which can be obtained from the mapping table using the UE's SINR, we can compute the number of PRBs required to satisfy data rate demand of the UE [44]. The CQI is determined in a way that corresponds to the highest modulation and coding scheme (MCS), which also can be derived from the mapping table given in [44]. Given the throughput demand ω^{u}_{bwt} of UE $u \in \bar{N}_{ue}$, the number of required PRBs to meet the data rate demand of the UE from gNB $g \in N_{gnb}$ can be computed as follows [15]:

$$\omega^{u,g}_{prb} = \frac{\omega^{u}_{bwt}\, T_{sbf}}{2 N_{sbc} N_{sym} N^{u,g}_{modb} N_{ant}} \tag{3}$$

where T_{sbf} is the duration of one sub-frame (1ms) and ω^{u}_{bwt} represents the throughput requested from the UE. N_{sbc}

represents the number of sub-carries, which is equal to 12 sub-carries per PRB. N_{sym} represents the number of symbols per slot which is equal to 7 and we have 2 slots per sub-frame [45]. Also, $N_{modb}^{u,g}$ and N_{ant}, respectively, represent the number of modulated bits per symbol for a given MCS and the number of antennas per gNB that is considered to be 2 in our scenario.

IV. Problem Formulation

The joint user association, SFC placement, and resource allocation problem is modeled as a virtual network embedding (VNE) problem, which is NP-hard and has been studied extensively in the literature [46], [47]. The embedding process consists of two phases: the node embedding and the link embedding. In the node embedding phase, each virtual node (e.g., UEs and VNFs) in the request is mapped to a substrate node (e.g., gNBs, core servers, and cloud nodes in the substrate network). In the link embedding phase, instead, each virtual link is mapped to a single substrate path. In both cases, the constraints of the nodes and links must be satisfied in order for a solution to be valid. In this section we first describe the MILP model formulation of the problem followed by the proposed heuristic algorithm.

A. MILP Formulation

The described VNE problem has been formulated by employing MILP techniques. As mentioned earlier, three objectives are defined for the model. The considered MILP models have the same constraints; however, they differ in terms of their optimization objectives. The objective (4) tends to minimize the service provisioning cost, which encompasses the cost of using computing, link transmission, radio access network resources, and state exchange cost of the UEs. While the costs of using link transmission, radio access network resources, and state exchange are the same for, respectively, all the links, gNBs, and UEs, the cost of the computing resources depends on the type/location of the host node (e.g., edge, core, cloud). The closer the host node is located to the cloud, the more abundant and the cheaper are its resources and, therefore, the cheaper is VNF instantiation. Table III represents the binary and continuous variables used in the MILP model.

$$\boldsymbol{Cost_M} : min \Bigg(\sum_{n\in N} \sum_{s\in N_{ser}} \sum_{v\in N_{vnf}^{s}} \sum_{i\in N_{inst}^{v}} \xi_{cpu}^{n} \omega_{cpu}^{i} \chi_{n}^{i} + \sum_{u\in \bar{N}_{ue}} \sum_{\bar{e}\in \bar{E}^{u}} \sum_{e\in E} \xi_{bwt}^{e} \omega_{bwt}^{u} \chi_{e}^{u,\bar{e}} + \sum_{u\in \bar{N}_{ue}} \sum_{g\in N_{gnb}} \xi_{prb}^{g} \omega_{prb}^{u,g} \chi_{g}^{u} + \sum_{n\in N} \sum_{u\in \bar{N}_{ue}} \sum_{s\in N_{ser}^{u}} \sum_{v\in N_{vnf}^{s}} \sum_{i\in N_{inst}^{v}} \xi_{sta}^{n} \omega_{sta}^{u,i} \chi_{mig}^{u,i}(n) \Bigg) \quad (4)$$

The following objective (5) aims at minimizing the bandwidth consumption of the transport network. This objective is particularly useful for the cases in which the transport network lacks of capacity, or the UE requested service is latency sensitive.

$$\boldsymbol{Link_M} : min \sum_{u\in \bar{N}_{ue}} \sum_{\bar{e}\in \bar{E}} \sum_{e\in E} \omega_{bwt}^{u} \chi_{e}^{u,\bar{e}} \quad (5)$$

TABLE III
Binary (χ) and Continuous (T) Variables

Variables	Description
χ_{g}^{u}	Indicates if UE $u \in \bar{N}_{ue}$ is associated to gNB $g \in N_{gnb}$.
χ_{n}^{i}	Indicates if instances $i \in N_{inst}^{v}$ of VNF $v \in N_{vnf}^{s}$ of service $s \in N_{ser}$ is running on node $n \in N$.
$\chi_{u,n}^{i}$	Indicates if instances $i \in N_{inst}^{s}$ of service $s \in N_{vnf}^{s}$ is running on node $n \in N$ and assigned to UE $u \in \bar{N}_{ue}$.
$\chi_{e}^{u,\bar{e}}$	Indicates if the virtual link $\bar{e} \in \bar{E}$ belonging to the request by UE $u \in \bar{N}_{ue}$ is mapped on the substrate link $e \in E$.
$\chi_{mig}^{u,i}(n)$	Indicates if the hosting node $n \in N$ of instance $i \in N_{inst}^{s}$ of service $s \in N_{vnf}^{s}$ that was serving node UE $u \in \bar{N}_{ue}$ has been changed.
$T_{proc}^{i}(n)$	Processing time of instance $i \in N_{inst}^{v}$ of VNF $v \in N_{vnf}^{s}$ of service $s \in N_{ser}$ on node $n \in N$.
$T_{proc}^{i}(u,n)$	Processing time of instance $i \in N_{inst}^{v}$ of VNF $v \in N_{vnf}^{s}$ of service $s \in N_{ser}$ on node $n \in N$ for UE $u \in \bar{N}_{ue}$.
T_{tx}^{e}	Transmission time over link $e \in E$.
$T_{tx}^{u,\bar{e}}(e)$	Transmission time over link $e \in E$ for virtual link $\bar{e} \in \bar{E}$.

Finally, the goal of the last objective function (6) is to minimize service interruption for the UEs, which greatly impacts the overall perceived QoE for the end-users. Due to the mobility of UEs, the impact of new arriving requests and migration of VNFs caused as a result of the system performance optimization purposes, there are some cases in which the UEs have to change their serving node (a node that hosts the VNF(s) of the UE). It is obvious that this event causes service interruption for the UEs because a process should be executed to provide the new VNF on a destination node for the UE, assign the VNF to the UE, and transfer the state of the UE from the source node to the destination node (the new serving node of the UE). In reality, the longer a UE uses a VNF instance, the more is its states and the more costly it is to transfer the states of that UE. Accordingly, we define a parameter $\Lambda_{ue}^{i}(n)$ that keeps tracking the time (number of runs) that a UE has spent in the network. Consequently, this objective tries to minimize service interruption for the UEs by decreasing the number of times the serving node of a UE changes. In case if it is inevitable to change some UEs' serving nodes, those with minimum time spent in the network will be selected such as to minimize the amount of states that should be transferred and consequently the time that it is needed to transfer the state.

$$\boldsymbol{Mig_M} : min \sum_{n\in N} \sum_{u\in \bar{N}_{ue}} \sum_{s\in N_{ser}^{u}} \sum_{v\in N_{vnf}^{s}} \sum_{i\in N_{inst}^{v}} \Lambda_{ue}^{i}(n) \chi_{mig}^{u,i}(n) \quad (6)$$

In the following, we present the constraints that, regardless of the objective function, have to be satisfied for a solution to be valid. Constraint (7) pertains to the UE association, making sure that each UE is connected to only one candidate gNB, which has to have sufficient air interface capacity (enforced by constraint (8)) and sufficient amount of PRBs in order to satisfy the UEs' data rate demand (enforced by constraint (9)).

$$\forall u \in \bar{N}_{ue} : \sum_{g \in N_{gnb}} \chi_g^u = 1 \tag{7}$$

$$\forall g \in N_{gnb} : \sum_{u \in \bar{N}_{ue}} \omega_{bwt}^u \chi_g^u < \mathcal{C}_g^u \tag{8}$$

$$\forall g \in N_{gnb} : \sum_{u \in \bar{N}_{ue}} \omega_{prb}^{u,g} \chi_g^u \leq \omega_{prb}^g \tag{9}$$

As stated before, our model assumes that each UE requests only one service. Thus, constraint (10) enforces each UE $u \in \bar{N}_{ue}$ to be connected to only a single instance of the VNFs that compose the requested service.

$$\forall u \in \bar{N}_{ue},\ \forall s \in \bar{N}_{ser}^u, \forall v \in N_{vnf}^s : \sum_{n \in N} \sum_{i \in N_{inst}^v} \chi_{u,n}^i = 1 \tag{10}$$

The following constraint guarantees that a VNF instance is spawned/instantiated only if at least one UE is mapped on that VNF instance.

$$\forall n \in N, \forall s \in N_{ser}, v \in N_{vnf}^s,\ \forall i \in N_{inst}^v:$$
$$\sum_{u \in \bar{N}_{ue}} \chi_{u,n}^i - \mu * \chi_n^i \leq 0 \tag{11}$$

Before placing a VNF instance on a node, it should be checked if that node has a sufficient amount of resources to host the VNF, making sure that the number of CPU resources assigned to a VNF instances running on a node does not exceed the CPU capacity of that node (constraint (12)).

$$\forall n \in N : \sum_{s \in N_{ser}} \sum_{v \in N_{vnf}^s} \sum_{i \in N_{inst}^v} \omega_{cpu}^i \chi_n^i \leq \mathcal{C}_{cpu}^n \tag{12}$$

As stated, depending on the service type and the number of CPU cores assigned to a VNF instance, a limited number of UEs can be served from a VNF instance at the same time. In this regard, constraint (13) sets an upper bound on the number of UEs that can use the same VNF instance.

$$\forall n \in N,\ \forall s \in N_{ser}, \forall v \in N_{vnf}^s, \forall i \in N_{inst}^v:$$
$$\sum_{u \in \bar{N}_{ue}} \chi_{u,n}^i \leq \mathcal{C}_{ue}^i(n) \tag{13}$$

Constraint (13) ensures that the virtual links can be mapped onto a substrate link as long as the link has sufficient capacity:

$$\forall e \in E : \sum_{u \in \bar{N}_{ue}} \sum_{\bar{e} \in \bar{E}} \omega_{bwt}^u \chi_{u,\bar{e}}^e \leq \mathcal{C}_{btw}^e \tag{14}$$

Constraint (15) indicates if the serving node *n* of user $u \in \bar{N}_{ue}$ for the VNF $v \in \bar{N}_{vnf}$ of service $s \in \bar{N}_{ser}$ has been changed.

$$\forall n \in N, \forall u \in \bar{N}_{ue}, \forall s \in \bar{N}_{ser}^u, \forall v \in N_{vnf}^s:$$
$$\sum_{i \in N_{inst}^v} \tilde{\chi}_{u,n}^i - \sum_{i \in N_{inst}^v} \chi_{u,n}^i - \chi_{mig}^{u,i}(n) \leq 0 \tag{15}$$

The processing time $T_{proc}^i(n)$ of the i^{th} instance of VNF *v* of service *s* on the node *n* is computed by constraint (16) considering the aggregated data to be processed by that VNF instance, while constraint (17) ensures that if the UE *u* uses that VNF instance ($\chi_{u,n}^i = 1$) then the VNF processing time $T_{proc}^i(u,n) = T_{proc}^i(n)$ is taken into account.

$$\forall n \in N, \forall s \in N_{ser}, \forall v \in N_{vnf}^s, \forall i \in N_{inst}^v:$$
$$\sum_{u \in \bar{N}_{ue}} \frac{\omega_{bwt}^u}{\mathcal{C}_{proc}^i(n)} \chi_{u,n}^i - T_{proc}^i(n) = 0 \tag{16}$$

$$\forall n \in N, \forall u \in \bar{N}_{ue}, \forall s \in N_{ser}^u, \forall v \in N_{vnf}^s, \forall i \in N_{inst}^v:$$
$$\mu * \chi_{u,n}^i + T_{proc}^i(n) - T_{proc}^i(u,n) \leq \mu \tag{17}$$

A similar approach is adopted by constraint (18) to compute the transmission time T_{tx}^e over the substrate link e, while constraint (19) handles the accurate transmission time computation over the virtual link $\bar{e}$.

$$\forall e \in E : \sum_{u \in \bar{N}_{ue}} \sum_{\bar{e} \in \bar{E}^u} \frac{\omega_{bwt}^u}{\mathcal{C}_{bwt}^e} \chi_e^{u,\bar{e}} - T_{tx}^e = 0 \tag{18}$$

$$\forall e \in E, \forall \bar{e} \in \bar{E}, \forall u \in \bar{N}_{ue}:$$
$$\mu * \chi_e^{u,\bar{e}} + T_{tx}^e - T_{tx}^{u,\bar{e}}(e) \leq \mu \tag{19}$$

Constraint (20) ensures that there is a continues path between the instance $i \in N_{inst}^v$ of the VNF $v \in N_{vnf}^s$ of service $s \in \bar{N}_{ser}^u$ requested by the UE $u \in \bar{N}_{ue}$.

$$\forall m, n \in N, \forall \bar{e} \in \bar{E}, \forall u \in \bar{N}_{ue}:$$
$$\sum_{e \in E^{n\rightarrow}} \chi_e^{e^{(n,m)}} - \sum_{e \in E^{\rightarrow n}} \chi_e^{e^{(n,m)}} = \begin{cases} -1 & \text{if } \ i = n \\ 1 & \text{if } \ i = m \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where $E^{n\rightarrow}$ represents the links originating from node $n \in N$, while $E^{\rightarrow n}$ represents all the links entering node $n \in N$.

The delay of a service $s \in N_{ser}$ is computed from the time the request is issued until the time the requested data is received by the UE. We consider the propagation delay, transmission delay, and the computing delay of all the VNFs $v \in N_{vnf}^s$ composing the service $s \in \bar{N}_{ser}^u$ requested by UE $u \in \bar{N}_{ue}$. Both the air interface delay and the transport link delay are taken into account in the calculation of the propagation and transmission delay. Constraint (21) guarantees that the aggregated delay does not exceed the maximum delay budget defined for the UE *u*:

$$\forall u \in \bar{N}_{ue} : \sum_{n \in N} \sum_{s \in \bar{N}_{ser}^u} \sum_{v \in N_{vnf}^s} \sum_{i \in N_{inst}^v} T_{proc}^i(u,n)$$
$$+ \sum_{e \in E} T_{tx,prp}^{u,\bar{e}}(e) + \sum_{g \in N_{gnb}} T_{tx,prp}^u(g) \leq T_{max}^u. \tag{21}$$

B. Heuristic

Although the MILP model achieves the optimal solution in all the scenarios, it becomes computationally intractable with the increase in the network size. Therefore, to combat the scalability issue of the MILP model, this section presents a heuristic algorithm, as shown in the algorithm (1), that

Algorithm 1: Mig_H

```
Input: $(G, \bar{G})$
Output: UEs association, VNF placement and resource allocation;
Phase 1: Find candidate gNBs for each UE;
for $u \in \bar{N}_{ue}$ do
    $cand\_gnb(u) \leftarrow \emptyset$;
    for $g \in N_{gnb}$ do
        $\omega^{u,g}_{prb} \leftarrow Calc\_PRB(u, g)$;
        if $\omega^{g}_{prb} \geq \omega^{u,g}_{prb}$ and $C^{u}_{g} \geq \omega^{u}_{bwt}$ then
            $cand\_gnb(u) \leftarrow g$;
Phase 2: Find the highest priority gNB and computing server for requests of each UE and then allocate the resources;
for $u \in \bar{N}_{ue}$ do
    for $v \in N^{s(u)}_{vnf}$ do
        for $i \in N^{v}_{inst}$ do
            for $n \in N$ do
                $serv\_prior[v, i, u, n] \leftarrow Calc\_Prior(v, i, u, n)$;
    $flag \leftarrow False$;
    • Sort the $cand\_gnb(u)$ in ascending order according to the # of PRBs;
    for $g \in cand\_gnb(u)$ do
        for $v, i, n \in serv\_prior[v, i, u, n] \downarrow$ do
            $T^{v,i}_{proc}(u, n) \leftarrow Calc\_Proc\_Delay(v, i, u, n)$;
            $T^{u,\bar{e}}_{tx,prp}(g, n) \leftarrow Calc\_Link\_Delay(u, \bar{e}, g, n)$;
            $T^{u}_{tx,prp}(g) \leftarrow Calc\_Air\_Delay(u, g)$;
            $T_{tot} \leftarrow T^{v,i}_{proc}(u, n) + T^{u,\bar{e}}_{tx,prp}(g, n) + T^{u}_{tx,prp}(g)$;
            if $T_{tot} \leq T^{u}_{max}$ then
                $flag \leftarrow True$;
                break;
        if $flag$ is $True$ then
            • Allocate path $P_{g,n}$;
            • Allocate and update network resources;
            break;
```

aims at reaching a near-optimal solution for the problem in a considerably shorter time.

Similar to the Mig_M algorithm, the objective of the proposed heuristic algorithm is to minimize the service interruption for the UEs by avoiding frequent changes in the serving nodes of the UEs and curtail the amount of state that should be exchanged for the UEs. The algorithm is divided into two phases. The first phase aims at finding the list of the candidate gNBs *cand_gnb*(*u*) for each UE *u*. A gNB *g* is considered to be a candidate for the UE *u* only if that gNB has the required amount of PRBs $\omega^{u,g}_{prb}$ computed by formula (3) and higher air interface capacity computed by formula (2) in order to support the data rate demand of the UE *u*. This phase of the algorithm is of order *O*(*mn*), in which *m* is the number of UEs and *n* is the number of gNBs.

The second phase of the algorithm attempts to find the highest priority gNBs and computing server for each request and allocate enough resources to accommodate the UE. As the first step, a 4D matrix (*serv_priority*[*v*, *i*, *u*, *n*]) is used to store each computing server's priority for hosting the instances of the VNFs that compose the requested service. The matrix is populated by a function called *Calc_Prior*(*v*, *i*, *u*, *n*) that gives a score to each combination of VNF, instance, UE, and computing node. The logic behind the *Calc_Prior*(*v*, *i*, *u*, *n*) function is to prioritize serving the UEs from VNFs at the same node compared to the previous run and associate the UEs to the same VNFs as before unless the UE requirement cannot be fulfilled with the current allocation that mostly happens due to the mobility of the UEs. The function computes the priority of embedding the VNFs of requested service with different instances on different nodes for each of the given UEs. There are several parameters involved in calculating the priority of embedding a VNF instance on a node for a specific UE. When a UE was assigned to a VNF instance on a specific node in the previous run, the same assignment will get the highest priority — if not, assigning the UE to an instance of the same VNF type embedded in the previous run, which did not serve the UE get the highest priority. Next, if in the current run a VNF is embedded, the aim will be to reuse the same VNF instance for the other UE in the same batch that asks for the same service type. The last priority is to embed the requested service type on a node with the highest resource capacity. It is worth noting that the number of CPU resources needed for VNF instantiation and the amount of bandwidth required on the links is considered in the priority calculation process for all the cases. The next step is to sort the candidate gNBs for each UE in an ascending order based on the number of PRBs required to associate the UE to the corresponding gNB. After that, for each candidate gNB, the algorithm loops over all the servers, starting from the one with the highest priority. The VNF processing delay on the node, transmission, and propagation delay over the transport link and air interface are computed in each run. If the overall delay of a placement solution is lower than the maximum delay tolerance of the UE, it will be considered as the best solution and break the loop to allocate the required resources to the request. This process is repeated until all the requests are embedded on the substrate network. As noted, finding a proper placement and allocation is the dominant procedure in the second phase; in this regards, the time complexity of this phase is of order *O*(*mnkpq*), where m, n, k, p, and q are, respectively, the number of UEs, gNBs, VNFs, instances, and computing nodes. It is worth mentioning that, in order to ensure the correctness of the solutions, we pass all the solutions found by the heuristic through the same constraints defined for the Mig_M formulation defined in Section IV. Overall, the complexity of the algorithm is $O(c_1 mn + c_2 mnkpq)$, where c_1 and c_2 are constants and negligible. Therefore, the complexity of the algorithm is of the order *O*(*mnkpq*).

V. Performance Evaluation

The goal of this section is to compare the presented MILP-based and heuristic algorithms. We shall first describe the simulation setup used in our study. We will then discuss the outcomes of the numerical simulations carried out in Python using Gurobi mathematical optimization solver [48].

A. Simulation Environment

The mobile network considered in this work is composed of 6 nodes, out of which one is a cloud server, one is a core server, and the rest are gNBs, referred to as edge nodes. All of the edge nodes and the core node have a collocated MEC server. The cloud server is connected to the core server via 4 Gbps BH link, whereas the edge nodes are connected to the core server via 1.5 Gbps FH links. The edge nodes, the core, and the cloud have, respectively, 4, 16, and 48 CPU cores, each of which has a 1.5 GHz clock rate. The capacity of a VNF instance depends on the number of allocated CPU cores. We assume that at least a single CPU core is required in order to spawn/instantiate a VNF. The maximum number of UEs that can use the same VNF depends on the service type and the number of CPU cores allocated to that VNF. More precisely, VNFs of the services that demand a higher security level are considered to be shared among fewer UEs because UEs can make security threats for each other. Thus, once a VNF is instantiated on a node, it can be used by a certain number of UEs under the condition of not violating the E2E latency of the UEs connected to the VNF instance.

Every minute, which is considered a single time slot, a new batch arrives composed of 5 UEs making a service request. Given that our model supports UEs' mobility, it is assumed that with the arrival of a new batch, UEs from the previous batches might change their locations by moving in random directions with speed selected from the set {5, 25, 50}km/h, mimicking pedestrians, cyclists, and cars, while still keeping their latency and data rate requirements. Moreover, we have considered state exchange costs as UEs change their VNF allocation from one serving node to another. Changing the serving node for a UE causes service interruption due to the fact that a process should be done to instantiate a new VNF instance on a destination node for the UE, associate the UE to the new VNF instance, and finally, transfer the state data of the UE from the source node to the destination node (the new serving node of the UE). In reality, the longer a UE uses a VNF instance, the more is its states and the more costly it is to transfer the states of that UE; therefore, we also consider the time (discrete number of runs that a UE has spent in the system). Here, we have considered the UEs' state of being 10% of their requested data rate. Upon receiving the service requests, the algorithms try to associate the UEs to the gNBs, place the VNFs of the requested service on the computing servers, and allocate enough resources to the spawned VNFs. We consider 18 batches of service requests (90 UEs in total) due to the scalability issue of the MILP-based algorithms. We assume that 3 types of service classes exist differentiated by their data rate and E2E delay tolerance (strict, medium, and loose) requirement. Examples of the services, together with their E2E delay tolerance and data rate requirements, are given in Table IV. If the UE association and its service request are accepted, the service provider must guarantee that the required data rate and the E2E delay tolerance are always satisfied.

For the sake of simplicity, for both downlink and uplink, the data size and data rate are considered to be the same. Although the transmission time interval (TTI) can be dynamically tuned in 5G networks, we consider $T_{tx} = 1ms$ as fixed TTI. The transmission time and processing time for each UE are computed considering all other UEs mapped, respectively, on the same FH/BH link and VNF. Specifically, T_{tx} for the UEs using the same FH or BH link at the considered moment is obtained by dividing the aggregated data size by the respective link rate. As for the processing time T^{i}_{proc} of a service/VNF, it is obtained by dividing the aggregated data demand on the VNF by the processing capacity of that VNF, which is the product of the number of CPU cores allocated to that VNF instance, clock rate of each CPU and the number of CPU cycles required to process one bit of information.

TABLE IV
SERVICE REQUIREMENTS

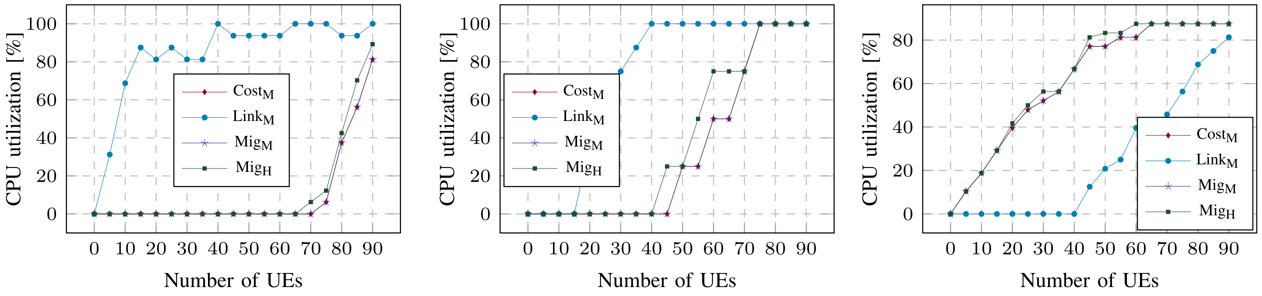
Service type	**E2E delay tolerance**	**Data rate requirement**
Webcasting	$2s$	$30Mbps$
See-through	$40ms$	$10Mbps$
IMA	$20ms$	$5Mbps$

B. Simulation Results

The reported results are the average of 5 simulations. The 95% confidence interval was never greater than 3% and thus, for the sake of improving readability, is not reported on the plots. During each simulation, the algorithms try to sequentially associate to the network and embed the service requests of up to 90 UEs, whose requests arrive in batches, each composed of 5 UEs. It is important to mention that all the algorithms employ a dynamic embedding strategy, that is, with the arrival of a new service request, the request along with the ones that have been previously embedded are re-embedded. Thus, with every embedding, the optimal embedding solution is found for all the UEs' requests.

Node and VNF utilization: CPU utilization and VNF utilization are two interrelated performance metrics, and investigating them provides a more comprehensive understanding of the performance of the proposed algorithms. While CPU utilization at nodes represents the number of CPUs assigned for the deployment of the VNFs on the nodes, VNF utilization exhibits how effectively the assigned resources are used by showing the number of UEs that employ the deployed VNFs. More precisely, CPU utilization at a node is the ratio between the number of CPU cores assigned to the deployed VNF instances and the maximum available CPU cores on that node. Regarding the VNF utilization, as previously mentioned, the number of UEs that can use a single CPU core depends on the service type and on the capacity of the VNF in terms the number of CPU cores assigned to that VNF instance. In our scenario, we consider that a single CPU can be used by 5 UEs for the webcasting and 3 UEs for the See-through and IMA services. Thus, the utilization of a VNF is the ratio between the number of UEs using that VNF and the maximum number of UEs that can use that VNF, which is the multiplication of the number of CPU cores of the VNF and the number of UEs that can use an instance of that VNF with a single CPU core.

Figures 3 and 4, show respectively, the CPU utilization and VNF utilization on all the computing nodes for a single simulation run. Figure 3(a) depicts the CPU utilization of the edges as a function of the number of UEs for all the algorithms. As can be inferred, the $Link_M$ algorithm begins the process of VNF placement by utilizing edge resources. This stems

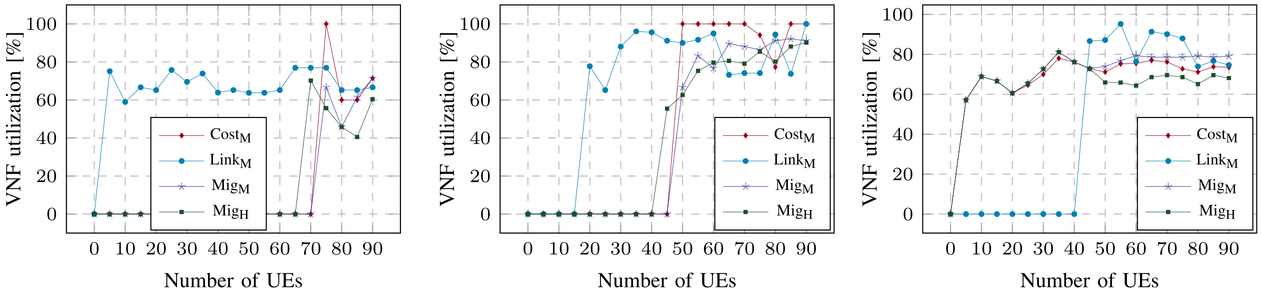


(a) Edge utilization.

(b) Core utilization.

(c) Cloud utilization.

Fig. 3. Node (CPU) utilization of edge, core and cloud nodes.



(a) VNF utilization at edge.

(b) VNF utilization at core.

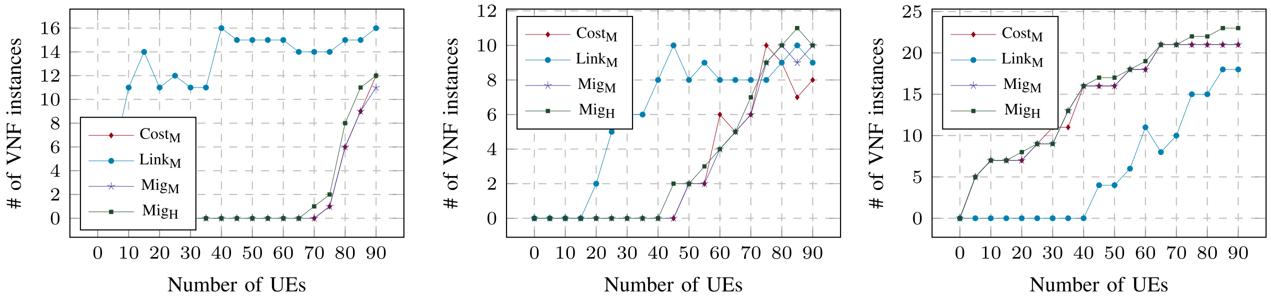
(c) VNF utilization at cloud.

Fig. 4. VNF utilization at edge, core and cloud nodes.

from the fact that the $Link_M$ algorithm aims to minimize the transport network utilization, which is achieved by embedding the service requests at the edge servers, which are the closest ones from the UEs' perspective. Figure 4(a) exhibits the same pattern for the VNF utilization. As can be inferred from the figure, with the arrival of the first batch, the VNF utilization reaches 80% due to the fact that there are not still many VNFs deployed, and UEs utilize a small set of VNFs. Later, with the arrival of new batches, the currently deployed VNFs cannot respond to the UE demands; therefore, new VNF instances are deployed, the load is distributed among VNFs, and consequently, the average VNF utilization of the VNFs decreases. Due to the scarcity of the processing resources at the edge, however, $Link_M$ shortly runs out of the edge resources and starts utilizing the core resources, as shown in Fig. 3(b). It is worth noting that average VNF utilization at the core (see Fig. 4(b)) is higher than at the edge, which stems from the fact that VNFs at the core are easily accessible through one FH link from the edge nodes in case the requested service is not present at their local premises, while accessing the same service on adjacent edge nodes requires usage of FH links. For what concerns the cloud resources (see Fig. 3(c)), we can observe that $Link_M$ starts embedding VNFs in the cloud when 45 UEs are making a service request, achieving the lowest CPU utilization. As expected, VNF utilization at the cloud (see Fig. 4(c)) gradually increases because the $Link_M$ algorithm always prefers not to use VNFs at the cloud as long as the requests can be fulfilled at the edge or core.

A reverse trend can be observed for the $Cost_M$ objective in terms of CPU utilization at the computing nodes. Specifically, it can be observed that $Cost_M$ tends to instantiate the VNFs starting from the cloud. This is due to significantly more processing resource available at the cloud compared to the edge and the core, which makes the total embedding cost much cheaper, regardless of the extra transport resource consumption. As expected, for the same reason, the CPU utilization at both edge and core is the smallest in most of the cases, in comparison with the ones achieved by the rest of the algorithms. As can be seen in 4(b) and 4(a) when a new VNF is embedded at the core or edge, the $Cost_M$ objective utilizes the maximum capacity of that VNF and even changes the previous VNF allocation of some of the UEs in order to reduce the cost of resources, which is the cost of link in this case. It is worth mentioning that the decrease in VNF utilization for the $Cost_M$ objective at the edge happens due to the mobility of the UEs.

As for the MILP-based and heuristic algorithms (i.e., Mig_M, Mig_H), their CPU and VNF utilization at the edge and cloud resembles the $Cost_M$ with a slightly more VNF utilization for the Mig_M in the long run. The reason for the higher VNF utilization at the cloud is that the Mig_M and Mig_H are trying not to let the UEs change their serving nodes aiming at minimizing service interruption and state exchange. More specifically, we can observe that similar to the $Cost_M$, Mig_M and Mig_H start by embedding VNFs at the cloud, but different from the cost objective, they tend to use the service from the same node in order not to trigger any state exchange and consequently to cause any service interruption for the UEs. Although similar to the $Cost_M$ algorithm, starting from 45 and 50 UEs, they start to employ core and later edge resources when it is necessary to serve some of the UEs in their proximity due to the increase in the transmission time over the FH/BH links (T_{tx}).



(a) Number of VNFs at edge.

(b) Number of VNFs at core.

(c) Number of VNFs at cloud.

Fig. 5. Number of VNF instances at edge, core and cloud nodes.

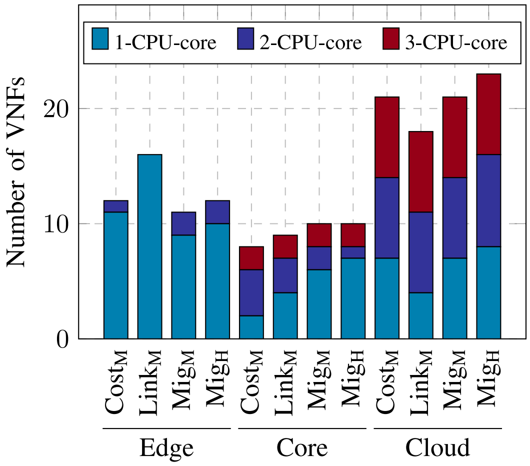


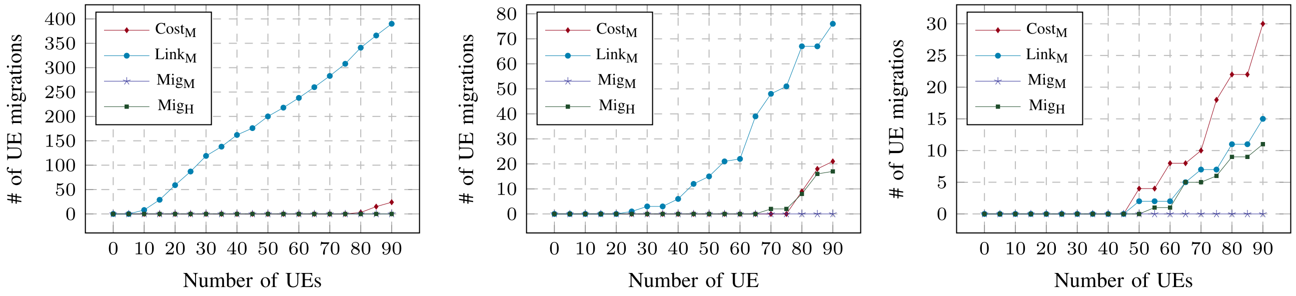
Fig. 6. Number of VNF instances with different capacity for all the algorithms at edge, core, and cloud.

Number of VNFs: VNF instantiation requires computing resources, which incurs higher management costs on the network. In order to get an insight into how are the VNFs distributed across the computing nodes, let us analyze Fig. 5, which shows the result of a single simulation run. We can observe that after the fourth batch embedding (20 UEs), $Link_M$ consumes the majority of the resources at the edges, employing most of their CPU cores. However, it is seen that still all the resources at the edge are not consumed with the $Link_M$ objective when it starts utilizing core resources. This stems from the fact that due to the UE mobility, it is more beneficial to serve some of the UEs from the core, which requires a single FH link from the UE, rather than employ two FH links to get the service from the edge. Although from this point, a lesser number of VNFs can be deployed at the edge, it does not restrict the UEs from using the VNF instances already available on the edges and, therefore, increase their utilization, as shown in Fig. 4(a). Similarly, $Link_M$ saturates also the CPU cores of the core node by instantiating 10 VNFs when there are 45 UEs making network association and service request; while, as expected, it utilizes a small portion of the cloud node by ultimately instantiating 16 VNFs (see Fig. 5(c)).

Regarding the Mig_M algorithm, it is interesting to note that, even though it achieves the least amount of CPU utilization at the core node up to 75 UEs (see Fig. 3(b)), it instantiates more VNFs at the core up to 90 UEs compared to both of the ($Link_M$, and $Cost_M$) algorithms, as displayed in Fig. 5(b). In essence, this means that Mig_M tries not to instantiate more VNFs on different nodes, which may lead the UEs to change their serving node and consequently need state data to be exchanged and impose service interruptions to the UEs. For what concerns to the number of VNFs instantiated by $Cost_M$ on the edges and cloud, plotted, respectively, in Fig. 5(a) and Fig. 5(c), they follow the same pattern of the CPU utilization at their corresponding nodes.

As for the MILP-based and heuristic migration algorithms, their performances resemble each other, especially in the cloud. In general, it can be observed that Mig_M and Mig_H utilize the VNF instances more efficiently compared to the rest of the algorithms since with the same number of VNFs; they achieve a higher CPU utilization in most of the cases. This is a consequence of the fact that Mig_M and Mig_H strives to continue with using the services on the same node and minimize the service interruption and its effect on the QoE of the UEs, leading to their higher utilization.

As stated before, the more CPU cores are assigned to a VNF instance, the more is its processing capacity, resulting in faster execution of UEs' tasks; nonetheless, the much more is also its instantiation cost. While Fig. 5 shows the total number of VNF instances across the edges, the core, and the cloud, it does not show the capacity of those VNFs. In order to have a better understanding of how the CPU cores of the computing nodes are allocated to the VNF instances and how many VNFs with different capacities are instantiated on the edges, the core, and the cloud, let us analyze Fig. 6. It can be observed that after all embeddings, mostly 1-CPU-core and rarely 2-CPU-core VNFs are instantiated on the edge nodes. This is due to the fact that the computational capacity of the edge nodes is very limited in comparison with the core and cloud nodes, and starting from the first embedding, it starts by instantiating 1-CPU-core VNFs, and then there are not enough resources to customize the CPUs assigned to the VNF. Therefore, the algorithms prefer to instantiate more VNF types on the edges to meet the E2E latency requirement of the UEs with various service/VNF requests rather than to instantiate a few of them with more computational capacity. For the core node, we observe that it gradually starts by embed 1-CPU-core VNFs, but since more resources are available compared to the core, then it increases the resources on VNFs, and at the end, we see some VNF even with 3-CPU-core instantiated at the core. Moreover, the

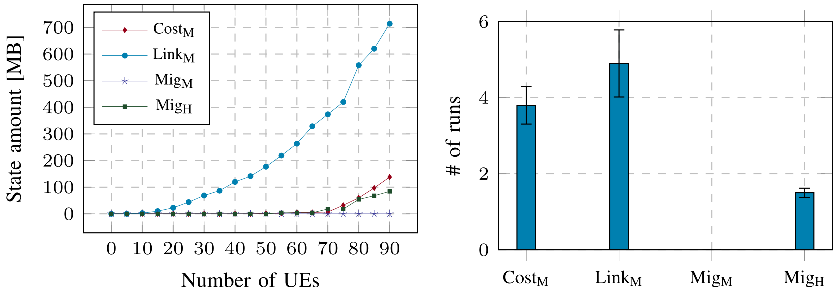


(a) Number of UE migrations from edge.

(b) Number of UE migrations from core.

(c) Number of UE migrations from cloud.

Fig. 7. Cumulative number of UEs changed their serving node from edge, core, and cloud.



(a) Cumulative amount of state exchange.

(b) Avg. UE presence when migrated

Fig. 8. Cumulative amount of state exchanged and average number of UEs presence in the system before change their VNF.

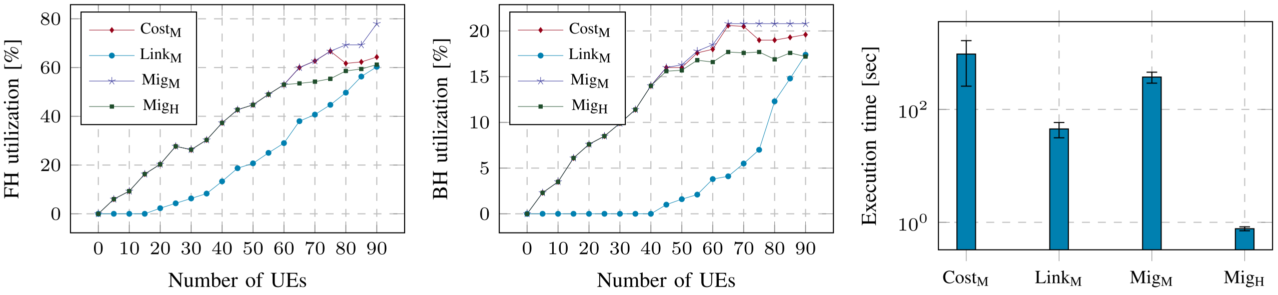
core is far closer to the UEs since it requires no BH resources, curtailing the E2E latency experienced by the UEs. As for the cloud node, we can observe that all the algorithms instantiate multiple VNF with 2-CPU-core and 3-CPU-core. The rationale behind this behavior is that the cloud node has plenty of CPU cores, which makes the VNF instantiation much cheaper. Moreover, in some cases, due to the extra transmission delay imposed by the transmissions in the FH and BH to reach the cloud, the algorithms have to increase the CPU resources of the VNF in order to decrease the processing delay somehow compromise the delay. As expected, among all the algorithms, the lowest number of VNFs with different capacities is instantiated by $Link_M$ since, as opposed to the rest of the algorithms, it always prefers to embed the VNFs at the edge as long as all of its constraints are satisfied.

Service Interruption: Clearly, service interruption is one of the prominent factors that negatively impact the QoE perceived by the UEs. Service interruption can happen due to several reasons including, UE mobility, VNF migration, and VNF consolidation. In this regard, as previously mentioned, one of the main objectives of this paper is to minimize service interruption by reducing the number of UEs that change their serving node. Moreover, when a UE changes its serving node, it is important to migrate the UE state from the hosting node (serving node) to the destination node. Furthermore, the longer a UE is in the system, the higher is its state amount. In this regard, when a system has to change the serving node of a set of UEs, it is preferable to move those with a lesser state.

As shown in Fig. 7, the $Link_M$ objective has the highest number of UEs that change their serving node both from the edge and core. While in the beginning, a few UEs change their serving node between the edge nodes due to the mobility, with the increase in the number of UEs and lack of resources at the edge, some of the VNFs and their associated UEs are migrated to the core. We observe from Fig. 8(a) that this algorithm also has the highest amount of state exchange. Moreover, due to the fact that $Link_M$ algorithm does not take into account the time that a UE has spent in the system before changing its serving node, it resulted in moving the services belonging to the UEs that spent a long time in the system (see Fig. 8(b)), which in return causes higher service interruption.

Regarding the $Cost_M$ objective, even though it causes some of the UEs to change their hosting node from the cloud, core, and very late at the edge, still it shows a better performance compared to the $Link_M$ objective. The better performance of the $Cost_M$ objective stems from the fact that it considers a cost value for each Mbps of the transferred state of the UEs. This behavior of the $Cost_M$ objective is proven by Fig. 8(a), which shows a very lower amount of state exchange compared to Fig. 8(a). Moreover, Fig. 8(b) shows that minimizing the amount of state exchange also positively affects the number of UEs that changed their serving node (see Fig. 8(b)).

Regarding the Mig_M and Mig_h objectives, Fig. 7 shows that Mig_M objective did not change the serving node of any of the UEs. In other words, this objective could reach minimum service interruption and highest UE satisfaction in this regard. This is because Mig_M objective gives a very high priority in minimizing service interruption especially when the UE has spent a longer time in the network. The Mig_H follows the same objective but in the end it has to change serving node of some of the UEs.



(a) Fronthaul link utilization.

(b) Backhaul link utilization.

(c) Execution time.

Fig. 9. FH and BH Link utilization in the entire network and execution time.

Link Utilization: Figure 9(a) and Fig. 9(b) illustrate, respectively, the FH and BH link utilization as a function of the number of UEs for a single simulation run. We can observe that $Link_M$ achieved the lowest FH and BH link utilization. This is justified by the fact that the other algorithms tends to utilize the cloud node resources as long as it does not violate the E2E latency constraints imposed by the service requests, therefore consuming also FH and BH link resources. Conversely, $Link_M$ aims at minimizing the transport network consumption in the network, therefore achieving the lowest FH and BH link utilization. For what concerns the $Cost_M$ objective, it experiences FH and BH utilization which is close to the one's of Mig_M algorithm when there are around 50 service requests; whereas, it get close to the $Link_M$ when the number of requests increases.

Execution Time: The main intention of the proposed heuristic algorithm is to combat the scalability issue of the MILP-based algorithms, which become computationally intractable when large substrate networks and more complex service requests composed of multiple VNFs are considered. The results given in Fig. 9(c) demonstrate the substantial improvement of the heuristic algorithm compared to its MILP-based counterparts in terms of execution time. Although the heuristic, due to its sub-optimal mapping solutions, performs poorer in terms of CPU utilization, service interruption, and link utilization, it proves to be competitive and also applicable to extensive size networks in real-world scenarios.

Figure 9(c) depicts the execution time of all the algorithms. It is obvious that the $Cost_M$ has much longer execution time compared to the other algorithms, which is due to the more parameters involved in the objective function. Moreover, the execution time of the Mig_M is higher than the $Link_M$. On the other hand, the execution time of the heuristic algorithm is much smaller, and it can reach a near-optimal solution in a matter of seconds.

VI. Conclusion

Network function virtualization is considered an essential enabler for next-generation mobile networks. The topics of user association and SFC placement have been studied extensively; however, the impact of user (re)association on SFC placement has not been studied so far. In this paper, we compared three strategies for solving a joint user association, SFC placement, and resource allocation problem in MEC-enabled 5G networks. The problem has been formulated as a MILP, and scalable heuristics have been proposed to find a near-optimal solution in a polynomial time. Based on the reported results, we can conclude that the proposed heuristic, Mig_H, is capable of finding the best trade-off between the computational capacity of the computing nodes and the FH/BH bandwidth, resulting in a negligible number of UE and VNF migrations. Moreover, at the expense of suboptimal UE associations and SFC placements compared to its MILP-based counterpart, Mig_H demonstrated the fastest execution time, making it suitable for larger-scale problems. As part of the future work we plan to extend the problem formulation to more distributed and heterogeneous MEC deployments. Moreover, we also plan to study how channel quality fluctuation and connectivity issues can affect the availability of a certain computational resource and to study the associate trade-offs.

Acknowledgment

The authors would like to acknowledge CERCA Programme/Generalitat de Catalunya for sponsoring part of this work.

References

[1] A. Gupta and R. K. Jha, "A survey of 5G network: Architecture and emerging technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.

[2] "View on 5G architecture," 5G PPP Archit. Working Group, Eur. Commission Eur. ICT Ind., White Paper, Jul. 2016.

[3] "MEC in 5G networks," Sophia Antipolis, France, Eur. Telecommun. Stand. Inst., White Paper, Jun. 2018.

[4] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Network function virtualization in 5G," *IEEE Commun. Mag.*, vol. 54, no. 4, pp. 84–91, Apr. 2016.

[5] *Toward Fully Connected Vehicles: Edge Computing for Advanced Automotive Communications*, 5G Automotive Assoc., Munich, Germany, Dec. 2017.

[6] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[7] R. Behravesh, E. C. Calero, D. Harutyunyan, and R. Riggio, "Joint user association and VNF placement for latency sensitive applications in 5G networks," in *Proc. IEEE CloudNet*, Coimbra, Portugal, 2019, pp. 1–7.

[8] D. Liu *et al.*, "User association in 5G networks: A survey and an outlook," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1018–1044, 2nd Quart., 2016.

[9] D. Liu, Y. Chen, K. K. Chai, and T. Zhang, "Nash bargaining solution based user association optimization in HetNets," in *Proc. IEEE CCNC*, Las Vegas, NV, USA, 2014, pp. 587–592.

[10] Y. Lei, G. Zhu, C. Shen, Y. Xu, and X. Zhang, "Delay-aware user association and power control for 5G heterogeneous network," *Mobile Netw. Appl.*, vol. 24, no. 2, pp. 491–503, 2019.

[11] M. Amine, A. Walid, A. Kobbane, and J. Ben-Othman, "New user association scheme based on multi-objective optimization for 5G ultra-dense multi-RAT HetNets," in *Proc. IEEE ICC*, Kansas City, MO, USA, 2018, pp. 1–6.

[12] A. S. Cacciapuoti, "Mobility-aware user association for 5G mmWave networks," *IEEE Access*, vol. 5, pp. 21497–21507, 2017.

[13] M. Amine, A. Kobbane, and J. Ben-Othman, "New network slicing scheme for UE association solution in 5G ultra dense HetNets," in *Proc. IEEE ICC*, Dublin, Ireland, 2020, pp. 1–6.

[14] S. Goyal, M. Mezzavilla, S. Rangan, S. S. Panwar, and M. Zorzi, "User association in 5G mmWave networks," in *Proc. IEEE WCNC*, San Francisco, CA, USA, 2017, pp. 1–6.

[15] D. Harutyunyan, A. Bradai, and R. Riggio, "Trade-offs in cache-enabled mobile networks," in *Proc. IEEE CNSM*, Rome, Italy, 2018, pp. 116–124.

[16] X. Ge, X. Li, H. Jin, J. Cheng, and V. C. M. Leung, "Joint user association and user scheduling for load balancing in heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, pp. 3211–3225, May 2018.

[17] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "Robust user association for ultra dense networks," in *Proc. IEEE INFOCOM*, Honolulu, HI, USA, 2018, pp. 2690–2698.

[18] A. Alleg, T. Ahmed, M. Mosbah, R. Riggio, and R. Boutaba, "Delay-aware VNF placement and chaining based on a flexible resource allocation approach," in *Proc. of IEEE CNSM*, Tokyo, Japan, 2017, pp. 1–7.

[19] Q. Zhang, F. Liu, and C. Zeng, "Adaptive interference-aware VNF placement for service-customized 5G network slices," in *Proc. IEEE INFOCOM*, Paris, France, 2019, pp. 2449–2457.

[20] S. Yang, F. Li, R. Yahyapour, and X. Fu, "Delay-sensitive and availability-aware virtual network function scheduling for NFV," *IEEE Trans. Services Comput.*, early access, Jul. 9, 2019, doi: 10.1109/TSC.2019.2927339.

[21] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar, and B. Akbari, "Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 1, pp. 374–388, Mar. 2019.

[22] L. Wang, Z. Lu, X. Wen, R. Knopp, and R. Gupta, "Joint optimization of service function chaining and resource allocation in network function virtualization," *IEEE Access*, vol. 4, pp. 8084–8094, 2016.

[23] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF placement and CPU allocation in 5G," in *Proc. IEEE INFOCOM*, Honolulu, HI, USA, 2018, pp. 1943–1951.

[24] Q. Zhang, Y. Xiao, F. Liu, J. C. Lui, J. Guo, and T. Wang, "Joint optimization of chain placement and request scheduling for network function virtualization," in *Proc. IEEE ICDCS*, Atlanta, GA, USA, 2017, pp. 731–741.

[25] Y. Bi, C. Colman-Meixner, R. Wang, F. Meng, R. Nejabati, and D. Simeonidou, "Resource allocation for ultra-low latency virtual network services in hierarchical 5G network," in *Proc. IEEE ICC*, Shanghai, China, 2019, pp. 1–7.

[26] D. Zhang, X. Lin, and X. Chen, "Multiple instances mapping of service function chain with parallel virtual network functions," *J. Algorithms Comput. Technol.*, vol. 13, Sep. 2019, Art. no. 1748302619868537.

[27] H. Moens and F. De Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *Proc. IEEE CNS*, San Francisco, CA, USA, 2014, pp. 418–423.

[28] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," in *Proc. IEEE CNSM*, Barcelona, Spain, 2015, pp. 50–56.

[29] M. Huang, W. Liang, Y. Ma, and S. Guo, "Throughput maximization of delay-sensitive request admissions via virtualized network function placements and migrations," in *Proc. IEEE ICC*, Kansas City, MO, USA, 2018, pp. 1–7.

[30] N. Kiran, X. Liu, S. Wang, and C. Yin, "VNF placement and resource allocation in SDN/NFV-enabled MEC networks," in *Proc. IEEE WCNCW*, Seoul, South Korea, 2020, pp. 1–6.

[31] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1409–1434, 2nd Quart., 2019.

[32] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[33] X. Li and C. Qian, "A survey of network function placement," in *Proc. IEEE CCNC*, Las Vegas, NV, USA, 2016, pp. 948–953.

[34] J. Xia, Z. Cai, and M. Xu, "Optimized virtual network functions migration for NFV," in *Proc. IEEE ICPADS*, Wuhan, China, 2016, pp. 340–346.

[35] D. Cho, J. Taheri, A. Y. Zomaya, and P. Bouvry, "Real-time virtual network function (VNF) migration toward low network latency in cloud environments," in *Proc. IEEE CLOUD*, Honolulu, HI, USA, 2017, pp. 798–801.

[36] F. Carpio, A. Jukan, and R. Pries, "Balancing the migration of virtual network functions with replications in data centers," in *Proc. IEEE/IFIP NOMS*, Taipei, Taiwan, 2018, pp. 1–8.

[37] H. Hawilo, M. Jammal, and A. Shami, "Orchestrating network function virtualization platform: Migration or re-instantiation?" in *Proc. IEEE CloudNet*, Prague, Czech Republic, 2017, pp. 35–40.

[38] I. Sarrigiannis, E. Kartsakli, K. Ramantas, A. Antonopoulos, and C. Verikoukis, "Application and network VNF migration in a MEC-enabled 5G architecture," in *Proc. IEEE CAMAD*, Barcelona, Spain, 2018, pp. 1–6.

[39] R. Cziva, C. Anagnostopoulos, and D. P. Pezaros, "Dynamic, latency-optimal VNF placement at the network edge," in *Proc. IEEE INFOCOM*, Honolulu, HI, USA, 2018, pp. 693–701.

[40] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency and mobility-aware service function chain placement in 5G networks," *IEEE Trans. Mobile Comput.*, early access, Oct. 1, 2020, doi: 10.1109/TMC.2020.3028216.

[41] "Intelligent transport systems (ITS); V2X applications," Sophia Antipolis, France, Eur. Telecommun. Stand. Inst., White Paper, Jun. 2018.

[42] M. Mehrabi, D. You, V. Latzko, H. Salah, M. Reisslein, and F. H. Fitzek, "Device-enhanced MEC: Multi-access edge computing (MEC) aided by end device computation and caching: A survey," *IEEE Access*, vol. 7, pp. 166079–166108, 2019.

[43] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Comput. Surveys*, vol. 51, no. 6, pp. 1–36, 2019.

[44] A. Chiumento, M. Bennis, C. Desset, L. Van der Perre, and S. Pollin, "Adaptive CSI and feedback estimation in LTE and beyond: A Gaussian process regression approach," *EURASIP J. Wireless Commun. Netw.*, vol. 2015, no. 1, p. 168, 2015.

[45] S. Sesia, I. Toufik, and M. Baker, *LTE-the UMTS Long Term Evolution: From Theory to Practice*. Chichester, U.K.: Wiley, 2011.

[46] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 206–219, Feb. 2012.

[47] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 1888–1906, 4th Quart., 2013.

[48] *Gurobi Mathematical Optimization Solver*. Accessed: Feb. 20, 2021. [Online]. Available: https://www.gurobi.com/

Rasoul Behravesh received the B.Sc. degree in IT from the Payam-e-Noor University of Mahabad in 2012, and the M.Sc. degree in computer networks from Qazvin Islamic Azad University, Iran, in 2016. He is currently pursuing the Ph.D. degree with the Smart Networks and Services (SENSE) Unit, Fondazione Bruno Kessler, Trento, Italy, and the Department of Electrical, Electronic, and Information Engineering, University of Bologna, Bologna, Italy. He is currently working on service management and orchestration in 5G networks. His main research interests include 5G networks, multiaccess edge computing, network function virtualization, and network slicing.

Davit Harutyunyan received the bachelor's and master's degrees (Hons.) in telecommunication engineering from the National Polytechnic University of Armenia in 2011 and 2015, respectively, and the Ph.D. degree (Hons.) in information and communication technology from the University of Trento in 2019. He was an Expert Researcher with the SENSE Research Unit, FBK. He is currently a Research Engineer in industrial 5G in the corporate research sector with Robert Bosch. He has published more than ten papers in internationally recognized journals/conferences. His main research interests include nonpublic mobile networks, next-generation radio access networks, multiaccess edge computing, and network slicing. He was a recipient of the Best Student Paper Award of IEEE CNSM 2017 and IEEE NetSoft 2019.

Estefanía Coronado (Member, IEEE) received the first M.Sc. degree in computer engineering, and the second M.Sc. degree in advanced computer technologies from the University of Castilla-La Mancha, Spain, in 2014 and 2015, respectively, and the Ph.D. degree in multimedia content delivery over SD-WLANs from the University of Castilla-La Mancha. She is a Senior Researcher with Fundació i2CAT, Spain. From 2018 to 2020, she was an Expert Researcher with FBK, Italy. She published around 35 papers in international journals and conferences. Her research interests include AI-driven network automation, wireless/mobile networks, network slicing, and SDN/NFV.

Roberto Riggio (Senior Member, IEEE) received the Ph.D. degree from the University of Trento, Italy. He is a Senior Researcher with Connected Intelligence Group, RISE AB, Stockholm, Sweden. He was a Postdoctoral Fellow with the University of Florida, a Researcher/Chief Scientist with CREATE-NET, Trento, the Head of Unit at FBK, Trento, and a Senior 5G Researcher with i2CAT Foundation, Barcelona, Spain. He has published more than 130 papers in internationally refereed journals and conferences. His current fields of applications are edge automation platforms, intelligent networks, and demand-attentive networking. His research interests revolve around optimization and algorithmic problems in networked and distributed systems.